Online Convex Optimization and its application to Online Portfolio Selection

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Puja Das

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTORATE OF PHILOSOPHY

ARINDAM BANERJEE

May, 2014

Acknowledgements

First, I would like to thank my advisor Prof. Arindam Banerjee for his invaluable support and guidance throughout my graduate study. His enthusiasm for research and and drive for excellence is infectious. Most of what I have learnt in machine learning, I owe it to him. I am truly grateful for his encouragement over the years.

I want to thank Prof. Daniel Boley, Prof. Snigdhansu Chatterjee, and Prof. Vipin Kumar for being a part of my committee and providing me valuable feedback and suggestions at various levels of my PhD, which has helped to shape my work. Moreover, I want to thank all the professors with whom I have interacted, within and outside classes, over the years.

I am indebted to IBM for supporting my research in the academic year 2013-14 through a fellowship. I want to thank Juhnyoung Lee for the opportunity to intern with the Business Solutions and Research group at IBM for two summers. I specially want to thank Stacy Hobson for being a wonderful mentor and overall being my friend and guide during my internships at IBM. I also want to thank Rong Liu for being my mentor, collaborating with me and providing me guidance during my time at IBM.

I want to thank Venkat Sunderanatha, Ranjan Sinha, and the entire Merchandising and Applied Research team at eBay. My internship experience at eBay helped both in my professional and personal growth.

I am grateful to Karen Monsen, with whom I have collaborated during my early PhD days. It was a great pleasure to work with her. I will miss my discussion sessions and interaction with Nicholas Johnson, Soumyadeep Chatterjee, Andre Goncalves and Vidyashankar Sivakumar, with whom I have closely collaborated during my PhD.

I would also like to thank my lab mates Amrudin Agovic, Hanhuai Shan, Qiang Fu, Soumyadeep Chatterjee, Huahua Wang, Amir Taheri, Karthik Subbian, Farideh

Fayzeli, Igor Melnyk, Konstantina Christakopoulou and Vidyashankar Sivakumar. It was a great experience, being able able to interact and share lab space with them.

I am grateful for my wonderful friends: Sounak Basu, Sauptik Dhar, Ayan Paul, Sohini Roychowdhury, Sanjoy Dey, Soumyadeep Chatterjee, Shameek Bose, Somnath Kundu, Kaushik Basu who formed the backbone of my support system in Minneapolis. I will miss our weekend "adda". I will always fondly remember BSSM and the people associated with it for giving me such wonderful memories to cherish.

Finally, I want to thank my Dad, Mom and Bonnie for their unconditional support and for having faith in me always. I couldn't have done it without them.

Dedication

To my parents and sister.

Abstract

Today, whether we consider the data from the internet, consumers, financial markets, a common feature emerges: all of them involve huge amounts of dynamic data that arrive sequentially and need to be understood and processed quickly. Online learning is concerned with the task of making decisions on-the-fly as observations are received. Online learning has attracted a lot of attention due to the recent emergence of large-scale applications such as online web advertisement placement, online topic-detection in social communities, online web ranking, finding shortest path for internet packet routing, email spam filtering, portfolio selection and many more. In recent years, tools from convex optimization have influenced the design of many online learning algorithms. As a result, Online Convex Optimization (OCO) has emerged as a unified abstraction, which helps in solving problems efficiently and reliably and also facilitates the theoretical analysis. In this thesis, we contribute to the development of OCO and present solutions to several complex problems motivated by real world applications.

Although OCO has been studied widely, one drawback of the existing algorithms is the cost of updating model parameters with every incoming data point, especially when the number of parameters is in the order of millions or billions. An example is online portfolio selection, where changing ones portfolio aggressively everyday will incur huge amount of transaction costs. In the first part of the thesis, we present an algorithm that performs *lazy updates* to the parameters and show that its performance is competitive with optimal strategies which have the benefit of hindsight. The resulting convex optimization problem is non-smooth, and we use an efficient primal-dual based alternating direction method of multipliers (ADMM) to carry out lazy updates in every step. We successfully establish the effectiveness of our online lazy updates algorithm for online portfolio selection with transaction costs with experiments on real world datasets.

Several machine learning algorithms use iterative optimization methods for learning predictive models. It is difficult to know upfront, which of these routines will converge faster or give the best possible solution. This is a common problem in online portfolio selection, where there could be a number of heuristics or theoretically motivated algorithms which suggest portfolios for each day. It is not possible for an investor to

know which of these portfolio selection algorithms will make the most amount of money in a given market. In the second part of the thesis, we present two Meta Algorithms (MAs), which work by adaptively combining iterates from a pool of base optimization algorithms. We show that the performance of the MAs are competitive with the best convex combination of the iterates from the base algorithms. We illustrate the effectiveness of MAs on the problem of portfolio selection in the stock market.

OCO has emerged as a powerful large scale optimization approach, but much of the existing literature assumes a simple way to project onto the constraint set. This assumption is often not true, and the projection step can become the key computational bottleneck. Motivated by applications in risk-adjusted portfolio selection, we consider online quadratically constrained convex optimization (QCCO) problems, where the constraint set involves intersection of multiple quadratic and linear constraints. We show that the algorithm for solving online QCCOs has theoretical performance guarantees and can be posed as a quadratically constrained quadratic program (QCQP) in each step. We present an efficient algorithm for solving QCQPs based on ADMM. We show that risk adjusted meta portfolio selection is a special case of our general framework. With extensive experiments on two real world stock datasets, we establish that our algorithm RAMP is either competitive or can achieve significantly greater wealth than existing approaches at any given risk level.

A network is often defined as a collection of variables which are inter-dependent through a complex dependency structure. In many scenarios, the structure of the network is not known beforehand and the problem of interest lies in estimating the structure based on sample data from the network variables. Examples include social networks, communication system between operators, user interaction in social media, etc. Moreover, in many domains, the network structure is not static and can change over time. For example, the relationship structure amongst stocks may change significantly over years depending on the companies financial developments and international economic policies. In the final part of the thesis, we present an online convex optimization model for estimation of the dependency structure of a time-varying network, when the network variables follow a multivariate Gaussian distribution. Under this assumption, the conditional independence structure of the network is encoded in a sequence of precision matrices. We present an ADMM based algorithm for tracking dynamic networks and we

establish theoretical performance guarantees of our method with respect to batch methods which have the power of hindsight. With experiments over synthetic datasets and a real financial dataset, we establish that our algorithm can identify smoothly varying or abruptly evolving network structures even when the sample size is small.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Today, whether we consider the data from the internet, consumers, financial markets, a common feature emerges: all of them involve huge amounts of dynamic data that arrive sequentially and need to be understood and processed quickly. Online learning is concerned with the task of making decisions on-the-fly as observations are received. Online learning has attracted a lot of attention due to the recent emergence of large-scale applications such as online web advertisement placement, online topic-detection in social communities, online web ranking, finding shortest path for internet packet routing, email spam filtering, portfolio selection and many more. While online learning can be used for temporal data, its stochastic counterpart can be used for large-scale learning tasks where we can treat the training data as a stream, and the requirement is to process each data object only once. In particular, one can solve a batch problem by processing one/mini-batch of data points at a time e.g., in image classification, text categorization, bioinformatics (protein classification, cancer classification), etc. Online algorithms and their stochastic counterparts, have two key properties. Firstly, they are computationally efficient. Each step has no dependency on the data size and we do not need to store the entire training set in memory. The total number of steps is of the same order as of the number of examples. Secondly, they maintain theoretical performance guarantees of being comparable to their batch counterparts, which have access to the entire data. Hence, the study of online learning algorithms is an increasingly important area in machine learning, and one that has interesting theoretical properties and practical applications. In recent years, tools from convex optimization (a special

class of mathematical optimization problems) have influenced the design of many online learning algorithms. As a result, Online Convex Optimization (OCO) has emerged as a unified abstraction, which helps in solving problems efficiently and reliably and also facilitates the theoretical analysis.

1.1 Why Online Convex Optimization?

With the ever growing amount of data and the emphasis on scalable machine learning algorithms in recent times, we are concerned with machine learning over large data sets. In this setting, many common approaches fail, simply because they cannot load the data set into memory or they are not sufficiently efficient. We look at a few examples where online convex optimization is applied.

1. Social media, such as twitter, have become pervasive today. Twitter receives over 400 million tweets per day from its users. As microblogging sites such as twitter gained popularity, a myriad of applications that perform trend analysis on data have emerged. An example of one such task is the automatic identification of breaking news from the twitter stream. This requires detection of novel tweets from a voluminous stream of texts in a scalable manner. The high volume and velocity of data on microblogging sites like twitter, make them an ideal candidate for online learning, particularly online convex optimization.

2. Most real work networks are dynamic and evolve over time. The delays in overlay networks used in peer-to-peer applications change unpredictably as the load in the underlay network fluctuates. Since multiple paths can exist between any two nodes in a network, it is important to ascertain the path with the minimum delay, while routing packets between two nodes at any given time. Moreover, the shortest path in terms of delay between two node can change over time. However, the optimal path, can only be estimated by routing packets through the network and observing the realized delays. This problem is referred to the online shortest path and a well known application of online convex optimization framework.

3. Supervised learning on large data sets with million of data points for e.g. logistic regression, SVM classification can be computationally expensive. For example,

the google brain consists of 20 million images. Iterative methods which run over the entire dataset usually have a runtime which depends on the size of the data, and hence the per iteration complexity becomes a computational bottleneck. The alternative is to consider one data point at a time for processing, by treating the huge dataset as a stream for online convex optimization. This reduces the per iteration complexity but might increase the number of iterations for the optimization algorithms.

4. In financial data analysis, data also arrives in a stream. One particularly important problem in this context is portfolio selection. Here, the objective is to sequentially select portfolios: as a distribution over a set of assets. At the beginning of everyday, an investor has to choose a portfolio without knowing a priori the performance of the stocks for that particular day. At the end of the day, the investors gets to observe the performance of each of the assets, and incurs the gain or loss for the portfolio he invested with. The objective is then to maximize the multiplicative gain in wealth at the end of multiple periods/days. This problem is popularly known as online portfolio selection.

OCO has applications in a wide variety of domains, and an important application of OCO in computational finance is the online portfolio selection in the stock market (discussed above). Online portfolio selection has been a success story [36, 61, 32, 2, 18, 40, 82, 41, 85] over the last two decades. Online portfolio selection algorithms modeled in the OCO framework make no statistical assumptions regarding the movement of stocks [36, 37, 61] and in a well-defined technical sense are guaranteed to perform competitively with certain families of adaptive portfolios even in an adversarial market. In this thesis we particularly focus on online portfolio selection as an illustration of our key contributions to OCO.

1.2 Motivation

The aim of this thesis is to address the following problems using complex real world applications as illustrative examples. The focus is particularly on online portfolio selection.

1. With the ever increasing amount of data, particularly from search engines and social networks, stochastic optimization algorithms have become desirable for large-scale machine learning tasks because of their empirical efficiency and strong theoretical guarantees [19, 20, 12]. However, a major challenge that is encountered is the cost of updating model parameters especially when the number of parameters can be in the order of millions to billions. Often times when parameters are updated frequently, their values do not change significantly. As such, the cost of updating each parameter starts to outweigh the benefits.

   An important and relevant application where changing the model parameters might prove to be monetarily expensive is the domain of online portfolio selection. Here every time an investor changes his portfolio, he ends up buying or selling his stocks and incurring transaction costs. Trading aggressively might sometimes hurt an investor instead of proving to be beneficial.

   Hence, an important question in this context is, whether we can do *lazy updates* to parameters and still maintain theoretical performance guarantees.

2. While online convex optimization has emerged as a powerful large scale optimization approach, for constrained optimization problems, which forms the basis for learning most widely used models in data mining, the literature on online convex optimization is not as mature. Much of existing literature assumes a simple way to project onto a given feasible set. The assumption is often not true, and the projection step usually becomes the key computational bottleneck for every iteration. In practice, performing a projection onto a given feasible set is an excruciatingly time consuming process especially if the number of constraints in the feasible set is large.

   For online portfolio selection, modeling the risk of the portfolio based on Markowitz's mean-variance framework can show up as a constraint in the online convex optimization setting. In particular, the constraint set is an intersection of ellipsoidal constraints and hyperplanes which is not straightforward to solve.

   Hence, an interesting technical question is, whether it is possible to design algorithms which can handle time varying complex constraint sets (linear and quadratic constraints) efficiently, in an online convex optimization setting.

3. Several machine learning algorithms use iterative optimization methods for learning predictive models. It is not easy to determine upfront which optimization method will perform best or converge fast for such tasks. Typically, there are several choices for iterative update methods including gradient based or Newton step based optimization routines, stochastic gradient descent algorithms, domain specific methods, evolutionary and genetic algorithms, or plain heuristics.

   For online portfolio selection problem, oftentimes heuristics like Anticor [18], OLMAR [81] outperform theoretically motivated algorithms like EG and ONS, in terms of empirical performance. Although, the heuristics perform well on certain datasets, one cannot provide any guarantees regarding their performance in all possible market conditions. So it is not easy for an interested investor to decide upfront which algorithm to invest with.

   In this context, a key question which arises is, if iterates/solutions from multiple algorithms for the same problem can be meaningfully combined to guarantee good optimization performance. Ideally, one would like the combined iterates to outperform the best algorithm in the pool.

4. There is a growing desire for efficient algorithms that can identify and take advantage of underlying structure within data for predictive tasks. While for certain problems the structure is pre-specified, for others the structure is unknown and might even change over time. Examples include learning dependency structures in a wide variety of domains, such as natural and social sciences, stock markets, and the world wide web, etc. An example in social media is the network of user interactions, which can constantly evolve over time. OCO being an efficient way for predictive tasks which have to be carried out on-the-fly, makes it an ideal candidate for learning complex structures which are time variant.

   Investors are often interested in identifying groups of well performing stocks rather than isolated analysis of individual stocks. Hence, taking advantage of the group structure for the stocks is a promising direction for online portfolio selection. Moreover, the dependency structure between stocks could easily change over time.

   Hence, an important question which arises is, if its possible to learn structures which can vary over time, in an efficient manner. In particular, we are interested

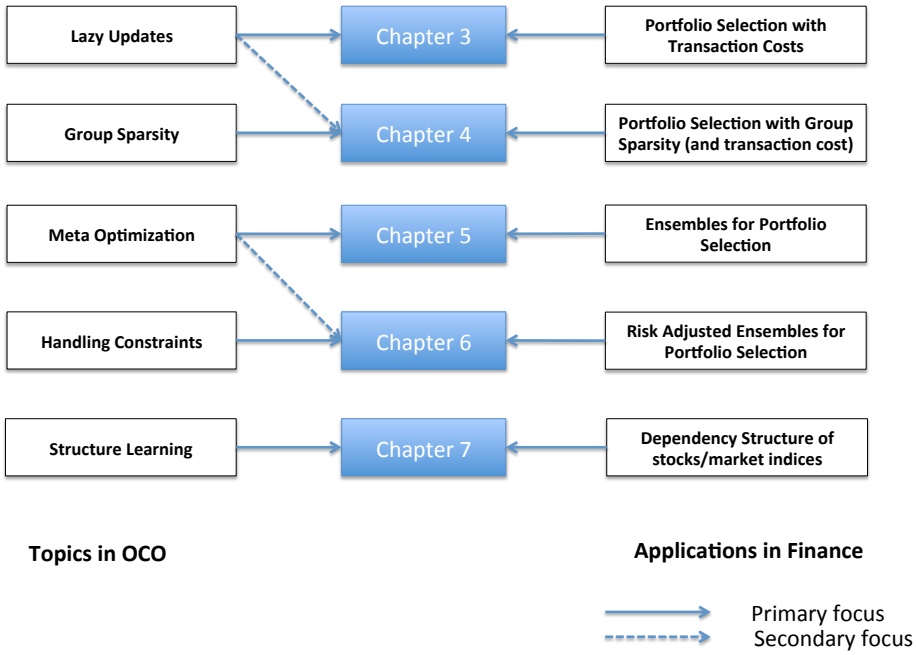


Figure 1.1: Overview of the thesis organization. Each chapter deals with one or many topics in OCO and can be used to solve key problems for online portfolio selection.

in algorithms which can track structures in an online fashion and have theoretical performance guarantees compared to their batch counterparts (which can observe the entire time-series data in hindsight).

1.3 Overview

We briefly discuss the outline of the thesis here and Figure 1.1 gives an overview of how the chapters are organized and what topics and themes they constitute.

1.3.1 Related Work

In Chapter 2, we discuss related work. We briefly cover the following three topics: (i) Online Convex Optimization (OCO), (ii) Online Portfolio Selection and (iii) Alternating Direction Method of Multipliers (ADMM). We present the basics of OCO, describing the

framework, the key algorithms and formally introduce *regret*: the theoretical measure of performance for OCO algorithms. As mentioned earlier, since we use online portfolio selection as the chosen application for OCO for demonstration of our contributions, we briefly survey the existing work in this domain. Finally, we discuss ADMM, which we will use to solve many of the composite objective convex optimization problems (smooth and non-smooth) with constraints.

1.3.2 Online Lazy Updates

A major challenge for stochastic optimization is the cost of updating model parameters especially when the number of parameters is very large. Updating parameters frequently can prove to be computationally or monetarily expensive. In Chapter 3, we propose sparse updates to the parameters by introducing an ℓ_1 penalty on the updates to parameters over consecutive iterations. We introduce an efficient primal-dual based online algorithm that performs *lazy updates* to the parameter vectors and show that its performance is competitive with reasonable strategies which have the benefit of hindsight. We demonstrate the effectiveness of our algorithm in the online portfolio selection domain where a trader has to pay proportional transaction costs every time his portfolio is updated. Our Online Lazy Updates (OLU) algorithm takes into account the transaction costs while evaluating an optimal portfolio and results in sparse updates to the portfolio vector. We successfully establish the robustness and scalability of our lazy portfolio selection algorithm with extensive theoretical and experimental results on two real world datasets.

1.3.3 Online Lazy Updates with Group Sparsity

In portfolio selection, it might often be preferable to focus on a few top performing industries/sectors to beat the market. These top performing sectors however might change over time. In Chapte 4, we propose an online portfolio selection algorithm that can take advantage of sector information through the use of a group sparsity inducing regularizer while making lazy updates to the portfolio. The lazy updates prevent changing ones portfolio too often which otherwise might incur huge transaction costs. The proposed formulation is not straightforward to solve due to the presence of

non-smooth functions along with the constraint that the portfolios have to lie within a probability simplex. We propose an efficient primal-dual based alternating direction method of multipliers algorithm, and demonstrate its effectiveness for the problem of online portfolio selection with sector information. We show that our algorithm OLU-GS is competitive with reasonable strategies which have the benefit of hindsight, through theoretical analysis. We successfully establish the robustness and empirical benefits of OLU-GS by performing extensive experiments on two real-world datasets for portfolio selection.

1.3.4 Meta Optimization

Several machine learning algorithms use iterative optimization methods for learning predictive models. It is not easy to determine upfront which optimization method will perform best or converge fast for such tasks. In Chapter 5, we analyze Meta Algorithms (MAs) which work by adaptively combining iterates from a pool of base optimization algorithms. We show that the performance of MAs are competitive with the best convex combination of the iterates from the base algorithms for online as well as batch convex optimization problems. We illustrate the effectiveness of MAs on the problem of portfolio selection in the stock market and use several existing ideas for portfolio selection as base algorithms. Using daily data for the past 21 years for two datasets, we show that MAs outperform existing portfolio selection algorithms with provable guarantees by several orders of magnitude, and match the performance of the best heuristics in the pool.

1.3.5 Constrained OCO

While online convex optimization has emerged as a powerful large scale optimization approach, much of existing literature assumes a simple way to project onto a given feasible set. The assumption is often not true, and the projection step usually becomes the key computational bottleneck. Motivated by applications in risk-adjusted portfolio selection, in Chpater 6 we consider online quadratically constrained convex optimization problems, where the feasible set involves intersections of ellipsoids. We show that regret guarantees for the online problem can be achieved by solving a suitable quadratically

constrained quadratic program (QCQP) at each step, and present an efficient algorithm for solving QCQPs based on the alternating directions method. We then specialize the general framework to risk adjusted ensembles portfolio selection. Through extensive experiments on two real world stock datasets, our proposed algorithm RAMP, a risk adjusted ensemble is shown to significantly outperform existing approaches at any given risk level and match the performance of best heuristics which do not accommodate risk constraints.

1.3.6 Dynamic Gaussian Networks

Estimation of network structure from observations over nodes of a network is an area of active research in various domains. A relatively recent approach for the estimation problem is to assume that the variables follow a multivariate Gaussian distribution, and estimate its conditional dependence structure. Although there exist several methods for estimation when samples are obtained from a static distribution, estimation of the conditional dependance structure of a time varying distribution is a more difficult problem to solve, and there exist only a few methods which can address this problem. In Chapter 7, we propose an online convex optimization model for estimation of the dependency structure of a time-varying Gaussian distribution. We do not make any statistical assumptions on the temporal evolution of the distribution, and establish theoretical performance guarantees of our method, *DyGN*, with respect to batch methods, which have the power of hindsight. We illustrate through experiments on synthetic datasets that DyGN can accurately estimate abruptly evolving network structures from even when the sample size is small. Further experiments on real financial data illustrate that DyGN can track smoothly varying structures that capture interesting dynamics present in the data.

1.4 Notation

Unless otherwise specified: lower-case bold alphabets, e.g. $\mathbf{x}, \mathbf{y}$ are used to represent vectors. Sets are represented by calligraphic upper-case alphabets, e.g. $\mathcal{X}, \mathcal{Y}$. Random variables are represented by upper-case alphabets, e.g., X, Y. Bold upper-case alphabets are used to represent matrices, e.g., $\mathbf{X}, \mathbf{Y}$. The symbols $\mathbb{R}$ and $\mathbb{R}^n$ denote the set

of reals and the n-dimensional vector space respectively.Further, $\mathbb{R}_{++}$ denotes the set of positive real numbers. $\triangle_n$ denotes the probability simplex of dimension n. For $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|$ denotes the ℓ_2 norm, while $\|\mathbf{x}\|_p$ denotes the p-norm where $p > 0$. For $\mathbf{X} \in \mathbb{R}^{m \times n}$, $\|\mathbf{X}\|_F$, denotes the Froebenius norm of $\mathbf{X}$ and $\mathrm{Tr}(\mathbf{X})$ demotes the trace of the matrix. For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\langle \mathbf{x}, \mathbf{y} \rangle$ denotes the inner product. The domain of a function f is specified as $\mathrm{dom}(f)$ and the inverse of f when well specified as f^{-1}. ∇f and f' are used interchangeably to denote the gradient of the function f, while ∂f denotes the subgradient of f. S_ρ denotes the soft thresholding operator with parameter ρ as defined in [22]. $\prod_{\triangle_n}$ denotes the euclidean projection operator to the probability simplex of dimension n. $\prod_{\triangle_n}^{A}$ denotes the projection operator to the probability simplex according to the norm induced by A.

Chapter 2

Related Work

In this thesis we focus on a class of convex optimization problem with linear and (or) quadratic constraints and the convex objective function changes over time. The convex objective can be composite with smooth and non-smooth parts. Online convex optimization (OCO) has been generalized to handle time-varying and non-smooth convex functions. Alternating direction method of multipliers on the other hand, has become widely popular for solving a variety of problems including composite objectives and are specialized to handle linear equality constraints, which makes distributed optimization with variable splitting straightforward. Online portfolio selection has emerged as an important application of the OCO setting and will be used as the main illustrative application in this thesis. This chapter contains a brief discussion on these three key topics: (i) Online Convex Optmization, (ii) Online Portfolio Selection and (iii) Alternating Direction Method of Multipliers (ADMM). The chapter is divided into three main sections, each one appropriately dedicated to one of the above mentioned topics.

In Section 2.1, we discuss the Online Convex Optimization model and introduce *regret* the theoretical measure of performance in this setting. We then go on to describe popular first-order algorithms and a second-order algorithm for solving problems in the OCO setting. Amongst the first-order methods, we discuss Online Mirror Descent and Composite Objective Mirror Descent. Moreover, we show that the Online Gradient Descent and Exponentiated Gradient algorithms are special cases of the Online Mirror Descent algorithm. We also discuss the Online Newton Step method which is a second order method.

In Section 2.2 we focus on a real world problem in computational finance that can be cast in the OCO framework. We describe the online portfolio selection problem and discuss two families of algorithms which constitute the portfolio selection literature. The first family of methods have theoretical performance guarantees, i.e., they have a meaningful *regret* bound. The second family of methods are heuristics which have no performance guarantees but often outperform the theoretically motivated algorithms, empirically. We discuss first order and second order theoretically motivated algorithms, and focus on a particular class of heuristics which follow a strategy, popularly known as *mean reversion* in finance.

In Section 2.3, we discuss Alternating Direction Method of Multipliers (ADMM), which has been applied to many large scale problems in statistics and machine learning because of its computational benefits and fast convergence in practice. In this thesis, we have used ADMM to solve many of the non-smooth, constrained optimization problems in the OCO setting. We give a brief introduction to the general ADMM and discuss two recently proposed versions: the Bregman ADMM and the Online ADMM. While Bregman ADMM generalizes ADMM by replacing the quadratic penalty with a Bregman divergence, Online ADMM proposes a single pass algorithm to handle composite objectives with constraints in the online setting.

2.1 Online Convex Optimization

Online learning [32, 87, 31] has been studied in several research fields including game theory and machine learning. The goal of online learning is to make a sequence of decisions given the knowledge only of the previous tasks and their outcomes. In recent years, the design of many efficient online learning algorithms has been influenced by convex optimization tools. We give an overview of OCO framework in this section which we will refer to in the subsequent sections and also the rest of the thesis. We start with a brief recap of some of the key concepts used in convex optimization.

2.1.1 Convex Optimization

Convex optimization [23, 66, 102] is a special class of mathematical optimization problems which includes least-squares and linear programming problems. One of advantages

of formulating a problem as a convex optimization problem is that it can be solved reliably and efficiently by tools and methods designed specially for convex optimization problems. Convex optimization plays as key role in this thesis and below we discuss a few important properties related to this class of optimization.

A set $\mathcal{X}$ is convex if the line segment between any two points in $\mathcal{X}$ lies in $\mathcal{X}$, i.e., if for any $\mathbf{x}$, $\tilde{\mathbf{x}} \in \mathcal{X}$ and any θ with $0 \leq \theta \leq 1$, we have

$$\theta\mathbf{x} + (1-\theta)\tilde{\mathbf{x}} \in \mathcal{X}. \tag{2.1}$$

Let $\mathcal{X}$ be a convex set and $f : \mathcal{X} \mapsto \mathbb{R}$ be a function, then f is convex if $\forall \mathbf{x}, \tilde{\mathbf{x}} \in \mathcal{X}$, $\forall \alpha \in [0,1]$

$$f(\alpha\mathbf{x} + (1-\alpha)\tilde{\mathbf{x}}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\tilde{\mathbf{x}}). \tag{2.2}$$

Let the domain of f, i.e., $\mathrm{dom}(f)$ be denoted as S. The sub-differential set $\partial f(\mathbf{x}) : g \in \partial f(\mathbf{x})$ if

$$f(\tilde{\mathbf{x}}) \geq f(\mathbf{x}) + \langle \tilde{\mathbf{x}} - \mathbf{x}, g \rangle, \quad \forall \tilde{\mathbf{x}} \in S. \tag{2.3}$$

The sub-differential set $\partial f(\mathbf{x})$ is closed convex, even if f is not convex and each $g \in \partial f(\mathbf{x})$ is a sub-gradient. In this thesis we are interested in both: smooth as well as non-smooth convex functions.

2.1.1.1 Non-smooth functions

In many machine learning problems, the convex functions that we come across are non-smooth [101, 13, 14], e.g., hinge loss, ℓ_1-norm, etc. A non-smooth function is convex if $\partial f(\mathbf{x}) \neq \emptyset, \forall \mathbf{x} \in S$. For convex f, $\partial f(\mathbf{x})$ is non-empty, convex and compact. Moreover, for Lipschitz convex functions f on domain S, we have

1. f has a minimizer $\mathbf{x}^*$ in S,
2. f is convex on S, and
3. f is G-Lipschitz on S, i.e., for any $g \in \partial f(\mathbf{x})$, we have for $\|g\| \in \partial f(\mathbf{x})$, $\|g\| \leq G$.

2.1.1.2 Smooth functions

If f is convex and differentiable at $\mathbf{x}$, then $\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}$, i.e., its gradient is its only sub gradient. Hence, for a smooth convex function f [102, 14], with $\mathrm{dom}(f)$ as S, we can summarize:

1. f has a minimizer $\mathbf{x}^*$ in S,
2. f is convex and continuously differentiable on S, and
3. f is smooth, i.e., gradient ∇f is β-Lipschitz: $\forall \mathbf{x}, \tilde{\mathbf{x}} \in S$,

$$\|\nabla f(\mathbf{x}) - \nabla f(\tilde{\mathbf{x}})\| \leq \beta \|\mathbf{x} - \tilde{\mathbf{x}}\|. \tag{2.4}$$

2.1.1.3 Strongly convex functions

A function is *strongly* convex [14] with parameter $\lambda > 0$, if $\forall \mathbf{x}, \tilde{\mathbf{x}} \in S$,

$$(\nabla f(\mathbf{x}) - \nabla f(\tilde{\mathbf{x}}))^T (\mathbf{x} - \tilde{x}) \geq \lambda \|\mathbf{x} - \tilde{\mathbf{x}}\|^2. \tag{2.5}$$

2.1.1.4 Bregman Divergence

Bregman divergences [24, 29] are a class of distortion functions derived from convex functions that play a role in this thesis. Given any differentiable strictly convex function ϕ, the Bregman divergence between $\mathbf{x}$ and $\mathbf{y}$ is defined as

$$d_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla \phi(y) \rangle, \tag{2.6}$$

where $\nabla \phi(\mathbf{x})$ is the gradient of ϕ at $\mathbf{x}$. Bregman divergences have gained popularity in the machine learning literature over the past few years since they allow the unified analyses of a large class of algorithms while bringing out the key (convexity) properties used by such algorithms. Examples of Bregman divergence include the squared loss and the Kullback-Liebler divergence also known as relative entropy.

When $\phi(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$, we recover the squared loss as the Bregman divergence as follows

$$\begin{aligned} d_\phi(\mathbf{x}, \mathbf{y}) &= \frac{1}{2}\|\mathbf{x}\|^2 - \frac{1}{2}\|\mathbf{y}\|^2 - \langle \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle \\ &= \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2. \end{aligned} \tag{2.7}$$

When $\phi(\mathbf{x}) = \sum_{i=1}^n \mathbf{x}(i) \log \mathbf{x}(i)$, we recover relative entropy (un-normalized) as the Bregman divergence as follows:

$$d_\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \left\{ \mathbf{x}(i) \log \left(\frac{\mathbf{x}(i)}{\mathbf{y}(i)} \right) - \mathbf{x}(i) + \mathbf{y}(i) \right\}. \tag{2.8}$$

2.1.2 The OCO model

In an OCO setting, optimization proceeds in rounds, where in round t, the algorithm has to first pick $\mathbf{x}_t \in \mathcal{X}$; then, nature reveals f_t and the value of the objective function $f_t(x_t)$ is determined. $\mathcal{X}$ is non-empty, bounded and closed [127]. The strictly convex function f_t chosen by nature can be arbitrary, even adversarial, as long as it satisfies some minimal regularity conditions, which we discuss shortly. Here we discuss in particular the full information model, where all the information about the function f_t is available to us. In a *bandit* model (not discussed in this thesis), only the loss $f_t(\mathbf{x}_t)$ is available to us. Ideally, over T rounds we would like to minimize the quantity,

$$\min_{\mathbf{x}_t \in \mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}_t) \tag{2.9}$$

Absolute minimization of (2.9) is not reasonable because we do not know the sequence of f_ts *apriori*. If the f_ts are known, (2.9) reduces to a batch optimization problem. Alternatively, the goal is to design an algorithm which picks the sequence of $\mathbf{x}_t$ such that the cumulative objective function value of the adaptive algorithm is competitive with that of the single best $x \in F$ chosen in hindsight [32, 127]. More precisely, we want a sequence $\mathbf{x}_t$ such that

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) \leq \min_{x \in \mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}) + o(T) \ . \tag{2.10}$$

With the regret defined as $R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{x \in F} \sum_{t=1}^{T} f_t(\mathbf{x})$, we simply want the regret to be sublinear [57, 72] since this implies that "on the average" the algorithm performs as well as the best fixed strategy in hindsight. Recent work [62, 64, 63] has focused on the case where the comparator class can also shift or change over time, i.e., the sequence $\{\mathbf{x}_1^*, \cdots, \mathbf{x}_T^*\}$ is the minimizer of $\sum_{t=1}^{T} f_t$ in hindsight but more on this later.

Example of an important problems which can be cast in the OCO framework s the online shortest path problem. In the online shortest path problem the decision maker is given a directed graph $G = (V, E)$ and a source-sink pair $s, t \in V$. At each iteration t a path $\mathbf{p}_t \in \mathbb{P}_{s,t}$, where $\mathbb{P}_{s,t} \subseteq \{E\}^{|V|}$ is the set of all s-t-paths in the graph. The adversary independently chooses weights on the edges of the graph, given by a

function from the edges to reveal $f_t : E \mapsto \mathbb{R}$, which can be represented as a vector in m-dimensional space (for $m = |E|$): $f_t \in \mathbb{R}^m$. The decision maker suffers an observes loss, which is the weighted length of the chosen path $\sum_{e \in p_t} f_t(e)$. This online shortest path problem can be cast in the online convex optimization framework. The objective can be posed as a linear program which keeps changing over time [1, 39]. This succinct formulation inherently leads to computationally efficient algorithms. Other applications of OCO include prediction with expert advice, online portfolio selection problem (which we discuss in details in Section 2.2), etc.

Regret bounds in OCO have been derived under very general conditions. These results can also be used in stochastic batch setting by applying online-to-batch conversions [30] and build a new batch algorithm from an existing online one. In a stochastic setting, the optimization is based on i.i.d. samples $z_1, \cdots, z_T$ drawn from an unknown distribution. The goal is to choose $\mathbf{x}$ based on the full information of the function f and the samples to minimize the expectation $\mathbb{E}_Z[f(\mathbf{x}, Z)]$ with respect to Z, a random objective. One simple way to do so is to run the online algorithm on the stochastic sequence of functions $f(\cdot, z_1), \cdots, f(\cdot, z_T)$ and set the single predictor $\bar{\mathbf{x}}$ to be the average of online choices $\mathbf{x}_1, \cdots, \mathbf{x}_T$. It is possible to show that with probability of at least $1 - \delta$ we have

$$\mathbb{E}_{\mathbf{z}}[f(\bar{\mathbf{x}}; z)] \leq \mathbb{E}_{\mathbf{z}}[f(\mathbf{x}^*; z) + O(1/\sqrt{T}). \tag{2.11}$$

Using martingale inequalities, it is possible to convert an online algorithm to a batch algorithm with a stochastic guarantee. In this thesis we will concentrate on the online case but the algorithms and analyses can be converted to the stochastic case using standard online-to-batch conversion techniques [30].

Next, we discuss algorithms for OCO which are important in the context of this thesis. We have classified the algorithms into first-order and second-order approaches. While Online Mirror Descent and Composite Objective Mirror Descent are first-order methods, Newton updates constitute second-order methods. Most of the OCO algorithms assume a black-box approach for projecting onto the feasible set, which for complex constraints (linear and or quadratic) may require iterative cyclic projections [29].

2.1.3 Online Mirror Descent (OMD)

If the dimension n is large enough, Mirror Descent [96, 12] is optimal amongst the first order methods. A Mirror Descent updates in the online setting can be written as

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\ \eta\langle f_t'(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t\rangle + d_\psi(\mathbf{x}, \mathbf{x}_t), \tag{2.12}$$

where d_ψ is a Bregman divergence and f_t' denotes an arbitrary subgradient of f_t. Given a strictly convex differentiable function ψ, we recap that the Bregman divergence $d_\psi(\mathbf{x}, \mathbf{y})$ is defined as:

$$d_\psi(\mathbf{x}, \mathbf{y}) = \psi(\mathbf{x}) - \psi(\mathbf{y}) - (\mathbf{x} - \mathbf{y})^T \nabla\psi(\mathbf{y}) \tag{2.13}$$

Mirror Descent optimizes a first-order approximation of the function f_t at the current iterate $\mathbf{x}_t$ while forcing the next iterate $\mathbf{x}_{t+1}$ to lie close to $\mathbf{x}_t$. The step size η controls the trade-off between these two. If f_ts are general convex functions, such as hinge loss, OMD attains a regret bound of $O(\sqrt{T})$. For strongly convex f_ts, such as least square regression, OMD attains a regret bound $O(\log T)$.

Choosing different Bregman divergences, lead to different algorithms. We will next discuss the gradient descent and the exponentiated gradient algorithm which are special cases of mirror descent.

2.1.3.1 Online Gradient Descent (OGD)

The *online gradient descent* [127] algorithm is a special case of mirror descent where the Bregman divergence is replaced by the squared Euclidean distance, i.e. setting $d_\psi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2$, we have

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\ \eta\langle f_t'(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t\rangle + \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2. \tag{2.14}$$

Solving $\mathbf{x}_{t+1}$ from (2.14), results in the following closed form solution for $\mathbf{x}_{t+1}$, in every iteration:

$$\mathbf{x}_{t+1} = \textstyle\prod_{\mathcal{X}} \mathbf{x}_t - \eta f_t'(\mathbf{x}_t), \tag{2.15}$$

where $\prod_{\mathcal{X}}$ denotes the projection onto nearest point in $\mathcal{X}$.

2.1.3.2 Exponentiated Gradient (EG)

Moreover, if the Bregman divergence in (2.12) is replaced by Kullback-Liebler (KL) divergence or relative entropy (2.8), we recover the exponentiated gradient (EG) algorithm [61]:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\ \eta\langle f_t'(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t\rangle + KL(\mathbf{x}, \mathbf{y}). \tag{2.16}$$

Solving $\mathbf{x}_{t+1}$ from (2.16), results in the following closed form solution for $\mathbf{x}_{t+1}$, in every iteration:

$$\mathbf{x}_t(i) = \mathbf{x}_t(i) \exp\left(-\eta \frac{\partial f_t(\mathbf{x}_t)}{\partial \mathbf{x}_t(i)}\right). \tag{2.17}$$

It is easy to handle constraints like the unit simplex constraint ($\mathbf{x}(i) \geq 0$ and $\sum_i \mathbf{x} = 1$) with EG (normalize). Note that EG leads to a multiplicative update while gradient descent discussed above, leads to an additive update.

2.1.4 Composite Objective Mirror Descent (COMID)

In regularized convex optimization problems like ridge regression and lasso, one jointly minimizes a loss function plus a regularization term as follows:

$$f_t(\mathbf{x}) = \phi_t(\mathbf{x}) + g(\mathbf{x}). \tag{2.18}$$

That is the convex loss function ϕ_t changes in each round t but the convex regularization function g does not. One can ignore the composite structure of f_t and use mirror descent algorithm discussed above but this can lead to undesirable effects. For example, when $g(\mathbf{x}) = \|\mathbf{x}\|_1$, where the ℓ_1-norm is used to introduce sparsity in the solution. However, applying mirror descent directly does not lead to sparse updates. The composite objective mirror descent (COMID) [45] proposes to handle such composite structure as follows:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\ \eta\langle f_t'(\mathbf{x}_t), \mathbf{x}\rangle + \eta g(\mathbf{x}) + d_\psi(\mathbf{x}, \mathbf{x}_t) \tag{2.19}$$

Note that an important difference between (2.19) and (2.12) is that g is not linearized. [45] shows that COMID is no more costlier than the usual mirror descent update. Like OMD, for general convex functions, COMID achieves $O(\sqrt{T})$ and when the composite functions $f_t = \phi_t + g$ are strongly convex, COMID achieves $O(\log T)$ regret.

2.1.5 Online Newton Step (ONS)

The above discussed algorithms, such as OGD and EG algorithms, are first order methods. We now discuss the Online Newton Step (ONS) which is the analogue of the Newton-Rhapson method and is based on second-order information. However, the second-order information is used only in the analysis while the algorithm only uses the gradients. ONS uses the following updates for $\mathbf{x}_{t+1}$:

$$\mathbf{x}_{t+1} = \text{argmin} \sum_{\tau=1}^{t} f_t(\mathbf{x}) + \frac{\beta}{2}\|\mathbf{x}\|^2 \tag{2.20}$$

Using a second-order expansion of the first term

$$f_t(\mathbf{x}) \geq f_t(\mathbf{x}_t) + (\mathbf{x} - \mathbf{x}_t)^T \nabla f_t(\mathbf{x}_t) + \frac{\beta}{4}[(\mathbf{x} - \mathbf{x}_t)^T \nabla f_t(\mathbf{x}_t) \nabla f_t(\mathbf{x}_t)^T (\mathbf{x} - \mathbf{x}_t)], \tag{2.21}$$

then the ONS update is as follows:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\text{argmin}} \prod_{\mathcal{X}}^{A_t} (\mathbf{x}_t - \frac{2}{\beta} A_t^{-1} \bigtriangledown f_t(\mathbf{x}_t)), \tag{2.22}$$

where $\prod_{\mathcal{X}}^{A_t}(\mathbf{y}) = \underset{\mathbf{x} \in \mathcal{X}}{\text{argmin}} (\mathbf{y} - \mathbf{x})^T A_t^{-1} (\mathbf{y} - \mathbf{x})$ is the projection operator and $A_t = \sum_{\tau=1}^{t} \bigtriangledown f_t(\mathbf{x}_t) \bigtriangledown f_t(\mathbf{x}_t)^T + \epsilon \mathbf{I}$.

We see from (2.22), that the point chosen for a current iteration is the point chosen in the previous iteration added to an additional vector. Since just adding a multiple of the Newton vector to the current solution, may result in an update outside the convex set, we project back into the set to obtain $\mathbf{x}_t$ and it is the projection according to the norm defined by the mateix A_t, rather than the Euclidean norm. By using the second order information, ONS achieves $O(\log T)$ regret.

2.2 Online Portfolio Selection

We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over T periods. For ease of exposition, we will consider a period to be a day, but the analysis presented in the chapter holds for any valid definition of a 'period,' such as an hour or a month. Let $x_t(i)$ denote the *price relative* of stock s_i in day t, i.e., the multiplicative factor by which the price of s_i changes in day t. Hence, $x_t(i) > 1$ implies a gain, $x_t(i) < 1$ implies a loss, and $x_t(i) = 1$ implies the price remained unchanged. Further, $x_t(i) > 0$ for all i, t.

Let $\mathbf{x}_t = \langle x_t(1), \ldots, x_t(n) \rangle$ denote the vector of price relatives for day t, and let $\mathbf{x}_{1:t}$ denote the collection of such price relative vectors upto and including day t. A portfolio $\mathbf{p}_t = \langle p_t(1), \ldots, p_t(n) \rangle$ on day t can be viewed as a probability distribution over the stocks that prescribes investing $p_t(i)$ fraction of the current wealth in stock $x_t(i)$. Note that the portfolio $\mathbf{p}_t$ has to be decided before knowing $\mathbf{x}_t$ which will be revealed only at the end of the day. The multiplicative gain in wealth at the end of day t, is then simply $\mathbf{p}_t^T \mathbf{x}_t = \sum_{i=1}^{n} p_t(i) x_t(i)$. For a sequence of price relatives $\mathbf{x}_{1:t-1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}$ upto day $(t-1)$, the sequential portfolio selection problem in day t is to determine a portfolio $\mathbf{p}_t$ based on past performance of the stocks. At the end of day t, $\mathbf{x}_t$ is revealed and the actual performance of $\mathbf{p}_t$ gets determined by $\mathbf{p}_t^T \mathbf{x}_t$. Over a period of T days, for a sequence of portfolios $\mathbf{p}_{1:T} = \{\mathbf{p}_1, \ldots, \mathbf{p}_T\}$, the multiplicative gain in wealth is then

$$S(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \prod_{t=1}^{T} \left(\mathbf{p}_t^T \mathbf{x}_t \right) \ . \tag{2.23}$$

In the literature, one often looks at the logarithm of the multiplicative gain, given by

$$LS(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \sum_{t=1}^{T} \log \left(\mathbf{p}_t^T \mathbf{x}_t \right) \ . \tag{2.24}$$

Ideally, we would like to maximize $S(\mathbf{p}_{1:T}, \mathbf{x}_{1:T})$ over $\mathbf{x}_{1:T}$. Unfortunately, portfolio selection cannot be posed as an optimization problem due to the temporal nature of the choices: $\mathbf{x}_t$ is not available when one has to decide on $\mathbf{p}_t$. Further, in a stock market, (statistical) assumptions regarding $\mathbf{x}_t$ can be difficult to make. Existing literature for theoretically motivated portfolio selection [36, 61, 32] focus on designing algorithms whose overall gain in wealth is guaranteed to be competitive with reasonable strategies in hindsight.

Next, we discuss a couple of reasonable strategies which the theoretically motivated algorithms can use as baselines to compare their performance. We then go on to discuss the theoretically motivated portfolio selection algorithms, followed by a discussion of the heuristic, Anticor. We conclude this section with details on a family of heuristics based on the mean-reversion strategy in finance.

2.2.1 Buy-and-Hold Strategies

A reasonable strategy to outperform could be the best Buy-and-Hold (BH) strategy, where an investor distributes his wealth simply amongst the n stocks according to some fixed distribution $\mathbf{p}_{bh}$ at the very beginning of the trading period and does not trade anymore. With progress of time, the fraction of total wealth in each stock may change automatically depending on the stocks performance. Note that the best buy-and-hold strategy would be to pick the single best stock in hindsight and fully invest all wealth in that stock. Clearly, no forward moving algorithm would be able to pick the best stock in hindsight. However, buy-and-hold strategies form an important class of strategies to compete against.

2.2.2 Constant Rebalanced Portfolios

The best Constant Rebalanced Portfolio (CRP), could be another reasonable strategy to compete against. The CRP investment strategy maintains a fixed fraction of total wealth in each of the stocks. So, a CRP has a fixed portfolio vector $\mathbf{p}_{crp} = \langle p(1), \ldots, p(n) \rangle$ which is employed every day. Such a strategy requires vast amounts of trading every day to ensure that the investment proportions are rebalanced back to the vector $\mathbf{p}_{crp}$. To understand the strength of this family of strategies, consider the example of a market consisting of two stocks. The first stock is a no-growth stock and has price relatives $1, 1, 1, 1 \ldots$ over time. The second stock doubles in value on even days and on odd days its value gets halved. So, its price relatives are $\frac{1}{2}, 2, \frac{1}{2}, 2, \ldots$. The sequence of market vectors in this case is $(1, \frac{1}{2}), (1, 2), (1, \frac{1}{2}), (1, 2), (1, \frac{1}{2}), \ldots$. A Buy-and-Hold strategy will not make any gains in this market. On the other hand, if we use a CRP of $\mathbf{p}_{crp} = \left(\frac{1}{2}, \frac{1}{2}\right)$, then the growth of wealth every 2 days will be $\frac{9}{8}$ so that after T days, the multiplicative gain in wealth will be $\left(\frac{9}{8}\right)^{T/2}$. It is easy to see that the wealth accumulated by the best CRP will be at least as big as that by the best Buy-and-Hold strategy and hence by the single best stock. CRP is also known to have certain optimality properties when certain statistical assumptions regarding the price relatives can be made [38]. A special case of CRP is the Uniform Constant Rebalanced Portfolio (UCRP), that rebalances to the uniform portfolio $p = (\frac{1}{n}, \cdots, \frac{1}{n})$ at the beginning of each investment period/day.

2.2.3 Universal Algorithms

Theoretically motivated strategies are guaranteed to be competitive with the best CRP by achieving small regret. For any sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$ of price relatives, let $\mathbf{p}_1, \ldots, \mathbf{p}_T$ be the sequence of portfolios selected by the algorithm. The regret of a portfolio selection algorithm **ALG** is given by:

$$Regret(\mathbf{ALG}) \triangleq \max_{\mathbf{p}} \sum_{t=1}^{T} \log\left(\mathbf{p}^T \mathbf{x}_t\right) - \sum_{t=1}^{T} \log\left(\mathbf{p}_t^T \mathbf{x}_t\right) \ . \tag{2.25}$$

An investment strategy is deemed ***universal*** if it has sublinear regret, i.e., $Regret(\mathbf{ALG}) = o(T)$. We now briefly review a set of important sequential portfolio selection strategies from the literature which are universal.

2.2.3.1 Universal Portfolios (UP)

The seminal work of Cover [36] introduced Universal Portfolios (UP), the first algorithm which can be shown to be competitive with the best CRP. The algorithm has since been extended to various practical scenarios, such as investment with side information [37]. The key idea behind UP is to maintain a distribution over all CRPs and perform a Bayesian update after observing every x_t. Since each CRP $\mathbf{q}$ is a distribution over n stocks and hence lies in the n-simplex, one uses a distribution $\mu(\mathbf{q})$ over the n-simplex. A popular choice is the Dirichlet prior $\mu(\mathbf{q}) = \mathrm{Dir}(\frac{1}{2}, \ldots, \frac{1}{2})$ over the simplex. For any CRP $\mathbf{q}$, let $S_{t-1}(\mathbf{q}, \mathbf{x}_{1:t-1}) = \prod_{t'=1}^{t-1} \log\left(\mathbf{q}^T \mathbf{x}_{t'}\right)$ denote the wealth accumulated by $\mathbf{q}$ over $(t-1)$ days. Then, the universal portfolio $\mathbf{p}_t$ is defined as the weighted average over all such q:

$$p_t(i) = \frac{\int_{\mathbf{q}} q(i) S_{t-1}(\mathbf{q}, \mathbf{x}_{1:t-1}) \mu(\mathbf{q}) d\mathbf{q}}{\int_{\mathbf{q}} S_{t-1}(\mathbf{q}, \mathbf{x}_{1:t-1}) \mu(\mathbf{q}) d\mathbf{q}} \ . \tag{2.26}$$

UP has a regret of $O(\log T)$ with respect to the best CRP in hindsight. However, the updates for UP are computationally prohibitive. Discrete approximation or recursive series expansion are used to evaluate the above integrals. However, in either case, the time and space complexity for finding the new universal portfolio vector grows exponentially in the dimensionality of the simplex, i.e., number of stocks.

2.2.3.2 Exponentiated Gradient (EG) portfolios

The key motivation behind the Exponentiated Gradient (EG) strategy [61] was to design a computationally efficient portfolio selection algorithm which stays competitive with the best CRP. The EG algorithm scales linearly with the number of stocks and the portfolios are guaranteed to stay competitive with the best CRP. However, the regret is $O(\sqrt{T})$ under the *no-junk-bond* assumption and is weaker than that of UP. The *no-junk-bond assumption* states that all the $x_t(i)$ are bounded from below such that $x_t(i) > \alpha > 0$ for all t.[1] The EG investment strategy was introduced and analyzed by [61]. Their framework for updating a portfolio vector is analogous to the framework developed by [76] for online regression. In the online learning framework, the portfolio vector itself encapsulates the necessary information from all previous price relatives. At the start of day t, the algorithm computes its new portfolio vector $\mathbf{p}_t$ such that it stays close to $\mathbf{p}_{t-1}$ and does well on the price relatives x_{t-1} for the previous day. In particular, the new portfolio vector $\mathbf{p}_t$ is chosen so as to maximize

$$F(\mathbf{p}_t) = \eta \log(\mathbf{p}_t^T \mathbf{x}_t) - KL(\mathbf{p}_t, \mathbf{p}_{t-1}) \ , \tag{2.27}$$

where $\eta > 0$ is a parameter called the learning rate and $KL(\cdot, \cdot)$ is the KL-divergence ensuring $\mathbf{p}_t$ stays close to $\mathbf{p}_{t-1}$. Using an approximation of F based on Taylor expansion, the updated portfolio turns out to be

$$p_t(i) = \frac{p_{t-1}(i) \exp\left(\eta \frac{x_{t-1}(i)}{\mathbf{p}_{t-1}^T \mathbf{x}_{t-1}}\right)}{\sum_{i'=1}^{n} p_{t-1}(i') \exp\left(\eta \frac{x_{t-1}(i')}{\mathbf{p}_{t-1}^T \mathbf{x}_{t-1}}\right)} \ . \tag{2.28}$$

Note that $\mathbf{p}_{t-1}^T \mathbf{x}_{t-1}$ is the average price relative, and the wealth allocated to stock i relies on the ratio $\frac{x_{t-1}(i)}{\mathbf{p}_{t-1}^T \mathbf{x}_{t-1}}$. In particular, if the price relative for a stock is greater than the average in a particular round, the investment on that stock is increased accordingly.

2.2.3.3 Online Newton Step Method (ONS)

The Online Newton Step (ONS) method for portfolio selection is an application of the Newton step to the online setting. Recent work has shown that the ONS approach can

[1] The no-junk-bond assumption can be removed with a more advanced analysis yielding a regret of $O(T^{3/4})$ [61].

be used in online convex optimization to achieve sharper regret bounds compared to online gradient descent based methods [59, 3, 58]. Let $\mathbf{p}_t$ is a portfolio vector, such that $\mathbf{p}_t \in \Delta_n$, the n-simplex. The ONS algorithm uses following portfolio update method for round $t > 1$:

$$\mathbf{p}_t = \prod_{\Delta_n}^{\mathbf{A_t}-\mathbf{1}} \left(\mathbf{p_{t-1}} - \frac{1}{\beta} \mathbf{A_{t-1}^{-1}} \nabla_{t-1} \right) \tag{2.29}$$

where $\nabla_t = \nabla[\log(\mathbf{p}_t \cdot \mathbf{x}_t)] = \frac{1}{\mathbf{p}_t \cdot \mathbf{x}_t} \mathbf{x}_t$, $\mathbf{A}_t = \sum_{\tau=1}^{t} \nabla_\tau \nabla_\tau + \mathbb{I}$, β is a non-negative constant, and $\prod_{\triangle_\mathbf{n}}^{\mathbf{A_t}-\mathbf{1}}$ is the projection onto the $n$simplex Δ_n according to the norm induced by $\mathbf{A_{t-1}}$, i.e.,

$$\prod_{\Delta_n}^{\mathbf{A_t}-\mathbf{1}}(\mathbf{y}) = \underset{\mathbf{x} \in \Delta_n}{\operatorname{argmin}} \left(\mathbf{y} - \mathbf{x}\right)^T \mathbf{A}_{t-1} \left(\mathbf{y} - \mathbf{x}\right) \ . \tag{2.30}$$

Under the no-junk-bond assumption, ONS achieves a $O(\log T)$ regret.[2] ONS is computationally more efficient than UP. ONS has a better regret bound than EG, but is however less efficient than EG.

2.2.4 Anticor

Anticor (AC) is a heuristic which does not confirm to the universal property for portfolio selection algorithms [18]. In AC, learning the best stocks (to invest money in) is done by exploiting the volatility of the market and the statistical relationship between the stocks. It implements the 'reversal to the mean' market phenomenon rather aggressively. An important parameter for Anticor is the window length w. The version of Anticor implemented works with two most recent windows of length w. The strategy is to move money from a stock i to stock j if the growth rate of stock i is greater than the growth rate of j in the most recent window. An additional condition that needs to be satisfied is the existence of a positive correlation between stock i in the second last window and stock j in the last window. The satisfiability of this condition is an indicator that stock j will replicate stock i's past behavior in the near future. The amount of money that is transferred from stock i to stock j depends on the strength of correlation between the stocks and the strength of "self-anti-correlations" between each stock i over two consecutive windows.

[2] Without the no junk bond assumption, the regret of ONS is $O(\sqrt{T})$.

For a window length of w, LX_1 and LX_2 are defined as follows:

$$LX_1 = [\log(x_{t-2w+1}), \cdots, \log(x_{t-w})]^T, \tag{2.31}$$

and

$$LX_2 = [\log(x_{t-w+1}), \cdots, \log(x_t)]^T. \tag{2.32}$$

Thus, LX_1 and LX_2 are two $w \times n$ matrices over two consecutive time windows. The j^{th} column of LX_k is denoted by $LX_k(j)$, which tracks the performance of stock j in window k. Let $\mu_k(j)$ be the mean of $LX_k(j)$ and $\sigma(k)$ be the corresponding standard deviation. The cross-covariance matrix between the column vectors of LX_1 and LX_2 is defined as follows:

$$M_{cov}(i,j) = \frac{1}{w-1}(LX_1(i) - \mu_1(i))^T(LX_2(j) - \mu_2(j)) \ . \tag{2.33}$$

The corresponding cross-correlation matrix is given by:

$$M_{cor}(i,j) = \begin{cases} \frac{M_{cov}(i,j)}{\sigma_1(i)\sigma_2(j)} & \sigma_1(i), \sigma_2(j) \neq 0 \\ 0 & \text{otherwise} \ . \end{cases} \tag{2.34}$$

Following the reversal to mean strategy, the proportion of wealth to be moved from stock i to stock j is defined as:

$$C_{i\to j} = M_{cor}(i,j) + \lfloor -M_{cor}(i,i) \rfloor + \lfloor -M_{cor}(j,j) \rfloor \ , \tag{2.35}$$

where $\lfloor x \rfloor = \max(0, x)$. The normalized transfer is defined as

$$T_{i\to j} = p_t(i)\frac{C_{i\to j}}{\sum_{j'} C_{i\to j'}} \ . \tag{2.36}$$

Using these transfer values, the portfolio is defined to be

$$p_{t+1}(i) = p_t(i) + \sum_{j\neq i}(T_{j\to i} - T_{i\to j}) \ . \tag{2.37}$$

For more details on the Anticor algorithm please refer to [18]. Experiments with different variations of Anticor in [18] brought to the fore the exceptional empirical performance improvement that a suitable heuristic can achieve over theoretically well grounded approaches.

2.2.5 Reversion

Another class of online portfolio selection strategy which has recently attracted some interest is based on *mean reversion*. These strategies [82, 83, 81] often achieve good empirical performance on many real datasets when they make single-period or multi-period mean reversion assumption. Loosely, the mean-reversion strategies transfer wealth from the well performing stocks of today to the poorly performing stocks. The underlying assumption being that the stocks that perform well today will have poor performance the next trading day compared to the other stocks. Hence, in a recent survey [85] these heuristics have been referred to as *Follow-the-Loser* approach which tries to increase the relative weights of less successful stocks, or transfer money from the winners to the losers. However, these strategies do not come with any performance guaranties (unlike the universal algorithms) and thus run the risk of poor performance when the mean reversion assumption does not hold. Next, we briefly discuss the recent mean reversion strategies starting with PAMR [83] and CWMR [82] which assumes a single period mean reversion followed by OLMAR [81] which exploits a multiple-period mean reversion.

2.2.5.1 Passive Aggressive Mean Reversion(PAMR)

The main ides of PAMR is to first design a loss function which exploits the mean reversion property with the Passive Aggressive online learning [83]. If the expected return based on last price relative is larger than a threshold, the loss will linearly increase; otherwise, the loss is zero. In particular, the authors defined the ϵ-insensitive loss function for the t-th period as,

$$\ell_\epsilon(\mathbf{p}; \mathbf{x}_t) = \begin{cases} 0 & \mathbf{p} \cdot \mathbf{x}_t \leq \epsilon \\ \mathbf{p} \cdot \mathbf{x}_t - \epsilon & \text{otherwise,} \end{cases} \tag{2.38}$$

where $0 \leq \epsilon \leq 1$ is a sensitivity parameter to control the reversion threshold. Based on the loss function, PAMR passively maintains last portfolio if the loss is zero , otherwise aggressively approaches a new portfolio that can force the loss to zero. In summary , PAMR obtains the next portfolio via the following optimization problem,

$$\mathbf{p}_{t+1} = \underset{\mathbf{p} \in \triangle_n}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{p} - \mathbf{p}_t\|^2 \text{ s.t. } \ell_\epsilon(\mathbf{p} \cdot \mathbf{x}_t) = 0. \tag{2.39}$$

Solving the above optimization problem, PAMR has a closed form update for the unconstrained problem as follows,

$$\mathbf{p}_{t+1} = \mathbf{p}_t = \tau_t(\mathbf{x}_t - \bar{\mathbf{x}}_t\mathbf{1}), \quad \tau_t = \max\left\{0, \frac{\mathbf{p}_t \cdot \mathbf{x}_t - \epsilon}{\|\mathbf{x}_t - \bar{x}_t\mathbf{1}\|^2}\right\} \tag{2.40}$$

where $\bar{x}_t = \frac{\mathbf{x}_t}{n}$. This is followed by a projection to the simplex carried out as in [46]. Like Anticor, it is difficult to establish a meaningful regret bound for PAMR. However, empirically it outperforms other existing ideas on datasets for which the single period mean reversion idea holds. Its performance suffers serious degradation when the assumption is violated.

2.2.5.2 Confidence Weighted Mean Reversion

In Confidence Weighted Mean Reversion (CWMR) [82], the portfolio vector is modeled by a random variable which follows a multi-variate Gaussian distribution with mean $\mu \in \mathbf{R}^n$ and the diagonal covariance matrix $\Sigma \in \mathbf{R}^{n\times n}$. The authors propose that the value μ_i represents the knowledge of asset i in the portfolio, and the diagonal element $\Sigma_{i,i}$ stands for the confidence in μ_i. At the beginning of day $t+1$, a new portfolio is chosen by sampling from the Gaussian distribution as $\mathbf{p}_{t+1} \sim \mathcal{N}(\mu, \Sigma)$. Moreover, the return from the portfolio $\mathbf{p}_{t+1}$ is also modeled as a univariate Gaussian as follows $D \sim \mathcal{N}(\mu \cdot \mathbf{x}_t, \mathbf{x}_t^T\Sigma\mathbf{x}_t)$. The parameters of the multivariate Gaussian is then updated so that the distribution is close to the last in terms of Kullback-Liebler divergence if the probability of a portfolio return lower than ϵ is higher than a specified threshold. In summary, the optimization problem to be solved is

$$(\mu_{t+1}, \Sigma_{t+1}) = \underset{\mu\in\triangle_n,\Sigma}{\operatorname{argmin}} \mathrm{D}_{\mathrm{KL}}(\mathcal{N}(\mu,\Sigma))||\mathcal{N}(\mu_t,\Sigma_t)). \tag{2.41}$$

[84] proposed the following transformation for (2.41):

$$\begin{aligned}(\mu_{t+1}, \Sigma_{t+1}) = \operatorname{argmin}\ & \frac{1}{2}\left(\log\frac{|\Sigma_t|}{|\Sigma|}\right) + \mathrm{Tr}(\Sigma_t^{-1}\Sigma) + (\mu_t - \mu)^T\Sigma_t^{-1}(\mu_t - \mu) \\ \text{s.t.}\ \ & \epsilon - \mu^T\mathbf{x}_t \geq \mathbf{x}_t \\ & \mu^T\mathbf{1} = 1, \boldsymbol{\mu} \geq 0,\end{aligned} \tag{2.42}$$

where $\phi = \Phi^{-1}(\theta)$ and Φ is the cumulative distribution function of the Gaussian distribution. Solving the above optimization, the closed form update scheme proposed by [84]

is as follows:

$$\mu_{t+1} = \mu_t - \lambda_{t+1}\Sigma_t(\mathbf{x}_t - \bar{x}_t\mathbf{1})\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\lambda_{t+1}\phi\mathbf{x}_t\mathbf{x}_t^T. \tag{2.43}$$

λ_{t+1} corresponds to the Lagrangian multiplier as specified in [84] and $\bar{\mathbf{x}}_t = \frac{\mathbf{1}\Sigma_t\mathbf{x}_t}{\mathbf{1}\Sigma_t\mathbf{1}}$ denotes the confidence weighted price relative average. Similar to PAMR, CWMR adopts a mean reversion strategy with single period mean reversion assumption and demonstrates strong empirical performance. However, CWMR is a heuristic like its precursor and does not have any theoretical guarantee in the form of meaningful regret bounds.

2.2.5.3 Online Moving Average Mean Reversion (OLMAR)

Online Moving Average Mean Reversion (OLMAR) [81] defined multiple-period mean reversion named *moving average reversion*, and proposed OLMAR to exploit the multiple-period mean reversion. In particular, the authors predict the next price vector as the moving average within a window, which is calculated as $\frac{1}{w}\sum_{i=t-w+1}^{t}\mathbf{y}_i$. The price relative for the next day $t+1$ is calculates as,

$$\hat{\mathbf{x}}_{t+1} = \frac{1}{w}\left(1 + \frac{1}{\mathbf{x}_t} + \cdots + \frac{1}{\bigodot_{l=0}^{w-2}\mathbf{x}_{t-l}}\right), \tag{2.44}$$

where w is the window size and $\bigodot$ denotes the element-wise product. A strategy similar to PAMR [83] is used to find a portfolio:

$$\mathbf{p}_{t+1} = \underset{\mathbf{p}\in\triangle_n}{\operatorname{argmin}}\frac{1}{2}\|\mathbf{p} - \mathbf{p}_t\|^2 \text{ s.t. } \mathbf{p}\cdot\hat{\mathbf{x}}_{t+1} \geq \epsilon. \tag{2.45}$$

OLMAR achieves the best results among the existing mean reversion algorithms, and performs well on datasets on which PAMR and CWMR failed.

2.2.6 Assumptions

In general the following assumptions are made while evaluating the performance for the above discussed online portfolio selection algorithms:

1. Transaction cost: No transaction cost/tax exists.

2. Risk: Investors are not risk averse.

3. Market liquidity: One can buy and sell the desired quantities of the stocks at last closing prices.

4. Impact Cost: The market behavior is not affected by the portfolio selection strategies.

As a part of this thesis, we will address 1 and 2 in Chapter 3 and Chapter 5 respectively.

2.3 Alternating Direction Method of Multipliers (ADMM)

In this section we discuss the Alternating Direction Method of Multipliers (ADMM) [22], a tool that has recently become very popular due to its ease of applicability and empirical performance for a wide variety of problems including composite objectives (a detailed survey can be found in [22]). ADMM method is closely related to Bregman iterative algorithms for ℓ_1 problems and proximal point methods. ADMM solves problems of the form of

$$\begin{aligned} &\min_{\mathbf{x}\in\mathcal{X},\mathbf{z}\in\mathcal{Z}} f(\mathbf{x}) + g(\mathbf{z}) \\ &\text{subject to } \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = c, \end{aligned} \tag{2.46}$$

with variables $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^m$, where $A \in \mathbb{R}^{p\times n}$, $\mathbf{B} \in \mathbb{R}^{p\times m}$ and $c \in \mathbb{R}^p$. $\mathcal{X}$ and $\mathcal{Z}$ are convex sets which are dropped in the sequel for ease of exposition. An advantage of ADMM is that it can handle linear equalities as given in (2.46), which makes distributed optimization by variable splitting straightforward [22]. f and g are convex functions. The original variable has been split into two parts $\mathbf{x}$ and $\mathbf{z}$, such that the objective function is also separable across that splitting into $f(\mathbf{x})$ and $g(\mathbf{z})$. The augmented Lagrangian for (2.46) can be written as

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c\|^2, \tag{2.47}$$

where $\rho > 0$ is the augmented lagrangian parameter. Now, ADMM consists of the following iterations:

$$\mathbf{x}^{(k+1)} := \underset{\mathbf{x}}{\operatorname{argmin}}\ L_\rho(\mathbf{x}, \mathbf{z}^{(k)}, \mathbf{y}^{(k)}) \tag{2.48}$$

$$\mathbf{z}^{(k+1)} := \underset{\mathbf{x}}{\operatorname{argmin}}\ L_\rho(\mathbf{x}^{(k+1)}, \mathbf{z}, \mathbf{y}^{(k)}) \tag{2.49}$$

$$\mathbf{y}^{(k+1)} := \mathbf{y}^{(k)} + \rho(\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - c) \tag{2.50}$$

Hence, ADMM for (2.46) consists of a $\mathbf{x}$-minimization step, $\mathbf{z}$-minimization step and a closed form update of the dual variable $\mathbf{y}$ with a step size of ρ. The updates are done in an *alternating* fashion as follows:

$$\mathbf{x}^{(k+1)} := \underset{\mathbf{x}}{\operatorname{argmin}}\ f(\mathbf{x}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}^{(k)} - c\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}^{(k)} - c\|^2 \tag{2.51}$$

$$\mathbf{z}^{(k+1)} := \underset{\mathbf{z}}{\operatorname{argmin}}\ g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z} - c\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z} - c\|^2 \tag{2.52}$$

$$\mathbf{y}^{(k+1)} := \mathbf{y}^{(k)} + \rho(\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - c) \tag{2.53}$$

$$\tag{2.54}$$

Setting $\mathbf{u} = (1/\rho)\mathbf{y}$, which is the *scaled dual variable*, we can rewrite the ADMM updates for this *scaled form* as follows:

$$\mathbf{x}^{(k+1)} := \underset{\mathbf{x}}{\operatorname{argmin}}\ \left(f(\mathbf{x}) + (\rho/2)\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}^k - c + \mathbf{u}^{(k)}\|^2\right) \tag{2.55}$$

$$\mathbf{z}^{(k+1)} := \underset{\mathbf{x}}{\operatorname{argmin}}\ \left(g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z} - c + \mathbf{u}^{(k)}\|^2\right) \tag{2.56}$$

$$\mathbf{u}^{(k+1)} := \mathbf{u}^{(k)} + \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - c. \tag{2.57}$$

The convergence results of ADMM has been discussed in [22, 119] and a $O(1/K)$ confergence rate of ADMM has been established in [60, 119], where K is the total number of ADMM iterations. For strongly convex functions, an accelerated version of ADMM can converge at a rate of $O(1/K^2)$ [56].

Inexact Minimization: In situations where the closed form updates for $\mathbf{x}$ and $\mathbf{z}$ are not straightforward and are not carried out exactly or are carried out using an iterative update method, ADMM will still converge [22]. For iterative updates, $\mathbf{x}$ and $\mathbf{z}$ are solved more accurately with the progression of the iterations.

2.3.1 Bregman ADMM

Bregman ADMM (BADMM) [120] provides a unified framework for ADMM and its variants, including generalized ADMM and inexact ADMM. BADMM generalizes ADMM and shows that the quadratic penalty terms in the $\mathbf{x}$ and $\mathbf{z}$ updates in (2.52) and (2.53) can be replaced by a Bregman divergence. Since function f and g and constraints $\mathcal{X}$ and $\mathcal{Z}$ usually have different structures, it might not be possible to have efficient algorithms by simply using the same Bregman divergence in the $\mathbf{x}$ and $\mathbf{z}$ updates. To allow the use of different Bregman divergence in the $\mathbf{x}$ and $\mathbf{z}$ updates, additional Bregman divergences are used in the updates respectively. In particular, the quadratic penalty in (2.47) is replaced by a Bregman divergence as follows:

$$L_\rho^\phi(\mathbf{x},\mathbf{z},\mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c\rangle + \rho d_\phi(c - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}) \tag{2.58}$$

where the assumption is that $d_\phi(c - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z})$ is well defined.

$$\mathbf{x}^{(k+1)} := \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\ f(\mathbf{x}) + \langle \mathbf{y}^{(k)}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}^{(k)} - c\rangle + \rho d_\phi(c - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}^{(k)}) + \rho_{\mathbf{x}} d_{\phi_{\mathbf{x}}}(\mathbf{x}, \mathbf{x}^{(k)}) \tag{2.59}$$

$$\mathbf{z}^{(k+1)} := \underset{\mathbf{z}\in\mathcal{Z}}{\operatorname{argmin}}\ + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z} - c\rangle + \rho d_\phi(\mathbf{B}\mathbf{z}, c - \mathbf{A}\mathbf{x}^{(k+1)}) + \rho_{\mathbf{x}} d_{\phi_{\mathbf{z}}}(\mathbf{z}, \mathbf{z}^{(k)}) \tag{2.60}$$

$$\mathbf{y}^{(k+1)} := \mathbf{y}^{(k)} + \tau(\mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{B}\mathbf{z}^{(k+1)} - c) \tag{2.61}$$

Three Bregman divergences are used in BADMM and $\rho > 0$, $\tau > 0$, $\rho_{\mathbf{x}} \geq 0$, $\rho_{\mathbf{z}} \geq 0$. Two scenarios are discussed in [120]: if $\rho_{\mathbf{x}}$ and $\rho_{\mathbf{z}}$ are zero, BADMM replaces the quadratic penalty of ADMM by a single Bregman divergence. In this scenario, the $\mathbf{x}$ and $\mathbf{z}$ updates should be solved exactly. If one or both of $\rho_{\mathbf{x}}$ and $\rho_{\mathbf{z}}$ are positive, different Bregman divergences can be used in the $\mathbf{x}$ and $\mathbf{z}$ updates so that they can be solved inexactly. The second case usually takes more iterations to converge but might be less expensive. BADMM has been shown to have a convergence rate of $O(1/\sqrt{K})$ [120].

2.3.2 Online ADMM

ADMM makes distributed optimization easier by variable splitting. However, in an online or stochastic gradient setting one obtains a double-loop algorithm, which has

to be run till convergence after every new data point is revealed. Here we discuss the Online ADM (OADM) [119], which generalizes ADMM to the online setting. OADM considers optimization problems of the following form:

$$\min_{\mathbf{x}\in\mathcal{X},\mathbf{z}\in\mathcal{Z}} \sum_{t=1}^{T} (f_t(\mathbf{x}) + g(\mathbf{z})) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = c. \tag{2.62}$$

At round t, OADM considers solving the following optimization problem

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=c} f_t(\mathbf{x}) + g(\mathbf{z}) + \eta d_\phi(\mathbf{x}, \mathbf{x}_t), \tag{2.63}$$

where $\eta \geq 0$ is the learning rate and Bregman divergence $d_\phi(\mathbf{x}, \mathbf{x}_t) \geq \frac{\alpha}{2}\|\mathbf{x} - \mathbf{x}_t\|^2$. (2.63) can be solved by ADMM but it requires a double loop, where in the outer loop the function f_t changes, and the inner loop runs ADMM iteratively till convergence. Existing online methods, like projected gradient descent and variants [57, 45] assume a black-box for projecting into the feasible set, which for linear constraints may require iterative cyclic projections [29]. Instead of requiring the equality constraints to be satisfied in each round t, OADM requires the equality constraints to be satisfied in the long run, with a notion of regret associated with the constraints. OADM focuses on the following problem:

$$\begin{aligned} &\min_{\mathbf{x}_t,\mathbf{z}_t} \sum_{t=1}^{T} f_t(\mathbf{x}_t) + g(\mathbf{z}_t) - \min_{\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=c} \sum_{t=1}^{T} f_t(\mathbf{x}) + g(\mathbf{z}) \\ &\text{s.t.} \quad \sum_{t=1}^{T} \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c\|^2 = o(T), \end{aligned} \tag{2.64}$$

so that the cumulative constraint violation is sub linear in T. The augmented lagrangian function of (2.63) at time t is

$$L_t(\mathbf{x}, \mathbf{y}, \mathbf{z}) = f_t(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c\rangle + \eta d_\phi(\mathbf{x}, \mathbf{x}_t) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - c\|^2. \tag{2.65}$$

OADM makes just one pass through the following steps:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x}} \quad f_t(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - c\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - c\|^2 + \eta d_\phi(\mathbf{x}, \mathbf{x}_t) \tag{2.66}$$

$$\mathbf{z}_{t+1} = \operatorname*{argmin}_{\mathbf{z}} \quad g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - c\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - c\|^2 \tag{2.67}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - c). \tag{2.68}$$

(2.67) has two penalty terms: (1) a quadratic term and (2) a Bregman divergence. If the Bregman divergence is not a quadratic function, a way to handle this case is to linearize the objective, such that:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \langle f_t'(\mathbf{x}_t) + \mathbf{A}^T\{\mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_t) + \mathbf{B}\mathbf{z}_t - c)\}, \mathbf{x} - \mathbf{x}_t\rangle + \eta d_\phi(\mathbf{x}, \mathbf{x}_t). \quad (2.69)$$

This is an example of the inexact ADMM we discussed above. OADM looks at two types of regret analysis. The first type is the regret of the objective based on variable splitting, i.e.,

$$R_1(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) + g(\mathbf{z}_t) - \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = c \sum_{t=1}^{T} f_t(\mathbf{x}) + g(\mathbf{z}). \quad (2.70)$$

(2.70) is the standard regret in online learning. The second is the regret of the constraint violation defined as:

$$R^c(T) = \sum_{t=1}^{T} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - x\|^2 + \|\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t\|^2, \quad (2.71)$$

which is based on both primal and dual residuals. [119] shows that the regret bounds for the objective and constraint violation is $O(\sqrt{T})$ for general convex functions f_t and g. When both f_t and g are strongly convex functions, the standard regret for the objective and constraint violation is $O(\log T)$.

Chapter 3

Online Lazy Updates

3.1 Motivation

With the ever increasing amount of data, particularly from search engines and social networks, stochastic optimization algorithms have become desirable for large-scale machine learning tasks because of their empirical efficiency and strong theoretical guarantees [19, 20, 12, 106, 4].

However, a major challenge that is encountered is the cost of updating model parameters especially when the number of parameters can be in the order of billions. Often times when parameters are updated, their values do not change significantly. As such, the cost of updating each parameter starts to outweigh the benefit.

An important and relevant application where changing the model parameters might prove to be monetarily expensive is the domain of online portfolio selection. Here every time an investor changes his portfolio, he ends up buying or selling his stocks and incurring transaction costs. Hence, trading aggressively might sometimes hurt an investor instead of proving to be beneficial. In such a situation it might be helpful to make sparse or *lazy* updates to a portfolio.

Algorithms for automatically designing portfolios based on historical stock market data have been extensively investigated in the literature for the past five decades [91, 75, 108]. With the realization that any statistical assumptions regarding the stock market may be inappropriate and eventually counter-productive, over the past two decades,

new methods for portfolio selection have been designed which make no statistical assumptions regarding the movement of stocks [36, 37, 61].

Although theoretical and empirical performance of such online portfolio selection algorithms have been encouraging, they have ignored one crucial practical aspect of financial trading: transaction costs. These online algorithms [36, 61, 32, 2, 18] could be trading aggressively and a major concern is the cost they would incur in a real world scenario.

In this chapter, we introduce an online portfolio selection algorithm *with transaction costs*. The algorithm is penalized by a fixed percentage of the amount of transactions it makes on a per day basis. We pose this as a non-smooth online convex optimization problem and propose an efficient algorithm called Online Lazy Updates (OLU) to make lazy updates to our online portfolio. Furthermore, we go on to prove that our lazy portfolio is competitive with reasonable strategies which have the benefit of hindsight. We conduct extensive experiments on two real world datasets: 22 years of the benchmark NYSE dataset with 36 stocks and 20 years of a S&P500 dataset with 263 stocks. Our experiments show that our lazy portfolios are scalable with transaction costs and, interestingly, in some cases, can outperform their non-lazy counterparts in terms of wealth achieved. Our algorithm is especially beneficial for individual investors who are typically affected by transaction costs.

We arrange the rest of the chapter as follows. We recap the online portfolio selection framework in a costless environment in section 3.2. In section 3.3, we describe our framework with transaction costs, discuss related work, and propose our Online Lazy Updates (OLU) algorithm. We outline its analysis in section 3.4. Section 3.5 contains details of our experiments and their results. We conclude with directions of our future work in Section 3.6.

3.2 Online Portfolio Selection

We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over a span of T periods. For ease of exposition, we will consider a period to be a day, but the analysis presented holds for any valid definition of a 'period' such as an hour or a month. Let $x_t(i)$ denote the *price relative* of stock s_i in day t, i.e., the multiplicative factor by which the price

of s_i changes in day t. Hence, $x_t(i) > 1$ implies a gain, $x_t(i) < 1$ implies a loss, and $x_t(i) = 1$ implies the price remained unchanged. We assume, $x_t(i) > 0 \ \forall \ i, t$. Let $\mathbf{x}_t = \langle x_t(1), \ldots, x_t(n) \rangle$ denote the vector of price relatives for day t, and let $\mathbf{x}_{1:t}$ denote the collection of such price relative vectors up to and including day t. A portfolio $\mathbf{p}_t = \langle p_t(1), \ldots, p_t(n) \rangle$ on day t can be viewed as a probability distribution over the stocks that prescribes investing $p_t(i)$ fraction of the current wealth in stock s_i. Note that the portfolio $\mathbf{p}_t$ has to be decided before knowing $\mathbf{x}_t$ which will be revealed only at the end of the day. The multiplicative gain in wealth at the end of day t, is then simply $\mathbf{p}_t^T \mathbf{x}_t = \sum_{i=1}^{n} p_t(i) x_t(i)$. For a sequence of price relatives $\mathbf{x}_{1:t-1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}$ up to day $(t-1)$, the sequential portfolio selection problem in day t is to determine a portfolio $\mathbf{p}_t$ based on past performance of the stocks. At the end of day t, $\mathbf{x}_t$ is revealed and the actual performance of $\mathbf{p}_t$ gets determined by $\mathbf{p}_t^T \mathbf{x}_t$. Over T periods, for a sequence of portfolios $\mathbf{p}_{1:T} = \{\mathbf{p}_1, \ldots, \mathbf{p}_T\}$, the multiplicative gain in wealth and the logarithmic gain in wealth is then,

$$S_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \prod_{t=1}^{T} \left(\mathbf{p}_t^T \mathbf{x}_t \right) \tag{3.1}$$

$$LS_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \sum_{t=1}^{T} \log \left(\mathbf{p}_t^T \mathbf{x}_t \right) \tag{3.2}$$

respectively. Ideally, for a costless environment (no transaction costs) we would like to maximize

$LS_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T})$ over $\mathbf{p}_{1:T}$. However, online portfolio selection cannot be posed as an optimization problem due to the temporal nature of the choices: $\mathbf{x}_t$ is not available when one has to decide on $\mathbf{p}_t$. Further, in a stock market, (statistical) assumptions regarding $\mathbf{x}_t$ can be difficult to make.

3.3 Online Portfolio with Transaction Costs

Typically, there can be two types of transaction costs in real markets: (1) a *fixed percentage* of each transaction that the investor has to pay to a broker or (2) a *fixed amount* paid per transaction (sell or buy). In this work we look at costs of the first type also known as *proportional transaction costs* in financial modeling [42, 89]. To fully specify our model for online portfolio selection with transaction costs, we proceed by

discussing related work, our problem formulation, followed by our Online Lazy Updates (OLU) algorithm and its analysis.

3.3.1 Related Work

The need for considering transaction costs in the design and analysis of online portfolio selection algorithms has been raised in [61, 37, 97, 32, 2, 18]. So far, only [16] have extended the analysis of [36] to include proportional transaction costs. Their strategy involved first computing a target portfolio using [36] and then paying for the transactions proportionally from each stock. Their analysis shows that the performance guarantee of the Universal Portfolio still holds (and gracefully degrades) in the case of proportional commissions. However, [36] is computationally demanding and has been shown to have sobering empirical performance [61, 40]. [16] and heuristics like Anticor [18] and OLMAR [81] do not account for transaction costs in their algorithm design. Anticor and OLMAR rely on empirical results to show scalability of their strategies to small transaction costs only as a post-processing step.

3.3.2 Problem Formulation

We present a general formulation for our online lazy updates problem and go on to show how portfolio selection with transaction costs is a special case of this setting. In an online lazy updates setting the optimization proceeds in rounds where in round t the algorithm has to pick a solution, $\theta_t \in F$, from the feasible set such that it is close to the previous solution θ_{t-1}. Nature then reveals the convex loss function, ϕ_t, and we observe its value $\phi_t(\theta_t)$. Ideally, over T rounds we would like to minimize the quantity,

$$\sum_{t=1}^{T} \phi_t(\theta_t) + \gamma \sum_{t=2}^{T} ||\theta_t - \theta_{t-1}||_1 \tag{3.3}$$

The ℓ_1 penalty term ensures that the updates to the solution θ_t are lazy. Absolute minimization of (4.2) is not reasonable because we do not know the sequence of ϕ_t *a priori*. If the ϕ_ts are known, (4.2) reduces to a batch optimization problem: a special case is the fused lasso when ϕ_t is quadratic [115]. Hence, over T iterations we intend to

get a sequence of θ_t such that the following *regret bound* is sublinear in T,

$$R_T = \sum_{t=1}^{T} f_t(\theta_t) - \min_{\theta^*} \sum_{t=1}^{T} f_t(\theta^*) \leq o(T) \tag{3.4}$$

where $f_t(\theta) = \phi_t(\theta) + \gamma||\theta - \theta_{t-1}||_1$ is non-smooth. θ^* is the minimizer of $\sum_{t=1}^{T} f_t$ in hindsight. Note that while the θ_ts can change over time, θ^* is fixed. That is, the minimizer, $\theta^* = \text{argmin}_\theta \sum_{t=1}^{T} f_t(\theta) = \text{argmin}_\theta \sum_{t=1}^{T} \phi_t(\theta)$, since it incurs zero ℓ_1 penalty in every iteration. We well refer to this as the *non-shifting regret bound*.

Alternatively, over T iterations we intend to get a sequence of θ_t such that the following *shifting regret bound* is sublinear in T,

$$R_T = \sum_{t=1}^{T} f_t(\theta_t) - \min_{\theta_1^*, \cdots, \theta_T^*} \sum_{t=1}^{T} f_t(\theta_t^*) \leq o(T) \tag{3.5}$$

where the sequence $\{\theta_1^*, \cdots, \theta_T^*\}$ is the minimizer of $\sum_{t=1}^{T} f_t$ as it has the power of hinsight. The bound in (3.4) is a *shifting bound* [62] since it allows the $\{\theta_1^*, \cdots, \theta_T^*\}$ to also lazily change over time.

Online portfolio selection with transaction costs can now be viewed as a special case of our online lazy updates setting where $f_t(\mathbf{p}) = -\log(\mathbf{p}^T \mathbf{x}_t) + \gamma||\mathbf{p} - \mathbf{p}_{t-1}||_1$. The ℓ_1 penalty term on the difference of two consecutive portfolios measures the fraction of wealth traded. The parameter γ controls the amount that can be traded every day. Note that on setting $\gamma = 0$, our formulation reduces to the costless case as seen in (4.1).

3.3.3 Online Lazy Update (OLU) Algorithm

We now formulate an online lazy portfolio selection strategy that allows us to control the total amount of transaction everyday. It decides to trade or not depending on if the benefits of changing the portfolio outweigh the transaction costs. We find a new lazy portfolio vector $\mathbf{p}_{t+1}$ as follows:

$$\mathbf{p}_{t+1} = \underset{\mathbf{p} \in \triangle_n}{\text{argmin}} \quad -\log(\mathbf{p}^T \mathbf{x}_t) \; + \; \gamma||\mathbf{p} - \mathbf{p}_t||_1 + \frac{1}{2\eta} \; ||\mathbf{p} - \mathbf{p}_t||_2^2 \tag{3.6}$$

This can be rewritten as,

$$\mathbf{p}_{t+1} = \underset{\mathbf{p} \in \triangle_n}{\text{argmin}} \quad -\eta \log(\mathbf{p}^T \mathbf{x}_t) \; + \; \alpha||\mathbf{p} - \mathbf{p}_t||_1 + \frac{1}{2} \; ||\mathbf{p} - \mathbf{p}_t||_2^2, \tag{3.7}$$

Algorithm 1 Online Lazy Update (OLU) with ADMM

1: Input $\mathbf{p}_t, \mathbf{x}_t, \eta, \alpha, \beta$
2: Initialize $\mathbf{p}, \mathbf{z}, \mathbf{u} \in 0^n, k = 0$
3: ADMM iterations

$$\begin{aligned}
\mathbf{p}^{(k+1)} &= \prod\nolimits_{\triangle_n} \Big\{ - \frac{\eta \mathbf{x}_t}{(\beta+1)\mathbf{p}_t^T \mathbf{x}_t} + \mathbf{p}_t + \frac{\beta \mathbf{z}^{(k)}}{(\beta+1)} - \frac{\beta \mathbf{u}^{(k)}}{(\beta+1)} \Big\} \\
\mathbf{z}^{(k+1)} &= S_{\alpha/\beta}(\mathbf{p}^{(k+1)} - \mathbf{p}_t + \mathbf{u}^{(k)}) \\
\mathbf{u}^{(k+1)} &= \mathbf{u}^{(k)} + (\mathbf{p}^{(k+1)} - \mathbf{p}_t - \mathbf{z}^{(k+1)}) \ .
\end{aligned}$$

where $\prod_{\triangle_n}$ is a projection to the simplex and S_ρ is the shrinkage operator.
4: Continue until **Stopping Criteria** is satisfied

which we will use from here on (where $\alpha = \eta * \gamma$). In this online framework, the new portfolio vector is computed as a function of $\mathbf{p}_t$ and the price relatives $\mathbf{x}_t$ and lies in the probability simplex in n-dimension. The purpose of the first term is to maximize the logarithmic wealth if the current price relative $\mathbf{x}_t$ is replicated. The second term is the ℓ_1 penalty which accounts for the amount of transaction that would take place to update to a new portfolio. The parameter $\alpha > 0$ decides how often we trade; high values of α lead to *lazy* updates of the portfolio with small amount of transactions while low values allow the portfolio to change more often. Our framework for updating a portfolio vector is analogous to the framework of the EG algorithm [61]. We use $||\cdot||_{\ell_2}^2$ as the distance function instead of the relative entropy in EG. Unlike EG, we solve a non-smooth problem.

We propose an ADMM (Alternating Direction Method of Multipliers [22]) based efficient primal-dual algorithm to obtain the lazy portfolio $\mathbf{p}_{t+1}$ by solving (3.7). ADMM is an efficient distributed optimization method closely related to Bregman iterative algorithms for l_1 problems and proximal point methods. It has been applied in many large scale problems in statistics and machine learning because of its computational benefits and fast convergence in practice [22]. We can rewrite (3.7) in the ADMM form by introducing an auxiliary variable z as,

$$\underset{\mathbf{p}\in\triangle_n, \mathbf{p}-\mathbf{p}_t=\mathbf{z}}{\operatorname{argmin}} \quad -\eta \log(\mathbf{p}^T \mathbf{x}_t) \ + \ \alpha||\mathbf{z}||_1 \ + \ \frac{1}{2}||\mathbf{p}-\mathbf{p}_t||_2^2 \tag{3.8}$$

This ADMM formulation naturally lets us decouple the non-smooth l_1 term from the

smooth terms, which is computationally advantageous.We replace the log term in (3.8) by its first order Taylor expansion around $\mathbf{p}_t$. The *augmented Lagrangian* for the above problem is then,

$$L_\beta(\mathbf{p},\mathbf{z},\mathbf{u}) = \underset{\mathbf{p}\in\triangle_n}{\operatorname{argmin}} -\eta\left(\log(\mathbf{p}_t^T\mathbf{x}_t) + \frac{\mathbf{x}_t^T(\mathbf{p}-\mathbf{p}_t)}{\mathbf{p}_t^T\mathbf{x}_t}\right) + \alpha||\mathbf{z}||_1 + \frac{1}{2}||\mathbf{p}-\mathbf{p}_t||_2^2 + \frac{\beta}{2}||\mathbf{p}-\mathbf{p}_t-\mathbf{z}+\mathbf{u}||_2^2 \tag{3.9}$$

where $u = \frac{1}{\beta}\lambda$ is the scaled dual variable and λ is the dual variable. ADMM consists of the following iterations for solving $\mathbf{p}_{t+1}$,

$$\mathbf{p}_{t+1}^{(k+1)} = \underset{\mathbf{p}\in\triangle_n}{\operatorname{argmin}} -\eta\left(\log(\mathbf{p}_t^T\mathbf{x}_t) + \frac{\mathbf{x}_t^T(\mathbf{p}-\mathbf{p}_t)}{\mathbf{p}_t^T\mathbf{x}_t}\right) + \frac{1}{2}||\mathbf{p}-\mathbf{p}_t||_2^2 + \frac{\beta}{2}||\mathbf{p}-\mathbf{p}_t-\mathbf{z}^{(k)}+\mathbf{u}^{(k)}||_2^2 \tag{3.10}$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \ \alpha||\mathbf{z}||_1 + \frac{\beta}{2}||\mathbf{p}_{t+1}^{(k+1)}-\mathbf{p}_t-\mathbf{z}+\mathbf{u}^{(k)}||_2^2 \tag{3.11}$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + (\mathbf{p}_{t+1}^{(k+1)}-\mathbf{p}_t-\mathbf{z}^{(k+1)}) \tag{3.12}$$

Algorithm 1 shows the closed form updates derived for $\mathbf{p}_{t+1}^{k+1}$, $\mathbf{z}^{k+1}$, and $\mathbf{u}^{k+1}$. The update for $\mathbf{p}_{t+1}^{k+1}$ is derived by taking the derivative of (4.9) and setting it to zero. The projection to the simplex ($\prod_{\triangle_n}$) is carried out as in [46]. The stopping criteria for the OLU algorithm is based on the primal and dual residuals from [22].

Algorithm 4 is our online portfolio selection algorithm with transaction costs. It uses the OLU Algorithm to compute the lazy updates to the portfolio $\mathbf{p}_{t+1}$. It takes in an additional parameter γ which is a fixed percentage charged for the total amount of transaction everyday. S_t^γ is the transaction cost-adjusted cumulative wealth gain at the end of t days.

3.4 Analysis

In this section we theoretically analyze the performance (regret) of our OLU algorithm with two classes of batch solutions. The first class is the best fixed solution which can see the entire sequence of data in hindsight. The second class is the best sequence

Algorithm 2 Portfolio Selection with Transaction costs

Input η, γ, β; Compute $\alpha = \eta\gamma$
Initialize $p_{1,h} = \frac{1}{n}, h = 1, \ldots, n; p_0 = p_1; S_0^\gamma = 1$
For $t = 1, \ldots, T$
    Receive $\mathbf{x}_t$ vector of price relatives
    Compute cumulative wealth: $S_t^\gamma = S_{t-1}^\gamma \times (\mathbf{p}_t^T \mathbf{x}_t) - \gamma \times S_{t-1}^\gamma \times ||\mathbf{p}_t - \mathbf{p}_{t-1}||_1$
    Update portfolio: $\mathbf{p}_{t+1} = \text{OLU}(\mathbf{p}_t, \eta, \alpha, \beta)$
end for

of solutions in hindsght: i.e., the comparator class can also change over time. Our aim is to show that we can achieve sub linear regret in both the cases. Our analysis, like our framework is general and applies to any online convex optimization problem with lazy updates. We consider 2 sub cases for our analysis with the fixed solution in hindsight: (1) when we consider general convex function ϕ_t and (2) when we assume a little more about the the convex function ϕ_t: in particular strong convexity. We will show that with the strong convexity assumption, we achieve sharper regret bound ($O(\log T)$) compared to general convex functions ($O(\sqrt{T})$).

We obtain a sequence of lazy solutions $\mathbf{p}_1, \cdots, \mathbf{p}_T$ from Algorithm 1 and on day t suffer a loss $f_t(\mathbf{p}_t) = \phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_t - \mathbf{p}_{t-1}\|_1$. Our first goal is to minimize the *regret* with respect to the best fixed (non-shifting) solution $\mathbf{p}^*$ in hindsight and then we go on to analyze regret with respect to the best sequence of (shifting) solutions $\{\mathbf{p}_1^*, \cdots, \mathbf{p}_T^*\}$ in hindsight.

In this chapter, we consider updates of the following general form:

$$\mathbf{p}_{t+1} = \underset{\mathbf{p}\in\mathcal{P}}{\text{argmin}} \left\{ \eta\langle \nabla\phi_t(\mathbf{p}_t), \mathbf{p}\rangle + \eta\gamma\|\mathbf{p} - \mathbf{p}_t\|_1 + d_\psi(\mathbf{p}, \mathbf{p}_t) \right\}, \tag{3.13}$$

For our online lazy updates formulation, the Bregman divergence $d_\psi(\mathbf{p}, \mathbf{p}_t) = \frac{1}{2}||\mathbf{p} - \mathbf{p}_t||_2^2$. For the portfolio selection problem $\phi_t(\mathbf{p}_t) = -\log(\mathbf{p}_t^T \mathbf{x}_t)$ but our analysis holds for any convex ϕ_t.

In this section, we first establish the standard non-shifting regret bounds for our lazy updates using Theorem 1 and Theorem 2. We then go on to establish the shifting bounds using Theorem 6. For the non-shifting bounds, we focus on two sub-cases: (1) when ϕ_ts are *general* convex functions and (2) ϕ_ts are *strongly* convex functions.

3.4.1 Non-shifting Regret for OLU Algorithm

We first focus on the case where the comparator class is fixed or non-shifting, i.e., $\mathbf{p}^*$ is the minimizer of $\sum_{t=1}^{T} f_t$ in hindsight as it incurs zero ℓ_1 penalty in every iteration and we prove *regret bounds* as in (3.4). We show that for general convex functions the regret is $O(\sqrt{T})$ while for strongly convex functions the regret is $O(\log T)$.

3.4.1.1 General Convex Functions

We assume that ϕ_t are general convex functions with bounded (sub)gradients, i.e., for any $\hat{g} \in \partial\phi_t(\mathbf{p})$ we have $\|\hat{g}\| \leq G$.

Lemma 1. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let $d_\psi(\cdot,\cdot)$ is λ-strongly convex with respect to norm $||\cdot||$, i.e. $d_\psi(\mathbf{p},\hat{\mathbf{p}}) \geq \frac{\lambda}{2}\|\mathbf{p}-\hat{\mathbf{p}}\|_2^2$ and $||\mathbf{p}-\hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p},\hat{\mathbf{p}} \in \mathcal{P}$. Then, for any $\mathbf{p}^* \in \mathcal{P}$,*

$$\eta[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \leq d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\gamma L + \frac{\eta^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2. \tag{3.14}$$

The proof of this lemma can be found in the appendix.

Theorem 1. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let ϕ_t be a Lipschitz continuous function for which $\|\nabla\phi_t(\mathbf{p}_t)\|_2^2 \leq G$. Then, by choosing $\eta \propto \frac{1}{\sqrt{T}}$ and $\gamma \propto \frac{1}{\sqrt{T}}$, we have*

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \leq O(\sqrt{T}) \tag{3.15}$$

Proof. By Lemma 1, we have

$$\begin{aligned}\eta\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] &\leq \sum_{t=1}^{T} d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\gamma LT + \frac{\eta^2G^2T}{2\lambda} \\ &= \sum_{t=1}^{T} d_\psi(\mathbf{p}^*,\mathbf{p}_1) - d_\psi(\mathbf{p}^*,\mathbf{p}_{T+1}) + \eta\gamma LT + \frac{\eta^2G^2T}{2\lambda}\end{aligned}$$

Noting, that Bregman divergences are always non-negative and dropping the $\gamma\|\mathbf{p}_{T+1} - \mathbf{p}_T\|_1$ terms we have,

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*)] + \gamma\sum_{t=1}^{T-1}[\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1] \leq \frac{1}{\eta}d_\psi(\mathbf{p}^*, \mathbf{p}_1) + \gamma LT + \frac{\eta G^2 T}{2\lambda}$$
$$\leq \sqrt{T}d_\psi(\mathbf{p}^*, \mathbf{p}_1) + L\sqrt{T} + \frac{G^2\sqrt{T}}{2\lambda}$$

We get the last inequality by substituting $\eta \propto \frac{1}{\sqrt{T}}$ and $\gamma \propto \frac{1}{\sqrt{T}}$. □

Note: One can choose an adaptive $\gamma_t = \frac{\gamma_0}{t}$ in Algorithm 1 and then we have,

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*)] + \gamma\sum_{t=1}^{T-1}[\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1] \leq \frac{1}{\eta}d_\psi(\mathbf{p}^*, \mathbf{p}_1) + L\sum_{t=1}^{T}\gamma_t + \frac{\eta G^2 T}{2\lambda}$$
$$\leq \sqrt{T}d_\psi(\mathbf{p}^*, \mathbf{p}_1) + \gamma_0 L\sum_{t=1}^{T}\frac{1}{t} + \frac{G^2\sqrt{T}}{2\lambda}$$
$$\leq O(\sqrt{T}),$$

where $\{\gamma_t\}$ can be any permutation of $\{\frac{\gamma_0}{t}\}$ without affecting the analysis and γ_0 can be chosen so that its large enough to make the updates sparse.

3.4.1.2 Strongly Convex Functions

We assume that ϕ_t are all β-strongly convex functions so that for any $(\mathbf{p}, \mathbf{p}_t)$

$$\phi_t(\mathbf{p}) \geq \phi_t(\mathbf{p}_t) + \langle \mathbf{p} - \mathbf{p}_t, \nabla\phi_t(\mathbf{p}_t)\rangle + \frac{\beta}{2}\|\mathbf{p} - \mathbf{p}_t\|^2 . \tag{3.16}$$

Lemma 2. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let $d_\psi(\cdot,\cdot)$ is λ-strongly convex with respect to norm $||\cdot||$, i.e. $d_\psi(\mathbf{p}, \hat{\mathbf{p}}) \geq \frac{\lambda}{2}\|\mathbf{p} - \hat{\mathbf{p}}\|_2^2$ and $||\mathbf{p} - \hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p}, \hat{\mathbf{p}} \in \mathcal{P}$. Assuming ϕ_t are all β-strongly convex, for any $\gamma < \frac{\beta}{4}$ and any $\mathbf{p}^* \in \mathcal{P}$, we have*

$$\begin{aligned}&\eta_t[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1] \\ &\leq d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2 - \eta_t\left(\frac{\beta}{2} - 2\gamma\right)\|\mathbf{p}^* - \mathbf{p}_t\|^2 .\end{aligned} \tag{3.17}$$

The proof of this lemma can be found in the appendix.

Theorem 2. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* ϕ_t *be all* β*-strongly convex and* $\|\nabla\phi_t(\mathbf{p}_t)\|_2^2 \leq G$*. Then, for any* $\gamma < \beta/4$*, choosing* $\eta_t = \frac{2}{\kappa t}$*, where* $\kappa \in (0, \frac{\beta}{4} - \gamma]$*, we have*

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \leq O(\log T) \tag{3.18}$$

Proof. By Lemma 2, we have

$$\begin{aligned}
&\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \\
&\leq \sum_{t=1}^{T}\frac{1}{\eta_t}d_\psi(\mathbf{p}^*, \mathbf{p}_t) - \frac{1}{\eta_t}d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) - \left(\frac{\beta}{2} - 2\gamma\right)\|\mathbf{p}^* - \mathbf{p}_t\|^2 + \frac{\eta_t}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2 \\
&\leq \frac{1}{\eta_1}d_\psi(\mathbf{p}^*, \mathbf{p}_1) - \frac{1}{\eta_T}d_\psi(\mathbf{p}^*, \mathbf{p}_{T+1}) \\
&\qquad + \sum_{t=1}^{T-1}\left[d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})\left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) - \left(\frac{\beta}{2} - 2\gamma\right)\|\mathbf{p}^* - \mathbf{p}_t\|^2\right] + \frac{G^2}{2\lambda}\sum_{t=1}^{T}\eta_t \\
&\leq \frac{1}{\eta_1} \leq d_\psi(\mathbf{p}^*, \mathbf{p}_1) - \sum_{t=1}^{T-1}2\left(\frac{\beta}{4} - \gamma - \kappa\right)\|\mathbf{p}^* - \mathbf{p}_{t+1}\|^2 + \frac{cG^2}{\lambda\kappa}\log T \ ,
\end{aligned}$$

where $\sum_{t=1}^{T-1}\left[d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})\left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) - 2\kappa d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})\right] = 0$ and we have used $d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) = \frac{1}{2}\|\mathbf{p}^* - \mathbf{p}_{t+1}\|^2$ (for simplicity) and also used $\sum_{t=1}^{T}\frac{1}{t} = c\log T$. Now, since $(\frac{\beta}{4} - \gamma - \kappa) \geq 0$, we have

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \leq \kappa d_\psi(\mathbf{p}^*, \mathbf{p}_1) + \frac{cG^2}{\lambda\kappa}\log T = O(\log T) \ ,$$

which completes the proof. □

3.4.2 Shifting Regret for OLU Algorithm

We now focus on the case where the comparator class can also shift or change over time, i.e., the sequence $\{\mathbf{p}_1^*, \cdots, \mathbf{p}_T^*\}$ is the minimizer of $\sum_{t=1}^{T} f_t$ in hindsight and hence

incurs non-zero ℓ_1 penalty in every iteration and we prove *regret bounds* as in (3.5) as follows:

$$\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}_t^*)]+\sum_{t=1}^{T-1}[\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\gamma\|\mathbf{p}_{t+1}^*-\mathbf{p}_t^*\|_1]\leq c_2\text{size}(\langle\mathbf{p}_1^*,\cdots,\mathbf{p}_T^*\rangle)+o(T), \tag{3.19}$$

where size($\langle\mathbf{p}_1^*,\cdots,\mathbf{p}_T^*\rangle$) intuitively measures the amount of shifting that occurs in the best sequence of solutions in hindsight. In this chapter, our analysis uses the following measure of size($\langle\mathbf{p}_1^*,\cdots,\mathbf{p}_T^*\rangle$):

$$|||\langle\mathbf{p}_1^*,\cdots,\mathbf{p}_T^*\rangle|||_b=\sum_{t=1}^{T-1}\|\mathbf{p}_t^*-\mathbf{p}_{t+1}^*\|_b. \tag{3.20}$$

3.4.2.1 General Convex Functions

We assume that ϕ_t are general convex functions with bounded (sub)gradients, i.e., for any $\hat{g}\in\partial\phi_t(\mathbf{p})$ we have $\|\hat{g}\|\leq G$.

Theorem 3. *Let $\{\mathbf{p}_1^*,\cdots,\mathbf{p}_T^*\}$ be the best sequence obtained by minimizing (4.2). For, $\eta\propto\frac{1}{\sqrt{T}}$ and $\kappa\propto\frac{1}{\sqrt{T}}$, $\frac{1}{a}+\frac{1}{b}=1$ and $||\bigtriangledown\psi(\mathbf{p}_t)||_a\leq\zeta$, we have*

$$\begin{aligned}&\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}_t^*)]+\sum_{t=1}^{T-1}[\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\gamma\|\mathbf{p}_{t+1}^*-\mathbf{p}_t^*\|_1]\\&\leq O(\sqrt{T})+\frac{1}{\eta}\Big\{d_\psi(\mathbf{p}_1^*,\mathbf{p}_1)-d_\psi(\mathbf{p}_T^*,\mathbf{p}_{T+1})+\psi(\mathbf{p}_T^*)-\psi(\mathbf{p}_1^*)+\zeta\sum_{t=1}^{T-1}||\mathbf{p}_t^*-\mathbf{p}_{t+1}^*||_b\Big\}\end{aligned} \tag{3.21}$$

Proof. From, Lemma 1 we have the following by substituting $\mathbf{p}^*=\mathbf{p}_t^*$

$$\eta[\phi_t(\mathbf{p}_t)+\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\phi_t(\mathbf{p}_t^*)]\leq d_\psi(\mathbf{p}_t^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_t^*,\mathbf{p}_{t+1})+\eta\gamma L+\frac{\eta^2}{2\alpha}||\nabla\psi_t(\mathbf{p}_t)||^2 \tag{3.22}$$

This does not telescope, so we add

$$\psi(\mathbf{p}_t^*)-\psi(\mathbf{p}_{t+1}^*)-||\bigtriangledown\psi(\mathbf{p}_{t+1})||_a||\mathbf{p}_t^*-\mathbf{p}_{t+1}^*||_b\leq d_\psi(\mathbf{p}_t^*,\mathbf{p}_{t+1})-d_\psi(\mathbf{p}_{t+1}^*,\mathbf{p}_{t+1}) \tag{3.23}$$

We use Holder's inequality for $(\mathbf{p}_{t+1}^*-\mathbf{p}_t^*)^T\nabla\psi(\mathbf{p}_{t+1})\geq-||\bigtriangledown\psi(\mathbf{p}_{t+1})||_a||\mathbf{p}_t^*-\mathbf{p}_{t+1}^*||_b$.

Adding (4.28), (4.29) and then adding $-\eta\gamma\|\mathbf{p}^*_{t+1}-\mathbf{p}^*_t\|_1$ on both sides and summing over $T-1$ rounds, we have

$$\eta\sum_{t=1}^{T-1}[\phi_t(\mathbf{p}_t)+\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\phi_t(\mathbf{p}^*_t)-\gamma\|\mathbf{p}^*_{t+1}-\mathbf{p}^*_t\|_1]+\psi(\mathbf{p}^*_1)-\psi(\mathbf{p}^*_T)-\zeta\sum_{t=1}^{T-1}||\mathbf{p}^*_t-\mathbf{p}^*_{t+1}||_b$$
$$\leq d_\psi(\mathbf{p}^*_1,\mathbf{p}_1)-d_\psi(\mathbf{p}^*_T,\mathbf{p}_T)+\eta\gamma\sum_{t=1}^{T-1}L+\frac{\eta^2}{2\alpha}\sum_{t=1}^{T-1}G^2 \tag{3.24}$$

where we bound $\|\mathbf{p}^*_{t+1}-\mathbf{p}^*_t\|_1\leq L$ and ignore negative terms on the right hand side of the inequality. Adding 3.24 and (4.28) over the round $t=T$, dropping $\eta\kappa\|\mathbf{p}_{T+1}-\mathbf{p}_T\|_1$ and rearranging, we have

$$\eta\sum_{t=1}^{T}[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}^*_t)]+\sum_{t=1}^{T-1}[\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\gamma\|\mathbf{p}^*_{t+1}-\mathbf{p}^*_t\|_1]$$
$$\leq d_\psi(\mathbf{p}^*_1,\mathbf{p}_1)-d_\psi(\mathbf{p}^*_T,\mathbf{p}_{T+1})+\psi(\mathbf{p}^*_T)-\psi(\mathbf{p}^*_1)+\zeta\sum_{t=1}^{T-1}||\mathbf{p}^*_t-\mathbf{p}^*_{t+1}||_b+\eta\gamma\sum_{t=1}^{T}L+\frac{\eta^2}{2\alpha}\sum_{t=1}^{T}G^2 \tag{3.25}$$

Setting $\eta\propto\frac{1}{\sqrt{T}}$ and $\kappa\propto\frac{1}{\sqrt{T}}$ and dividing both sides by η, completes the proof.

□

3.5 Experiments and Results

In this section we present experimental results with our OLU algorithm for transaction cost adjusted portfolio selection on two real world datasets. We first describe our datasets and methodology and then go on to discuss in details the experimental results.

η	Weight on logarithmic gain in wealth.
γ	Fixed transaction (pre-specified) cost expressed as a percentage.
α	Weight on ℓ_1 penalty term: computed in Algorithm 2 as $\eta\times\gamma$.
β	Weight on the augmented lagrangian: parameter for OLU algorithm with ADMM.

Table 3.1: Parameter descriptions as given in (3.7) and used in Algorithm 1 and 2.

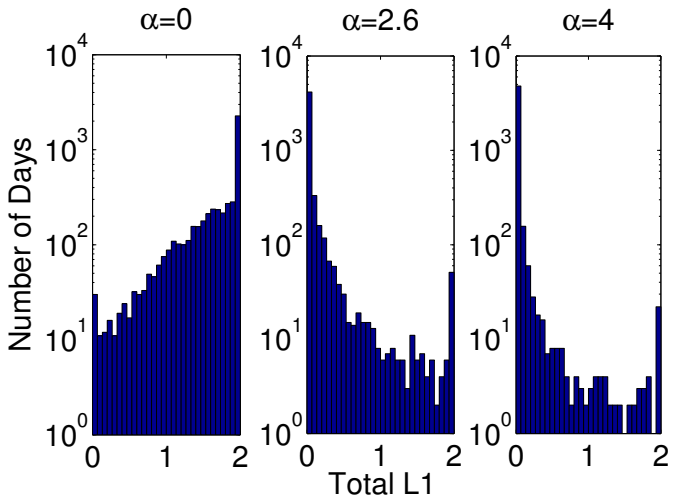
3.5.1 Datasets

The experiments were conducted on two real-world datasets: the New York Stock Exchange (NYSE) [36] and the Standard & Poor's 500 (S&P 500) [40] datasets. The NYSE dataset consists of 36 stocks with data accumulated over a period of 22 years from July 3, 1962 to December 31, 1984. The dataset captures the bear market that lasted between January 1973 and December 1974. However, all of the 36 stocks increase in value in the 22-year run. This is a benchmark dataset that has been used extensively in the online portfolio selection literature for demonstration of empirical results [61, 2, 18, 36]. The S&P500 dataset consists of 263 stocks which were present in the S&P500 index in 2010 and were alive since 1990. This period of 20 years from 1990 to 2010 covers the bull and bear markets of recent times such as the dot-com bubble which occurred between 1997-2000, the following bubble burst during March 2000, and the recent housing bubble burst occurring between 2006-2007.

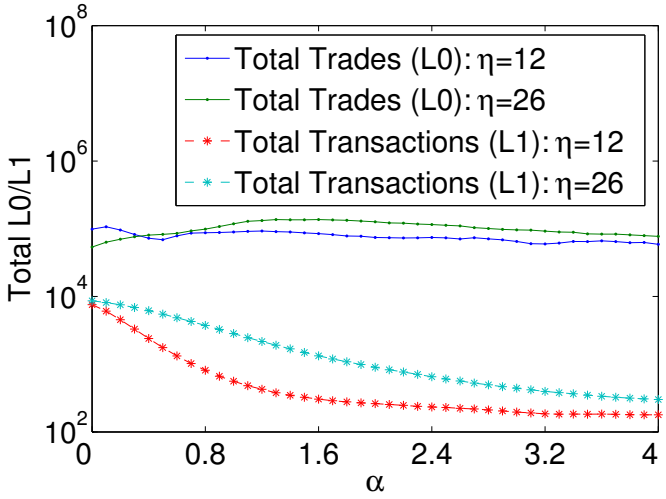
3.5.2 Methodology and Parameter Setting

In all our experiments we start with \$1 as our initial investment and an initial portfolio which is uniformly distributed over all the stocks. We use Algorithm 4 to obtain our portfolios sequentially and compute the transaction cost-adjusted wealth for each day. The parameters consist of $\boldsymbol{\eta}$: weight on logarithmic gain in wealth, $\boldsymbol{\gamma}$: fixed percentage transaction cost, and $\boldsymbol{\beta}$: the parameter for the augmentation term of the OLU algorithm. For all our experiments, we set $\beta = 0.1$ which we found to give reasonable accuracy. Table 3.1 contains the description of the various parameters.

Since the two datasets are very different in nature (stock composition and duration), we experimented extensively with a large range of η and α values to observe their effect on the lazy updates of our portfolio. Moreover, we chose a reasonable range of γ values (in percentage) to compute the proportional transaction costs incurred due to the portfolio update every day. The range of γ values we experimented with were between 0% and 2%. We have illustrated some of our results with representative plots from either the NYSE or S&P500 dataset.

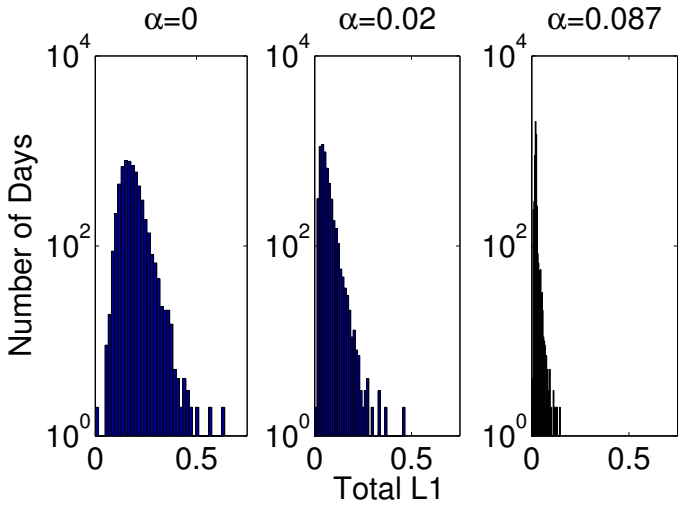


(a) Histogram of ℓ_1 values for the S&P500 dataset.

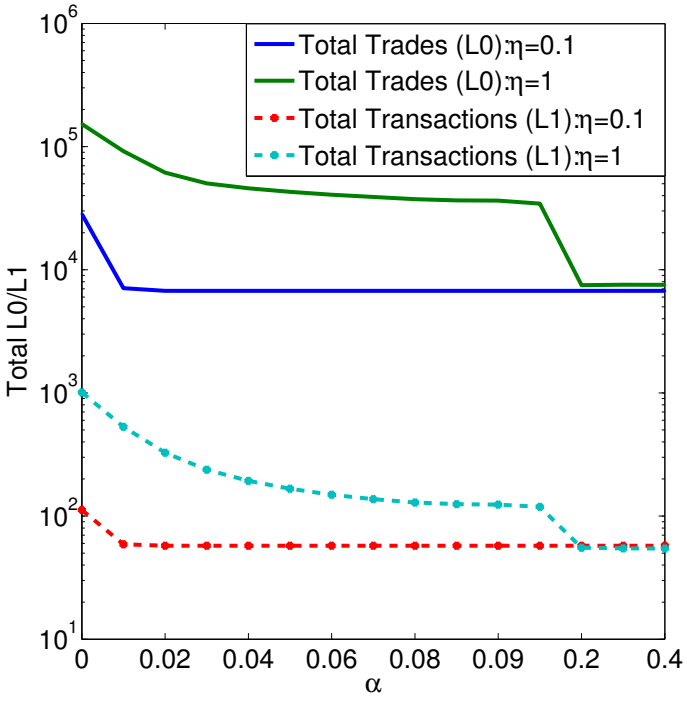


(b) Number of trades (ℓ_0 norm) and amount of trade (ℓ_1 norm) for varying α (S&P500 dataset).

Figure 3.1: Effect of α: as α increases, the total amount of the transaction decreases but the total number of trades may not decrease monotonically for S&P500 dataset.



(a) Histogram of ℓ_1 values for the NYSE dataset.



(b) Number of trades (ℓ_0 norm) and amount of trade (ℓ_1 norm) for varying α (NYSE dataset).

Figure 3.2: Effect of α: as α increases, the total amount of the transaction decreases and we observe that the total number of trades also decreases monotonically for NYSE dataset unlike S&P500.

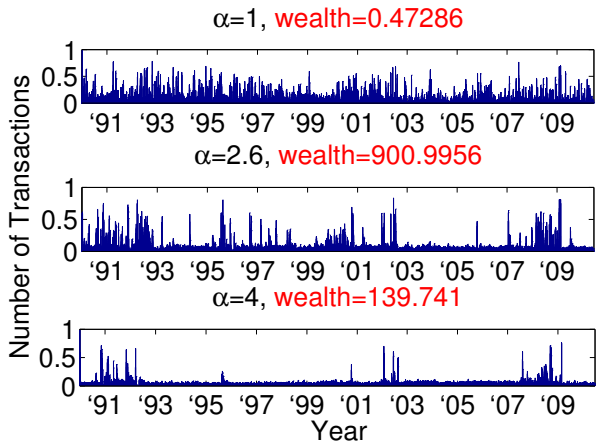
We use the wealth obtained (without transaction costs) EG algorithm and a Buy-and-Hold strategy as benchmarks for our experiments. EG has been shown to outperform a uniform constantly rebalanced portfolio [61, 40]. For the Buy-and-Hold case we start with a uniformly distributed portfolio and do a hold on the positions thereafter (i.e. no trades). We also use the S&P500 Index as a representative index for the US stock market to analyze the activity of our lazy update algorithm. We do not compare our method with Anticor or OLMAR because these heuristics do not account for transaction costs in their algorithmic framework and have no theoretical guarantees. We also do not compare with the Universal Portfolio algorithm because it has also been shown to have sobering empirical performance even in the absence of transaction costs when compared to EG and Buy-and-Hold [61, 40].

3.5.3 Effect of α and the ℓ_1 penalty

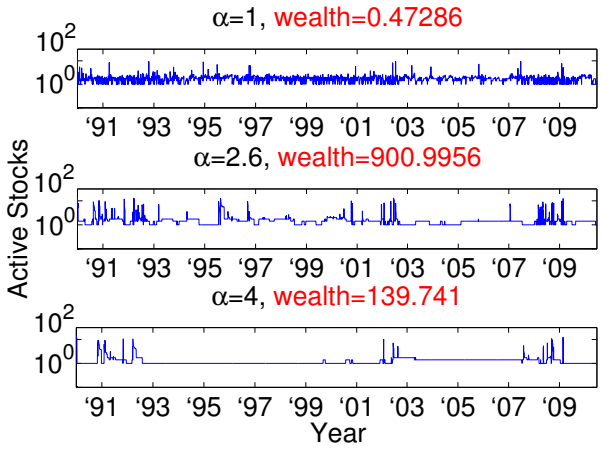
The parameter α is the weight on the ℓ_1 penalty term and can influence (a) the total amount of transactions, (b) the total amount of trades and transactions, (c) the daily amount of transactions, and (d) the stock activity. We now investigate the effect of α in the two datasets.

(a) Total Amount of Transactions: Let $\Upsilon_t = ||\mathbf{p}_{t+1} - \mathbf{p}_t||_1$, then $\sum_{t=1}^{T-1} \Upsilon_t$ is a measure of the total amount that a trader had to pay in transaction costs over T days (as a fraction of his wealth). Figure 3.1(a) plots a histogram of Υ_t for varying α values for S&P500 dataset. We observe that as α increases, the Υ_t value is small for most days. With $\alpha = 0$, Υ_t was 2 for most days denoting non-lazy portfolios which is how portfolios are expected to trade in a costless environment. The plot for the NYSE dataset in Figure 3.2(a) shows an identical trend.

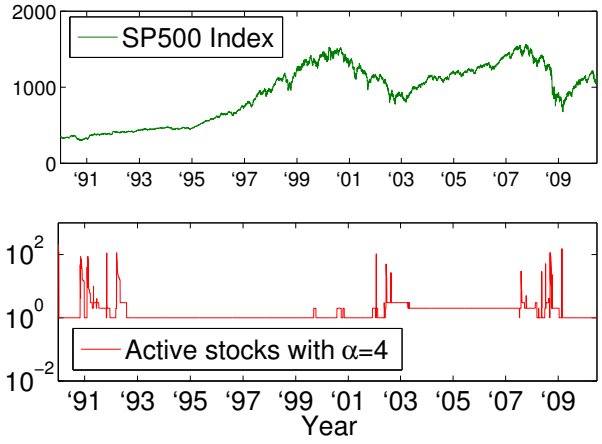
(b) Total Amount of Trades and Transactions: We now analyze the behavior of the total number of trades (ℓ_0 norm) and the total amount of transactions ($\sum_{t=1}^{T-1} \Upsilon_t$, ℓ_1 norm) as we increase the value of α. Figure 3.1(b) gives a holistic overview of the behavior of the aforementioned quantities as we increase α. Figure 3.1(b) and 3.2(b) confirms that the total ℓ_1 norm decreases as we increase α. The total ℓ_0 norm, however, does not always decrease as we increase α. Figure 3.1(b) shows such a situation for the S&P500 dataset. For the NYSE dataset in Figure 3.2(b), however we observe that the ℓ_0 norm also decreases as we increase α.



(a) Fraction of transactions per day.



(b) Stocks which comprise 80% of the wealth.



(c) S&P 500 Index value vs. Active stocks.

Figure 3.3: As α increases, there is a decline in the number of transactions and OLU tends to hold on to stocks interspersed with days of high activity (transactions). Days of high stock activity coincides with major movements in the market.

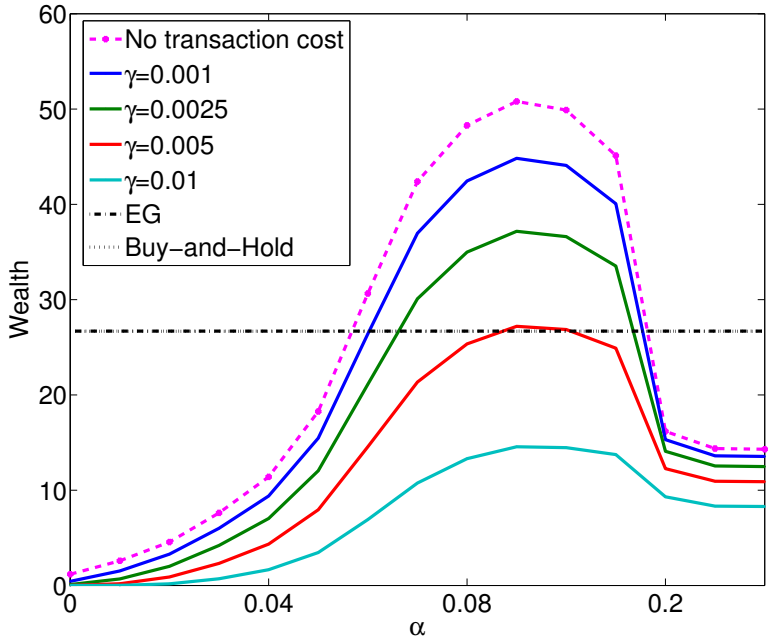
(c) Daily Amount of Transactions: Figure 3.3(a) plots the fraction of stocks traded per day for the S&P500 dataset for three values of α. We observe that as we increase the weight on the ℓ_1 penalty term by tuning our parameter α, the number of transactions decreases. Whereas a large amount of the 263 stocks were traded everyday for $\alpha = 0$, with higher values of α the number of transactions reduces significantly. We observe a similar trend for the NYSE dataset.

(d) Active Stocks: Figure 3.3(b) plots the number of stocks which comprise 80% of the total wealth on a per day basis which we call the *active* stocks. As α increases, the lazy behavior of the portfolios becomes more apparent. We observe that high weight on the ℓ_1 penalty term forces the online portfolios to change their composition only on a handful of days.

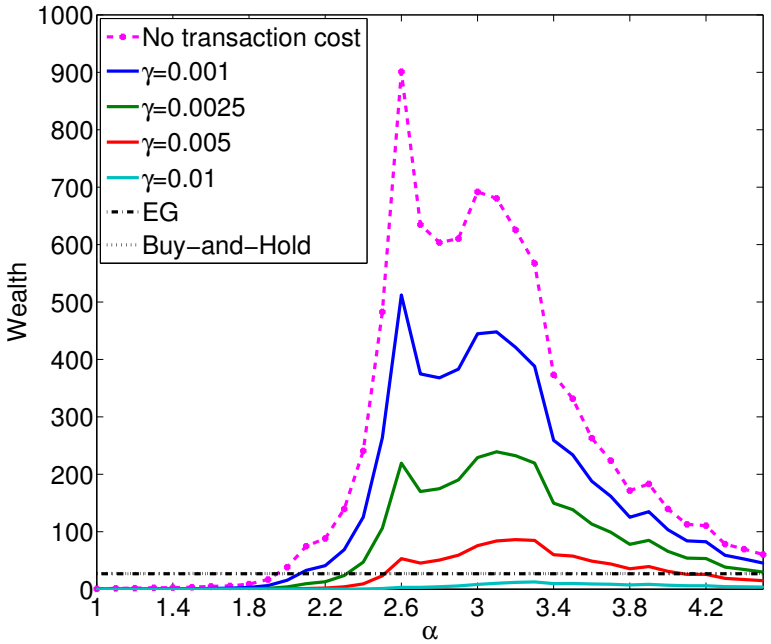
Correlation with the US market: In Figure 3.3(b), we observe significant activity between years 2002-2003 and between years 2008-2009. On plotting the value of the S&P500 index for the US market between 1990 and 2010 in Figure 3.3(c), we realized that the increase in trading activity reflected two major market movements: the dot-com and housing bubble bursts. Figure 3.3(c) shows that the days of high stock activity coincides with major market movements. Similar trends were observed for the NYSE dataset.

3.5.4 Wealth with Transaction Costs (S_T^γ)

To evaluate the practical application of our proposed algorithm, we now analyze its performance when calculating the transaction cost-adjusted cumulative wealth. Figure 3.4 shows how the choice of different α values affect the transaction cost-adjusted cumulative wealth for the two datasets (for a fixed η value). Figures 4.5 and 3.4(b) demonstrate that there exists a regime of $\alpha = \eta\gamma$ which makes an optimum choice between exploration and exploitation of stocks. Since γ can be fixed, the learning rate η can be adequately chosen to maximize our wealth. Very low values of η tends to aggressively change the portfolio too often. Whereas, with very high values of η, the algorithm becomes too conservative and might not be able to take advantage of short trends in the market. For a fixed α, decreasing η makes the portfolios lazy (higher γ) and increasing η (lower γ) encourages trading.



(a) Transaction cost adjusted wealth: S_T^γ for NYSE dataset.



(b) Transaction cost adjusted wealth: S_T^γ for S&P500 dataset

Figure 3.4: OLU with transaction costs can outperform (in terms of wealth) EG and Buy-n-Hold (without transaction costs).

EG and Buy-and-Hold: We compared the total wealth without transaction costs of OLU with that of EG and a Buy-and-Hold strategy. EG and Buy-and-Hold are plotted as horizontal lines and we can see that for the NYSE dataset EG returns \$26.70 and Buy-and-Hold returns \$26.78. For the S&P500 dataset EG returns \$26.68 and Buy-and-Hold returns \$27.25. In comparison, OLU returns \$50.80 and \$901.00 respectively without transaction costs. OLU returns almost 2x as much wealth for the NYSE dataset and 33x as much wealth for the S&P500 dataset as EG or Buy-and-Hold do. Figures 4.5 and 3.4(b) also show that OLU is able to return more wealth than EG and Buy-and-Hold with reasonable transaction costs (0.1%, 0.25%, and 0.5%).

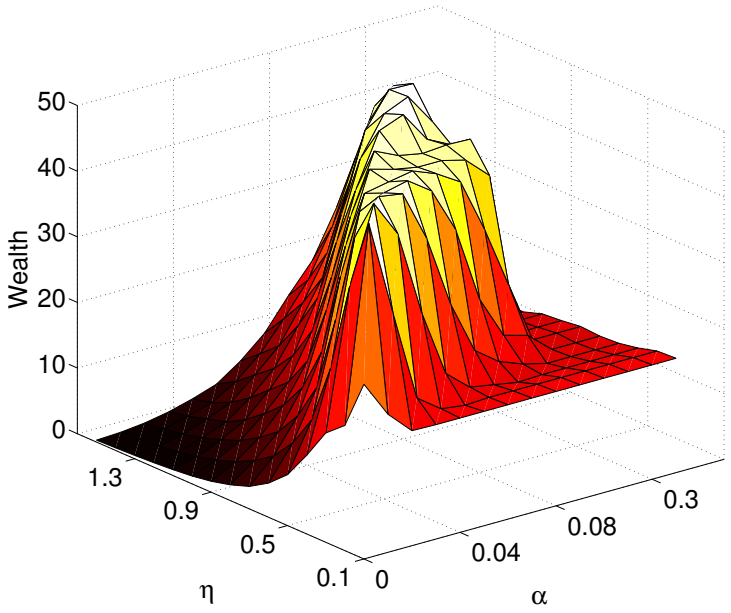
3.5.5 Parameter Sensitivity (η and α)

Figure 3.5 gives us more insight into how the transaction cost-adjusted wealth behaves as a function of $\frac{\eta}{\alpha} = \frac{1}{\gamma}$ for the two datasets. We can see that the cumulative wealth looks like a hill or ridge and that on either sides of the ridge the wealth is small. This particularly occurs when either η or α are too high or too low. Only when both η and α are in relative balance are we able to obtain significant cumulative wealth.

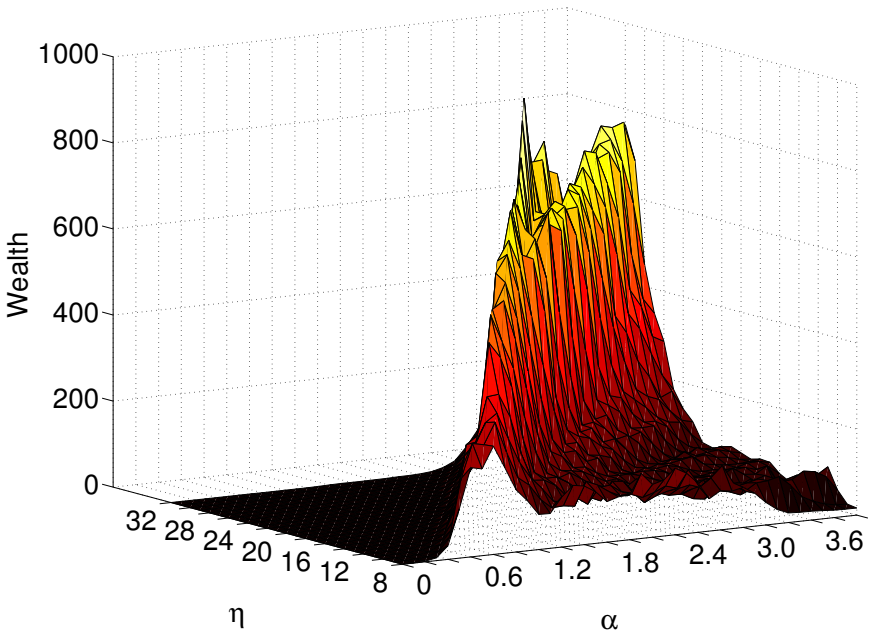
3.6 Conclusions

In this chapter, we have developed a framework and an online algorithm (OLU) to allow for lazy updates for the problem of portfolio selection with transaction costs. Our analysis shows that OLU is competitive with reasonable fixed strategies which have the power of hindsight. Our experimental results describe the behavior of such lazy updates and show that OLU is able to outperform EG and Buy-and-Hold even with reasonable transaction costs.

We comment briefly on the possible future directions of this framework. An extension of the current work could to explore the possibility of incorporating transactions costs and extending our analysis to other portfolio selection algorithms such as ONS [2] and meta-optimization algorithms [40] which have been shown to be theoretically grounded and empirically competitive. Finally, we think that our lazy updates framework has the potential to be generalized to help solve complex problems in other domains such as climate sciences, image processing, and social media analytics.



(a) S_T^γ for NYSE dataset.



(b) S_T^γ for S&P500 dataset.

Figure 3.5: Transaction cost-adjusted wealth: S_T^γ as a function of η and α for NYSE and S&P500 datasets.

Chapter 4

Online Portfolio Selection with Group Sparsity

4.1 Motivation

Investors often follow a top down approach which usually involves group selection followed by identifying the most profitable stocks within a group. One of the ways investors group stocks is by the type of business. The idea is to put companies in similar industries or sectors together. However, not all industries or sectors can yield profit and not all stocks in a particular industry or sector can be profitable. Moreover, different sectors might react differently during different economic conditions [86, 7]. For example, defensive sectors like utilities and consumer staples are robust to economic downturns whereas cyclical stocks which include technology, financials, health care, etc., tend to react quickly to fluctuations in the market. We are particularly interested in taking advantage and exploiting any underlying structure amongst the stocks for the problem of online portfolio selection.

Online portfolio selection has largely been a success story [36, 61, 32, 2, 18, 40, 82, 41, 85] over the last two decades. However, the existing work has not attempted to take advantage of the group structure that could exist amongst the stocks themselves.

The above application can be viewed as a particular instance of *structured sparsity*, which has lately been the focus of large amounts of research [125, 11, 67, 70, 69, 94].

The group lasso and its variants have been particularly popular and have been successfully employed in a number of applications [92, 124], such as gene finding, birthweight prediction and more.

We specifically focus on using a group sparsity inducing regularizer in an online learning framework where we also have to ensure that the updates to the solutions are sparse. Such lazy updates are motivated by our desire to handle proportional transaction costs in the portfolio selection problem. An investor could incur substantial transaction costs if his portfolio changes aggressively everyday [41, 42].

In this chapter, we first propose our general online lazy updates with group sparsity framework and go on to show that the online portfolio selection with sector information is a special case of this framework. Next, we introduce our OLU-GS algorithm which induces group sparsity and ensures that the updates are lazy. This results in solving a constrained non-smooth convex optimization problem at every iteration. We propose a novel alternating directions method of multipliers (ADMM) algorithm to solve this problem efficiently. In our analysis, which applies to any convex composite function with lazy updates, we show that our algorithm has $O(\sqrt{T})$ regret for general convex functions and $O(\log T)$ regret for strongly convex functions. Additionally, we prove regret bounds with respect to a shifting solution which has the benefit of hindsight.

We conduct extensive experiments on two real-world datasets and use the Industry Classification standard to group the stocks into sectors for 22 years of the benchmark NYSE dataset with 30 stocks and 8 sectors and 22 years of a S&P500 dataset with 243 stocks and 9 sectors. Our experiments show that our sparse group lazy portfolios can take advantage of the sector information to beat the market and are scalable with transaction costs. It shows an interesting group switching behavior and could be especially beneficial for individual investors who have expertise in select market sectors and are averse to changing their portfolio too often.

The chapter is arranged as follows. We introduce portfolio selection with group sparsity in section 4.3. We present our framework for online lazy updates with group sparsity in section 4.4. The analysis is presented in section 4.5 and section 4.6 discusses the experimental results in details. We conclude in section 4.7.

4.2 Online Portfolio Selection

We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over a span of T periods. We have already discussed the online portfolio selection setting in Chapters 2 and 3. For a sequence of price relatives $\mathbf{x}_{1:t-1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}$ up to day $(t-1)$, the sequential portfolio selection problem in day t is to determine a portfolio $\mathbf{p}_t$ based on past performance of the stocks. At the end of day t, $\mathbf{x}_t$ is revealed and the actual performance of $\mathbf{p}_t$ gets determined by $\mathbf{p}_t^T \mathbf{x}_t$. Over T periods, for a sequence of portfolios $\mathbf{p}_{1:T} = \{\mathbf{p}_1, \ldots, \mathbf{p}_T\}$, the multiplicative gain in wealth is $S_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \prod_{t=1}^{T} \left(\mathbf{p}_t^T \mathbf{x}_t\right)$ and the logarithmic gain in wealth is then,

$$LS_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T}) = \sum_{t=1}^{T} \log\left(\mathbf{p}_t^T \mathbf{x}_t\right). \tag{4.1}$$

Ideally, for a costless environment (no transaction costs) we would like to maximize $LS_T(\mathbf{p}_{1:T}, \mathbf{x}_{1:T})$ over $\mathbf{p}_{1:T}$. However, online portfolio selection cannot be posed as an optimization problem due to the temporal nature of the choices: $\mathbf{x}_t$ is not available when one has to decide on $\mathbf{p}_t$. Further, in a stock market, (statistical) assumptions regarding $\mathbf{x}_t$ can be difficult to make.

In a well defined technical sense, [36, 61, 2, 40, 41] have shown that their algorithms are guaranteed to perform competitively with certain families of adaptive portfolios even in an adversarial market without making any statistical assumptions regarding the movement of the stocks. However, none of the existing work in online portfolio selection has attempted to investigate or take advantage of the group structure (pre-specified or modeled) within the stocks in their algorithm setting.

4.3 Portfolio Selection with Group Sparsity

We focus on the problem of online portfolio selection with group sparsity where the groups are the pre-specified market sectors.The goal is to adaptively identify and invest in a few top performing sectors at any given period. In order to make our approach practical, we do not want the portfolios to change drastically everyday as an investor will have to pay transaction costs. So we encourage lazy updates to our portfolios along with group sparsity. We present a general formulation for our online lazy algorithm

with group sparsity and go on to show how the portfolio selection problem is a special case of our setting.

4.3.1 Problem Formulation

In an online lazy setting, the optimization proceeds in rounds where in round t the algorithm has to pick a solution, $\mathbf{p}_t \in \mathcal{P}$, from the feasible set such that it is sparse in the number of groups picked and close to the previous solution $\mathbf{p}_{t-1}$. Nature then reveals a convex loss function, f_t, and we observe its value $f_t(\mathbf{p}_t)$. Ideally, over T rounds we would like to minimize the quantity,

$$\sum_{t=1}^{T}\{f_t(\mathbf{p}_t) + \Omega_{\lambda_1}(\mathbf{p}_t)\} + \lambda_2 \sum_{t=1}^{T-1} ||\mathbf{p}_{t+1} - \mathbf{p}_t||_1 \tag{4.2}$$

In (4.2), the $\Omega(\cdot)$ penalty function can be any group norm which will ensure group sparsity. We adopt the "groupwise" ℓ_2-norm used in group lasso [124, 53], as our regularizer, i.e.,

$$\Omega_{\lambda_1}(\mathbf{p}) = \lambda_1 \Omega(\mathbf{p}) = \lambda_1 \sum_{g=1}^{\mathcal{G}} w_g \|\mathbf{p}_{|g}\| \tag{4.3}$$

We call $\mathcal{G}$ our set of groups and $\forall g \in \mathcal{G}, g \subseteq \{1, \cdots, n\}$. $\mathbf{p}_{|g}$ is the vector whose coordinates are equal to those of $\mathbf{p}$ for indices in the set g. $(w_g)_{g\in\mathcal{G}}$ denotes positive weights and $\|\cdot\|$ is the euclidean norm. To introduce group sparsity, it is also possible to impose other joint regularization on the weight, e.g. the $\ell_{1,\infty}$-norm [99]. We consider the case where the groups are disjoint, i.e. $\mathcal{G}$ is separable over $\{1, \cdots, n\}$, however our framework and algorithm can be extended to the overlapping group lasso case [69]. The ℓ_1 penalty term in (4.3) ensures that the updates to the solution $\mathbf{p}_t$ are lazy.

Absolute minimization of (4.2) is not reasonable because we do not know the sequence of f_t *a priori*. If the f_ts are known, (4.2) reduces to a batch optimization problem: a special case is the fused group lasso when f_t is quadratic [53, 115] or TV regularization [105]. Alternatively, over T iterations we intend to select a sequence of $\mathbf{p}_t$ such that the following *regret bound* is sub-linear in T,

$$R_T = \sum_{t=1}^{T} \phi_t(\mathbf{p}_t) - \min_{\mathbf{p}^* \in \mathcal{P}} \sum_{t=1}^{T} \phi_t(\mathbf{p}^*) \leq o(T) \tag{4.4}$$

where $\phi_t(\mathbf{p}) = f_t(\mathbf{p}) + \Omega_{\lambda_1}(\mathbf{p}) + \lambda_2||\mathbf{p} - \mathbf{p}_{t-1}||_1$ is non-smooth and $\mathbf{p}^*$ is the minimizer of $\sum_{t=1}^{T} \phi_t$ in hindsight. Note that while the $\mathbf{p}_t$s can change over time, $\mathbf{p}^*$ is fixed. That is, the minimizer, $\mathbf{p}^* = \text{argmin}_{\mathbf{p}} \sum_{t=1}^{T} \phi_t(\mathbf{p}) = \text{argmin}_{\mathbf{p}} \sum_{t=1}^{T} f_t(\mathbf{p}) + \Omega_{\lambda_1}(\mathbf{p})$, since it incurs zero ℓ_1 penalty in every iteration.

Additionally, we examine the case where the comparator class can also change over time. In particular, we consider the sequence $\{\mathbf{p}_1^*, \cdots, \mathbf{p}_T^*\}$ which has the power of hindsight. Then, over T iterations we ensure that the following *shifting* regret bound is sub-linear in T:

$$\sum_{t=1}^{T} \phi_t(\mathbf{p}_t) - \min_{\mathbf{p}_1^*, \cdots, \mathbf{p}_T^*} \sum_{t=1}^{T} \phi_t(\mathbf{p}_t^*) \leq o(T) \tag{4.5}$$

Online portfolio selection with group sparsity can now be viewed as a special case of the above setting where $f_t(\mathbf{p}) = -\log(\mathbf{p}^T \mathbf{x}_t))$ and the ℓ_1 penalty term on the difference of two consecutive portfolios measures the fraction of wealth traded. The parameters λ_1 controls how many groups are selected and λ_2 controls the amount that can be traded every day. Note that on setting $\lambda_1 = 0$ and $\lambda_2 = 0$, our formulation reduces to the costless case with no induced sparsity over the groups as seen in (4.1).

4.4 Online Lazy Updates with Group Sparsity

We now formally present our Online Lazy Updates with Group Sparsity (OLU-GS) algorithm. In the sequel, we show that using the solutions generated by OLU-GS, we can achieve sub-linear regret for the non-shifting (4.4) and shifting case (4.5). At the beginning of day $t + 1$, we find a new solution $\mathbf{p}_{t+1}$ by minimizing the following:

$$\begin{aligned} \mathbf{p}_{t+1} = \underset{\mathbf{p} \in \triangle_n}{\text{argmin}} \; & \langle \nabla f_t(\mathbf{p}_t), \mathbf{p} \rangle + \lambda_1 \Omega(\mathbf{p}) \\ & + \lambda_2 ||\mathbf{p} - \mathbf{p}_t||_1 + \frac{1}{2\eta} ||\mathbf{p} - \mathbf{p}_t||_2^2. \end{aligned} \tag{4.6}$$

where we have linearized f_t around $\mathbf{p}_t$. Our objective function in (4.6) is composite with smooth and non-smooth terms with the probability simplex as a constraint set (portfolio is a probability distribution). Although there is literature on solving composite functions [45, 123], composite functions with linear constraints have not been adequately investigated. We propose an Alternating Direction Method of Multipliers (ADMM) [22] based efficient primal-dual algorithm to solve (4.6). ADMM has been

applied in many large scale statistics and machine learning problems because of its computational benefits and fast convergence in practice [22]. We rewrite (4.6) in ADMM form by introducing auxiliary variables y and z as

$$\underset{\mathbf{p}\in\triangle_n,\mathbf{p}=\mathbf{y},\mathbf{p}-\mathbf{p}_t=\mathbf{z}}{\operatorname{argmin}} \langle\nabla\mathbf{f}_t(\mathbf{p}_t),\mathbf{p}\rangle+\lambda_1\Omega(\mathbf{y})+\lambda_2||\mathbf{z}||_1+\frac{1}{2\eta}||\mathbf{p}-\mathbf{p}_t||_2^2. \tag{4.7}$$

Next, using variable splitting, we write the *augmented lagrangian* for the problem as,

$$\begin{aligned} L(\mathbf{p},\mathbf{y},\mathbf{z},\mathbf{w},\mathbf{v}) &= \langle\nabla f_t(\mathbf{p}_t),\mathbf{p}\rangle + \lambda_1\Omega(\mathbf{y}) + \lambda_2||\mathbf{z}||_1 \\ &+ \frac{1}{2\eta}||\mathbf{p}-\mathbf{p}_t||_2^2+\frac{\beta}{2}||\mathbf{p}-\mathbf{y}+\mathbf{w}||_2^2+\frac{\beta}{2}||\mathbf{p}-\mathbf{p}_t-\mathbf{z}+\mathbf{v}||_2^2 \end{aligned} \tag{4.8}$$

where $\mathbf{w}$ and $\mathbf{v}$ are the scaled dual variables, and $\mathbf{p}\in\triangle_n$. Splitting the variables as we do in (4.7) has two advantages. Firstly, we will show that there is a closed form solution for each of our updates. Secondly, the updates for $\mathbf{y}$ and $\mathbf{z}$ can be done in parallel and the same is true for the scaled dual variables $\mathbf{w}$ and $\mathbf{v}$. ADMM consists of the following iterations for solving $\mathbf{p}_{t+1}$,

$$\begin{aligned} \mathbf{p}_{t+1}^{(k+1)} = &\underset{\mathbf{p}\in\triangle_n}{\operatorname{argmin}} \langle\nabla f_t(\mathbf{p}_t),\mathbf{p}\rangle+\frac{1}{2\eta}||\mathbf{p}-\mathbf{p}_t||_2^2 \\ &+ \frac{\beta}{2}||\mathbf{p}-\mathbf{y}^{(k)}+\mathbf{w}^{(k)}||_2^2+\frac{\beta}{2}||\mathbf{p}-\mathbf{p}_t-\mathbf{z}^{(k)}+\mathbf{v}^{(k)}||_2^2 \end{aligned} \tag{4.9}$$

$$\mathbf{y}^{(k+1)} = \underset{\mathbf{y}}{\operatorname{argmin}} \ \lambda_1\Omega(\mathbf{y}) \ + \ \frac{\beta}{2}||\mathbf{p}_{t+1}^{(k+1)}-\mathbf{y}+\mathbf{w}^{(k)}||_2^2 \tag{4.10}$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \lambda_2||\mathbf{z}||_1+\frac{\beta}{2}||\mathbf{p}_{t+1}^{(k+1)}-\mathbf{p}_t-\mathbf{z}+\mathbf{v}^{(k)}||_2^2 \tag{4.11}$$

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} \ + \ (\mathbf{p}_{t+1}^{(k+1)}-\mathbf{y}^{(k+1)}) \tag{4.12}$$

$$\mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} \ + \ (\mathbf{p}_{t+1}^{(k+1)}-\mathbf{p}_t-\mathbf{z}^{(k+1)}) \tag{4.13}$$

p-update: We take the derivative of (4.9) *w.r.t.* $\mathbf{p}$ and set it to zero to get a closed form update of $\mathbf{p}$ as $\nabla f_t(\mathbf{p}_t)+\frac{1}{\eta}(\mathbf{p}-\mathbf{p}_t)+\beta(\mathbf{p}-\mathbf{y}^{(k)}+\mathbf{w}^{(k)})+\beta(\mathbf{p}-\mathbf{p}_t-\mathbf{z}^{(k)}+\mathbf{v}^{(k)})=0$. Rearranging this and setting $\hat{a}=\frac{1+\eta\beta}{1+2\eta\beta}$ and $\hat{b}=\frac{\eta\beta}{1+2\eta\beta}$ and $\hat{c}=\frac{\eta}{1+2\eta\beta}$, we get (4.14). $\prod_{\mathbf{p}\in\triangle_n}$ is the projection operator which is carried out as in [46].

Algorithm 3 OLU-GS Algorithm with ADMM

1: Input $\mathbf{p}_t, \mathbf{x}_t, \nabla f_t(\mathbf{p}_t), \mathcal{G}, \lambda_1, \lambda_2, \eta, \beta$
2: Initialize $\mathbf{p}, \mathbf{y}, \mathbf{z}, \mathbf{w}, \mathbf{v} \in 0^n, k = 0$
3: Set $\hat{a} = \frac{1+\eta\beta}{1+2\eta\beta}$ and $\hat{b} = \frac{\eta\beta}{1+2\eta\beta}$ and $\hat{c} = \frac{\eta}{1+2\eta\beta}$
4: ADMM iterations

$$\mathbf{p}_{t+1}^{k+1} = \prod_{\mathbf{p}\in\triangle_n} \{\hat{a}\mathbf{p}_t - \hat{c}\nabla f_t(\mathbf{p}_t) + \hat{b}(\mathbf{y}^{(k)} + \mathbf{z}^{(k)} - \mathbf{w}^{(k)} - \mathbf{v}^{(k)})\} \tag{4.14}$$

$$\mathbf{y}_{|g}^{(k+1)} = S_{\lambda_1/\beta}(\mathbf{p}_{|g}^{(k+1)} - \mathbf{p}_{t|g} + \mathbf{w}_{|g}^{(k)}), \forall g \in \mathcal{G} \tag{4.15}$$

$$\mathbf{z}^{(k+1)} = S_{\lambda_2/\beta}(\mathbf{p}_{t+1}^{k+1} - \mathbf{p}_t + \mathbf{v}^k) \tag{4.16}$$

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + (\mathbf{p}^{(k+1)} - \mathbf{y}^{(k+1)} + \mathbf{w}^{(k)}) \tag{4.17}$$

$$\mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + (\mathbf{p}^{(k+1)} - \mathbf{p}_t - \mathbf{z}^{(k+1)} + \mathbf{v}^{(k)}) \ . \tag{4.18}$$

where $\prod_{\triangle_n}$ is the projection to the simplex and S_ρ is the shrinkage operator.
5: Continue until **Stopping Criteria** is satisfied

y-update: We can rewrite (4.10) as

$$\mathbf{y}^{(k+1)} = \underset{\mathbf{y}}{\operatorname{argmin}} \ \frac{1}{2}||\mathbf{p}^{(k+1)} + \mathbf{w}^{(k)} - \mathbf{y}||_2^2 + \frac{\lambda_1}{\beta}\Omega(\mathbf{y}) \tag{4.19}$$

When $\Omega(\cdot)$ is a group lasso penalty with l_2-norm, with $\mathcal{G}$ being a partition of $\{1, \cdots, n\}$, (4.19) is *separable* in every group, and the solution is a generalization of the soft thresholding operator to groups of variables [70]:

$$\forall g \in \mathcal{G}, \mathbf{y}_{|g} = \begin{cases} 0 & \text{if } ||\mathbf{q}_{|g}||_2 \leqslant \tilde{\lambda} \\ \frac{||\mathbf{q}_{|g}||_2 - \tilde{\lambda}}{||\mathbf{q}_{|g}||_2}\mathbf{r}_{|g} & \text{otherwise} \end{cases} \tag{4.20}$$

where $\mathbf{q} = \mathbf{p}^{(k+1)} + \mathbf{w}^{(k)}$ and $\mathbf{y}_{|g}$ is a vector of size n whose coordinates are equal to those of $\mathbf{y}$ for indices in the set g. We obtain a closed form solution for $\mathbf{z}^{k+1}$ by using the soft-thresholding operator $S_\rho(a)$ [22]. The updates for $\mathbf{w}^{(k+1)}$ (4.17) and $\mathbf{v}^{(k+1)}$ (4.18) are already in closed form. We iterate over the updates until convergence according to the stopping criteria in [22]. Algorithm 3 summarizes the ADMM updates for OLU-GS.

Algorithm 4 outlines our portfolio selection algorithm with group sparsity and computes the transaction cost-adjusted wealth S_T^γ, where γ is a proportional transaction cost [41]. Here, $f_t(\mathbf{p}) = -\log(\mathbf{p}^T\mathbf{x}_t)$ and $\nabla f_t(\mathbf{p}) = -\frac{\mathbf{x}_t}{\mathbf{p}^T\mathbf{x}_t}$. $\mathcal{G}$ is a set of disjoint groups over the n stocks that is provided to us.

Algorithm 4 Portfolio Selection with Group Sparsity

Input $\mathcal{G}, \lambda_1, \lambda_2, \eta, \beta$; Transaction cost γ
Initialize $p_{1,g} = \frac{1}{|\mathcal{G}|}, g = 1, \ldots, |\mathcal{G}|; p_0 = p_1; S_0^\gamma = 1$
For $t = 1, \ldots, T$
    Receive $\mathbf{x}_t$, the vector of price relatives
    Compute cumulative wealth: $S_t^\gamma = S_{t-1}^\gamma \times (\mathbf{p}_t^T \mathbf{x}_t) - \gamma \times S_{t-1}^\gamma \times ||\mathbf{p}_t - \mathbf{p}_{t-1}||_1$
    Update portfolio:
    $\mathbf{p}_{t+1} = \text{OLU-GS}(\mathbf{p}_t, \mathbf{x}_t, -\frac{\mathbf{x}_t}{\mathbf{p}^T \mathbf{x}_t}, \mathcal{G}, \lambda_1, \lambda_2, \eta, \beta)$
end for

4.5 Analysis

In this section we theoretically analyze the performance (regret) of our OLU-GS algorithm with two classes of batch solutions. The first class is the best fixed solution which can see the entire sequence of data in hindsight. The second class is the best sequence of solutions in hindsght: i.e., the comparator class can also change over time. Our aim is to show that we can achieve sub linear regret in both the cases. Our analysis, like our framework is general and applies to any convex *composite objective* with lazy updates. The composite comprises of a convex function f_t which changes over time and a fixed regularizer Ω. We consider 2 sub cases for our analysis with the fixed solution in hindsight: (1) when we consider general convex function f_t and (2) when we assume a little more about the the convex function f_t: in particular strong convexity. We will show that with the strong convexity assumption, we achieve sharper regret bound ($O(\log T)$) compared to general convex functions ($O(\sqrt{T})$). chapter

This chapter, we consider updates of the following form:

$$\mathbf{p}_{t+1} = \underset{\mathbf{p} \in \mathcal{P}}{\operatorname{argmin}} \left\{ \eta \langle \nabla f_t(\mathbf{p}_t), \mathbf{p} \rangle + \eta r(\mathbf{p}) + \eta \lambda_2 \|\mathbf{p} - \mathbf{p}_t\|_1 + d_\psi(\mathbf{p}, \mathbf{p}_t) \right\}, \tag{4.21}$$

where the $\lambda_1 \Omega(\mathbf{p}) = r(\mathbf{p})$, where the constant λ_1 has been absorbed in the function $\mathbf{r}$. For our formulation, the Bregman divergence $d_\psi(\mathbf{p}, \mathbf{p}_t) = \frac{1}{2}||\mathbf{p} - \mathbf{p}_t||_2^2$.

4.5.0.1 General Convex Functions

We assume that f_t are general convex functions with bounded (sub)gradients, i.e., for any $\hat{g} \in \partial f_t(\mathbf{p})$ we have $\|\hat{g}\| \leq G$.

Lemma 3. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* $d_\psi(\cdot,\cdot)$ *is* α*-strongly convex with respect to norm* $||\cdot||$*, i.e.* $d_\psi(\mathbf{p},\hat{\mathbf{p}}) \geq \frac{\alpha}{2}\|\mathbf{p}-\hat{\mathbf{p}}\|_2^2$ *and* $||\mathbf{p}-\hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p},\hat{\mathbf{p}} \in \mathcal{P}$*. Then, for any* $\mathbf{p}^* \in \mathcal{P}$*,*

$$\eta[f_t(\mathbf{p}_t) + r(\mathbf{p}_{t+1}) + \lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1] - \eta[f_t(\mathbf{p}^*) + r(\mathbf{p}^*)] \\ \leq d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\lambda_2 L + \frac{\eta^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2. \tag{4.22}$$

The proof of the lemma can be found in the appendix.

Theorem 4. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* f_t *be a Lipschitz continuous function for which* $\|\nabla f_t(\mathbf{p}_t)\|_2^2 \leq G$*. Then, by choosing* $\eta \propto \frac{1}{\sqrt{T}}$ *and* $\lambda_2 \propto \frac{1}{\sqrt{T}}$*, we have*

$$\sum_{t=1}^{T}[f_t(\mathbf{p}_t) + r(\mathbf{p}_t) + \lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - f(\mathbf{p}^*) - r(\mathbf{p}^*)] \leq O(\sqrt{T}) \tag{4.23}$$

Proof. By Lemma 3, we have

$$\eta\sum_{t=1}^{T}[f_t(\mathbf{p}_t) + r(\mathbf{p}_{t+1}) + \lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - f_t(\mathbf{p}^*) - r(\mathbf{p})] \\ \leq \sum_{t=1}^{T} d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\lambda_2 LT + \frac{\eta^2 G^2 T}{2\alpha} \\ = \sum_{t=1}^{T} d_\psi(\mathbf{p}^*,\mathbf{p}_1) - d_\psi(\mathbf{p}^*,\mathbf{p}_{T+1}) + \eta\lambda_2 LT + \frac{\eta^2 G^2 T}{2\alpha}$$

Noting, that Bregman divergences are always non-negative and adding $\eta(\mathbf{p}_1)$ on both sides and dropping the $r(\mathbf{p}_{T+1})$ and $\lambda_2\|\mathbf{p}_{T+1}-\mathbf{p}_T\|_1$ terms we have,

$$\sum_{t=1}^{T}[f_t(\mathbf{p}_t) + r(\mathbf{p}_t)] + \lambda_2\sum_{t=1}^{T-1}[\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1] - \sum_{t=1}^{T}[f_t(\mathbf{p}^*) + r(\mathbf{p}^*)] \\ \leq \frac{1}{\eta} d_\psi(\mathbf{p}^*,\mathbf{p}_1) + \lambda_2 LT + \frac{\eta G^2 T}{2\alpha} + r(\mathbf{p}_1) \\ \leq \sqrt{T} d_\psi(\mathbf{p}^*,\mathbf{p}_1) + L\sqrt{T} + \frac{G^2\sqrt{T}}{2\alpha}$$

We get the last inequality by substituting $\eta \propto \frac{1}{\sqrt{T}}$ and $\lambda_1 \propto \frac{1}{\sqrt{T}}$ and setting $\mathbf{r}(\mathbf{p}_1) = 0$ □ □

4.5.0.2 Strongly Convex Functions

We assume that f_t are all β-strongly convex functions so that for any $(\mathbf{p}, \mathbf{p}_t)$

$$f_t(\mathbf{p}) \geq f_t(\mathbf{p}_t) + \langle \mathbf{p} - \mathbf{p}_t, \nabla f_t(\mathbf{p}_t) \rangle + \frac{\beta}{2} \|\mathbf{p} - \mathbf{p}_t\|^2 \ . \tag{4.24}$$

Lemma 4. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let $d_\psi(\cdot, \cdot)$ is α-strongly convex with respect to norm $||\cdot||$, i.e. $d_\psi(\mathbf{p}, \hat{\mathbf{p}}) \geq \frac{\alpha}{2}\|\mathbf{p} - \hat{\mathbf{p}}\|_2^2$ and $||\mathbf{p} - \hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p}, \hat{\mathbf{p}} \in \mathcal{P}$. Assuming f_t are all β-strongly convex, for any $\lambda_2 < \frac{\beta}{4}$ and any $\mathbf{p}^* \in \mathcal{P}$, we have*

$$\begin{aligned} &\eta_t[f_t(\mathbf{p}_t) + r(\mathbf{p}_{t+1}) - f_t(\mathbf{p}^*) - r(\mathbf{p}^*) + \lambda_2 \|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1] \\ &\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \frac{\eta_t^2}{2\alpha} ||\nabla f_t(\mathbf{p}_t)||^2 - \left(\frac{\beta}{2} - 2\lambda_2\right) \|\mathbf{p}^* - \mathbf{p}_t\|^2 \ . \end{aligned} \tag{4.25}$$

The proof of this lemma is in the appendix.

Theorem 5. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let f_t be all β-strongly convex. Then, for any $\lambda_2 < \beta/4$, choosing $\eta_t = \frac{2}{\gamma t}$, where $\gamma \in (0, \frac{\beta}{4} - \lambda_2]$, we have*

$$\sum_{t=1}^{T} [f_t(\mathbf{p}_t) + r(\mathbf{p}_t) + \lambda_2 \|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - f(\mathbf{p}^*) - r(\mathbf{p}^*)] \leq O(\log(T)) \tag{4.26}$$

Proof. By Lemma 4, we have

$$\begin{aligned} &\sum_{t=1}^{T} [f_t(\mathbf{p}_t) + r(\mathbf{p}_t) + \lambda_2 \|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - f_t(\mathbf{p}^*) - r(\mathbf{p}^*)] \\ &\leq \sum_{t=1}^{T} \frac{1}{\eta_t} d_\psi(\mathbf{p}^*, \mathbf{p}_t) - \frac{1}{\eta_t} d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) - \left(\frac{\beta}{2} - 2\lambda_2\right) \|\mathbf{p}^* - \mathbf{p}_t\|^2 + \frac{\eta_t}{2\alpha} ||\nabla f_t(\mathbf{p}_t)||^2 + r(\mathbf{p}_1) \\ &\leq d_\psi(\mathbf{p}^*, \mathbf{p}_1) + \sum_{t=1}^{T-1} \left[d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\right) - \left(\frac{\beta}{2} - 2\lambda_2\right) \|\mathbf{p}^* - \mathbf{p}_t\|^2 \right] \\ &\qquad\qquad + \frac{G^2}{2\alpha} \sum_{t=1}^{T} \eta_t + r(\mathbf{p}_1) \\ &\leq d_\psi(\mathbf{p}^*, \mathbf{p}_1) + r(\mathbf{p}_1) - \sum_{t=1}^{T-1} 2 \left(\frac{\beta}{4} - \lambda_2 - \gamma\right) \|\mathbf{p}^* - \mathbf{p}_{t+1}\|^2 + \frac{cG^2}{2\alpha} \log T \ , \end{aligned}$$

where we have assumed $d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) = \frac{1}{2}\|\mathbf{p}^* - \mathbf{p}_{t+1}\|^2$ (for simplicity) and used $\sum_{t=1}^{T} \frac{1}{t} = c \log T$. Now, since $(\frac{\beta}{4} - \lambda_2 - \gamma) \geq 0$, we have

$$\sum_{t=1}^{T}[f_t(\mathbf{p}_t) + r(\mathbf{p}_t) + \lambda_2\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - f_t(\mathbf{p}^*) - r(\mathbf{p}^*)] \leq d_\psi(\mathbf{p}^*, \mathbf{p}_1) + r(\mathbf{p}_1) + \frac{cG^2}{2\alpha}\log T$$
$$= O(\log T) \ ,$$

which completes the proof. □ □

4.5.1 Shifting Bounds for OLU-GS Algorithm

Theorem 6. *Let* $\{\mathbf{p}_1^*, \cdots, \mathbf{p}_T^*\}$ *be the best sequence obtained by minimizing (4.2). For,* $\frac{1}{a} + \frac{1}{b} = 1$ *we have*

$$\begin{aligned}\sum_{t=1}^{T}[f_t(\mathbf{p}_t) + r(\mathbf{p}_t) - f_t(\mathbf{p}_t^*) - r(\mathbf{p}_t^*)] + \lambda_2 \sum_{t=1}^{T-1}[\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 \\ - \|\mathbf{p}_{t+1}^* - \mathbf{p}_t^*\|_1] \leq O(\sqrt{T}) + ||\bigtriangledown\phi(\mathbf{p}_{t+1})||_a \sum_{t=1}^{T-1}||\mathbf{p}_t^* - \mathbf{p}_{t+1}^*||_b\end{aligned} \tag{4.27}$$

The proof is similar to the proof of Theorem 3 and we only present a sketch of the proof below.

Proof Sketch: From, Lemma 1 we have

$$\begin{aligned}\eta[f_t(\mathbf{p}_t) + r(\mathbf{p}_{t+1}) + \lambda_2\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - f_t(\mathbf{p}_t^*) - r(\mathbf{p}_t^*)] \\ \leq d_\psi(\mathbf{p}_t^*, \mathbf{p}_t) - d_\psi(\mathbf{p}_t^*, \mathbf{p}_{t+1}) + \eta\lambda_2 L + \frac{\eta^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2\end{aligned} \tag{4.28}$$

This does not telescope, so we add

$$\begin{aligned}\phi(\mathbf{p}_t^*) - \phi(\mathbf{p}_{t+1}^*) - ||\bigtriangledown\phi(\mathbf{p}_{t+1})||_a||\mathbf{p}_t^* - \mathbf{p}_{t+1}^*||_b \\ \leq d_\psi(\mathbf{p}_t^*, \mathbf{p}_{t+1}) - d_\psi(\mathbf{p}_{t+1}^*, \mathbf{p}_{t+1})\end{aligned} \tag{4.29}$$

We use Holder's inequality for $(\mathbf{p}_{t+1}^* - \mathbf{p}_t^*)^T\nabla\phi(\mathbf{p}_{t+1}) \geq -||\bigtriangledown\phi(\mathbf{p}_{t+1})||_a||\mathbf{p}_t^* - \mathbf{p}_{t+1}^*||_b$.

Adding (4.28), (4.29) and then adding $\eta\lambda_2\|\mathbf{p}_{t+1}^* - \mathbf{p}_t^*\|_1$ on both sides and summing

Sector	Example Companies
Consumer Discretionary	Nike Inc., Target Corp.
Consumer Staples	ConAgra Foods Inc., Altria Group Inc.
Energy	Chevron Corp., Halliburton Co.
Financials	Equifax Inc., H&R Block Inc.
Health Care	Johnson & Johnson, Pfizer Inc.
Industrials	General Electric Co., 3M Co.
Information Tech	Apple Inc., Dell Inc.
Materials	Dow Chemical Co., Ecolab Inc.
Utilities	AGL Resources, Xcel Energy Inc.

Table 4.1: Overview of GISC sectors used in our dataset.

over all t, we have

$$\begin{aligned}
&\sum_{t=1}^{T}[f_t(\mathbf{p}_t)+r(\mathbf{p}_t)-f_t(\mathbf{p}_t^*)-r(\mathbf{p}_t^*)] \\
&+\lambda_2\sum_{t=1}^{T-1}[\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\|\mathbf{p}_{t+1}^*-\mathbf{p}_t^*\|_1 \\
&\leq\frac{1}{\eta}\{d_\psi(\mathbf{p}_1^*,\mathbf{p}_1)-d_\psi(\mathbf{p}_T^*,\mathbf{p}_{T+1})+\phi(\mathbf{p}_T^*)-\phi(\mathbf{p}_1)^* \\
&+||\bigtriangledown\phi(\mathbf{p}_{t+1})||_a\sum_{t=1}^{T-1}||\mathbf{p}_t^*-\mathbf{p}_{t+1}^*||_b+2\eta\lambda_2LT+\frac{\eta G^2T}{2\alpha}
\end{aligned} \tag{4.30}$$

where we bound $\|\mathbf{p}_{t+1}^*-\mathbf{p}_t^*\|_1\leq L$. □

shifting that occurs for the best sequence of $\mathbf{p}_t^*$s.

4.6 Experiments and Results

Dataset: The experiments were conducted on data taken from the New York Stock Exchange (NYSE) and Standard & Poor's 500 (S&P 500) stock market index. The NYSE dataset [61, 2, 18, 36] consists of 36 stocks with data accumulated over a period of 22 years from July 3, 1962 to December 31, 1984. The dataset captures the bear market that lasted between January 1973 and December 1974. The S&P500 dataset consists of 258 stocks with data accumulated over a period of 22 years from 1991 to 2012. The dataset captures the bull and bear markets of recent times such as the dot-com bubble which occurred between 1997-2000, the following bubble burst starting in

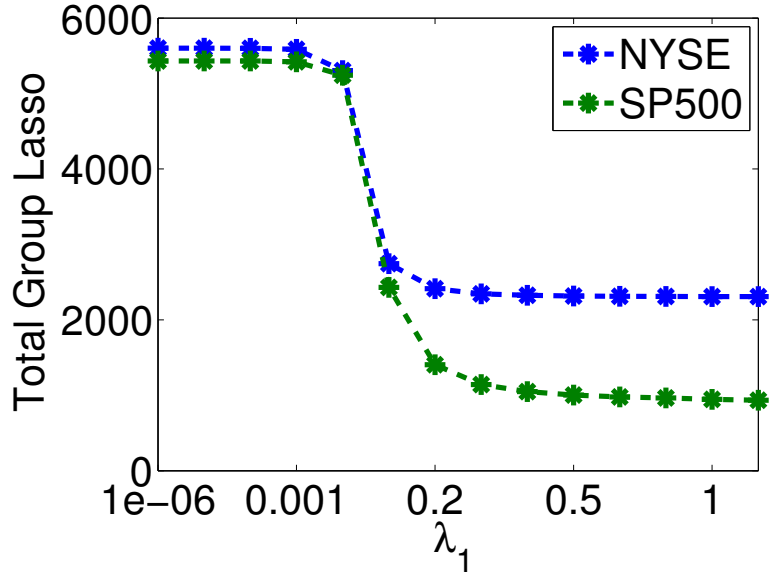
March 2000 and continuing through 2002, and the recent financial and housing bubble burst between 2007-2009.

We used the Global Industry Classification Standard to group the stocks in the datasets into their designated sectors: Consumer Discretionary, Consumer Staples, Energy, Financials, Health Care, Industrials, Information Technology, Materials, and Utilities. This resulted in 8 sectors and 30 stocks being represented in the NYSE dataset and 9 sectors and 243 stocks in the S&P500 dataset. Table 4.1 shows the sectors represented in the two dataset, and a couple representative companies from each sector.

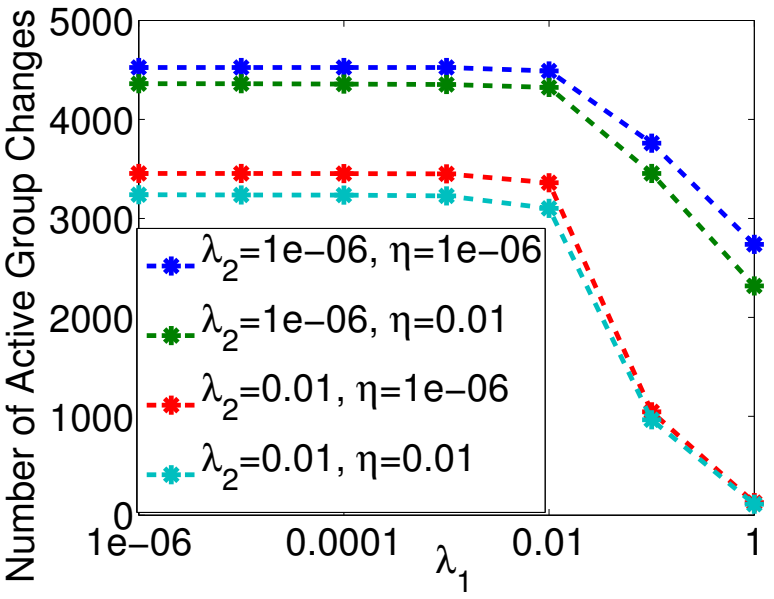
Methodology and Parameter Setting: In all experiments we started with \$1 as our initial investment and an initial portfolio uniformly distributed over the groups to avoid group bias. We use OLU-GS to obtain our portfolios sequentially and compute the transaction cost-adjusted wealth for each day. The parameters consist of $\boldsymbol{\lambda}_1$: weight on group sparsity norm, $\boldsymbol{\lambda}_2$: lazy updates weight, $\boldsymbol{\eta}$: weight on the ℓ_2 norm, and $\boldsymbol{\beta}$: the parameter for the augmentation term. For all our experiments, we set $\beta = 2$ which we found to give reasonable accuracy and use group lasso as the group sparsity norm.

Since the two datasets are very different in nature (stock composition and duration), we experimented extensively with a large range of λ_1, λ_2, and η values from 1e$-$9 to 1 to observe their effect on group sparsity and lazy updates to our portfolio. Moreover, we chose a reasonable range of γ values (in percentage) to compute the proportional transaction costs incurred due to the portfolio update every day. The range of γ values we experimented with were between 0% and 2%. We have illustrated some of our results with representative plots from either the NYSE or S&P500 dataset.

We use the wealth obtained (without transaction costs) from the EG algorithm with experimentally tuned parameters, a Buy-and-Hold strategy, and the best single stock as benchmarks for our experiments with initial investments of \$1. EG has been shown to outperform a uniform constantly rebalanced portfolio [61, 40]. For the Buy-and-Hold case we start with a uniformly distributed portfolio and do a hold on the positions thereafter (i.e. no trades). For the best single stock case we observe how the market performs and select the stock that has accumulated the most wealth at the end of the period. Note, in a real world situation this strategy is infeasible since it is not possible to know the best stock *a priori*.



(a) Total Group Lasso.



(b) Number of Active Group Changes.

Figure 4.1: As λ_1 increases the (a) total group lasso value and (b) number of active group changes decrease.

4.6.1 Effect of λ_1 for Group Sparsity ($\Omega(\mathbf{p})$)

The regularization parameter λ_1 for the group lasso term ($\Omega(\mathbf{p})$) is varied from [1e$-$9, 1] to obtain different levels of group sparsity. The value of λ_1 has a strong effect on (a) the total group lasso penalty value, (b) the number of active groups, and (c) which groups are active.

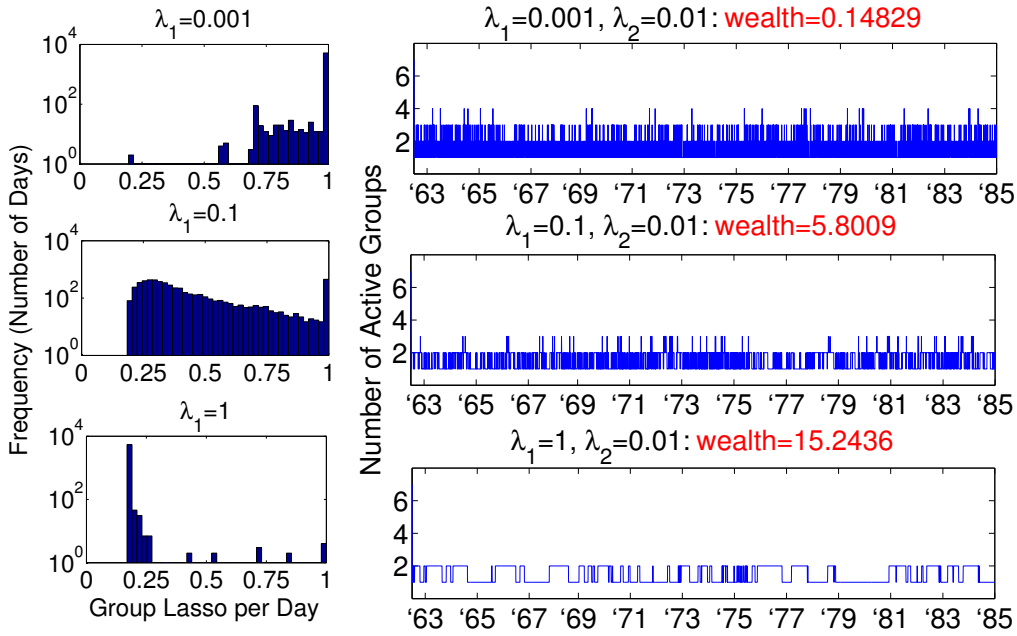
(a). Total Group Lasso penalty: Figure 4.1(a) plots the value of the total group lasso penalty ($\sum_{t=1}^{T} \Omega(\mathbf{p}_t)$) as we increase λ_1, keeping λ_2 and η fixed. For both the NYSE and S&P500 datasets, we observe that $\sum_{t=1}^{T} \Omega(\mathbf{p}_t)$ decreases as we increase λ_1, which is in conformance with our objective. Since the two datasets are different in terms of the total number of stocks and the number of stocks composing each sector, Figure 4.1(a) specifically illustrates how to choose λ_1 to attain a desired level of sparsity for each of the datasets. Figure 4.2(a) and Figure 1* (in the supplement)[1] plot histograms of the total per day group lasso penalty with increasing λ_1 values for the S&P500 and NYSE dataset respectively. It is fairly evident that there is a decrease in the number of days with high group lasso penalty as λ_1 increases.

(b). Active Groups: We compute the *active groups* each day by selecting the groups in which the majority (80%) of the wealth is invested. Figures 4.2(b) and 2* plot the number of active groups per day for the NYSE and S&P500 datasets respectively. For NYSE with $\lambda_1 =$ 1e$-$3, OLU-GS picks up to 4 groups on a particular day. For a higher value of $\lambda_1 = 1$ a maximum of 2 groups are selected to invest in. In particular, the two sectors picked are Basic Materials and Consumer Discretionary. Additionally, we can see this effect of λ_1 on group sparsity at the individual sector weights in Figure 3*.

(c). Active Groups Changes: Additionally, Figure 4.1(b) plots the total number of times the *active groups* change for the NYSE dataset (over 22 years) as λ_1 increases. We consider an active group change as anytime the group composition changes. The individual line plots indicate different values of λ_2 and η. For λ_1 between 1e$-$6 to 1e$-$2: with low values for λ_2, the total number of changes in the active groups are quite high but for a higher value of $\lambda_2 =$ 1e$-$2 we see a decrease in the number of active group changes illustrating the portfolio laziness. With larger values of $\lambda_1 \geq 1e-2$ we see a dramatic drop in the number of active group changes and high λ_2 values only

[1] Figure numbers appended with * are in the supplement.

reemphasizes this behavior.



(a) Group Lasso per day histogram.

(b) Number of Active Groups.

Figure 4.2: As λ_1 increases the number of days with high group lasso value and the number of active groups decrease.

4.6.2 Wealth and Group Sparsity

To evaluate the practical application of our proposed algorithm, we now analyze its performance when calculating the transaction cost-adjusted cumulative wealth. Figure 4.3 shows how the choice of different λ_1 values affect the transaction cost-adjusted cumulative wealth for the NYSE dataset (for a fixed λ_2 and η value). Figure 4.4 demonstrates that there exists a combination of λ_1 and λ_2 values which make an optimal choice between group sparsity and lazy updates.

EG, Buy-and-Hold, Best Single Stock: We compared the total wealth without transaction costs of OLU-GS with that of EG, a Buy-and-Hold strategy, and the best performing single stock. These strategies are plotted as horizontal lines and we can see that for the NYSE dataset EG returns $20.89, Buy-and-Hold returns $20.88, and

the best single stock, Phillip Morris (MO), returns \$54.14. In comparison, OLU-GS returns \$71.18 without transaction costs. OLU-GS returns over 3x as much wealth for the NYSE dataset as EG or Buy-and-Hold do and about \$15 more than the best stock. Figure 4.5 also shows that OLU-GS is able to return more wealth than EG and Buy-and-Hold with reasonable transaction costs (0.001%, 0.005%, and 0.01%).

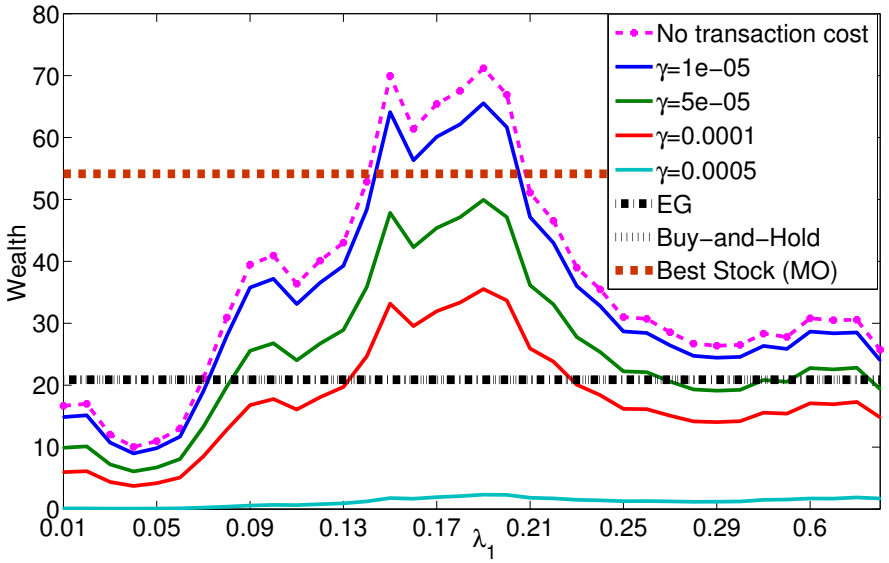


Figure 4.3: NYSE: Transaction cost-adjusted wealth with OLU-GS.

4.6.3 OLU-GS: Switching Sectors

We desire that OLU-GS to be able to identify the best sectors automatically. We illustrate the strength of OLU-GS in selecting the best sectors with two examples. A recurring trend that we observe from our experiments with both the NYSE and S&P500 datasets is that OLU-GS selects stocks in Consumer Staples during the bear markets. Figure 4.5(a) clearly shows that OLU-GS selects and invests in this defensive sector during the historical bear markets of 1969-1971 and 1975-1977. Another example of a defensive or non-cyclic sector is Utilities. Figure 4.5(b) shows that the weight on the Utilities sector sees a considerable increase during the dot-com crash. This is interesting because unlike other areas of the economy, even during bear markets, the demand for Consumer Staples and Utilities do not slow down. These sectors consist of stocks which are defensive in nature and usually outperform the S&P500 Index during

bearish markets and under-perform during bullish markets.

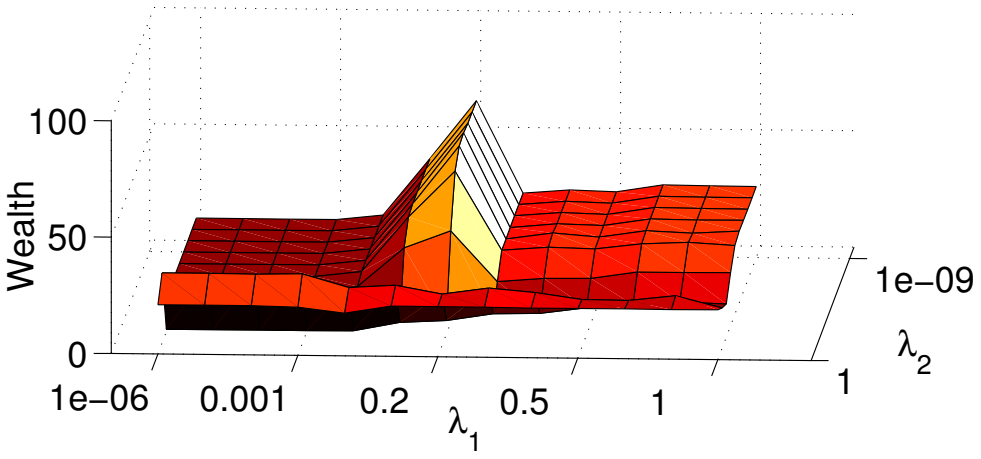
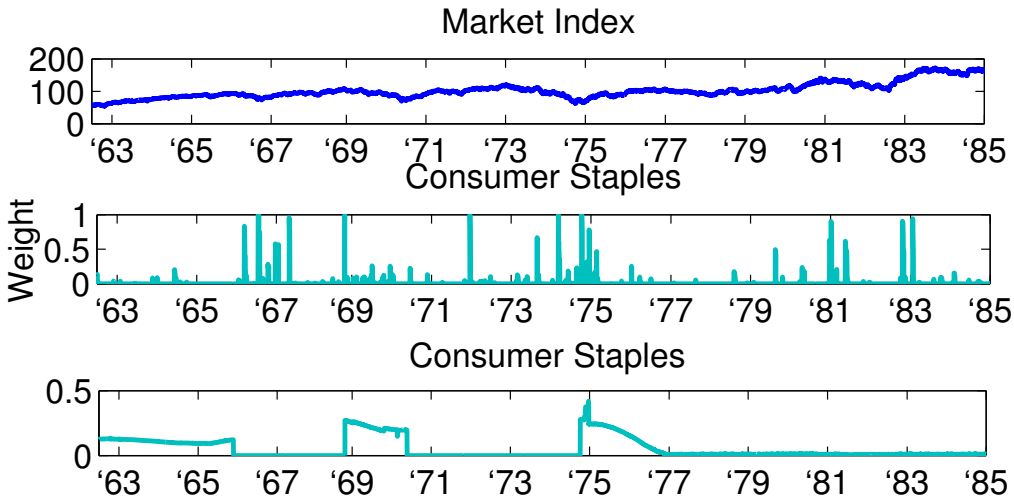


Figure 4.4: NYSE: Wealth as a function of λ_1 and λ_2.

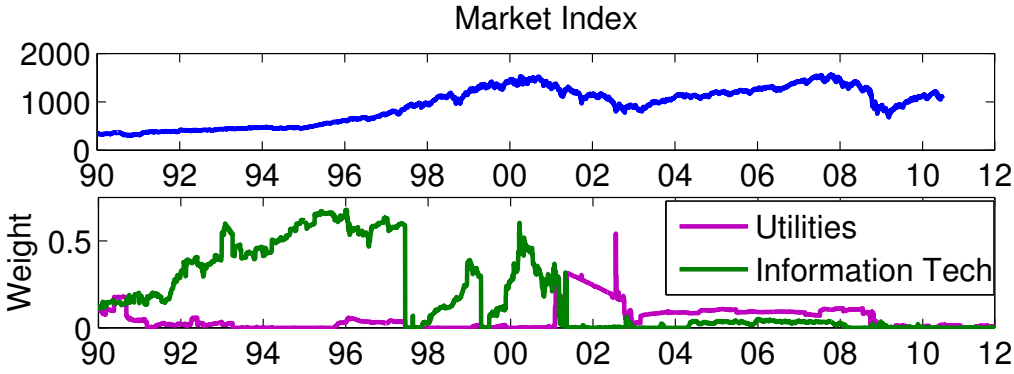
Sectors like Information Technology and Financials comprise of cyclical stocks which are sensitive to market movements and can take advantage of the bullish markets. In Figure 4.5(b), we see that Information Technology sector is picked up during the bullish markets which preceded the dot-com bubble.

4.7 Conclusions

In this chapter, we have developed a general lazy online learning with group sparsity framework and an online learning algorithm (OLU-GS) and show how it can be applied to the problem of online portfolio selection with sector information and transaction costs. Our analysis shows that OLU-GS is competitive with reasonable fixed and shifting strategies which have the power of hindsight. Our experimental results illustrate the behavior of group sparsity and lazy updates and show that OLU-GS is able to outperform baseline strategies with reasonable transaction costs. Finally, we demonstrate that OLU-GS is able to select the best performing sectors during different economic conditions. A possible future extension could be to explore the possibility of learning the group structure from the data itself.



(a) NYSE: Consumer staples picked consistently during bear market with varying parameters.



(b) S&P500: Utilities/Info Tech selected during bear/bull markets.

Figure 4.5: OLU-GS: Picking nonyclic sector during bear market and cyclic during bull market.

Chapter 5

Meta Optimization and its Application to Portfolio Selection

5.1 Motivation

Several data mining algorithms use iterative update methods for learning predictive models. Typically, there are several choices for iterative update methods including gradient based or Newton step based optimization routines, stochastic gradient descent algorithms, domain specific methods, evolutionary and genetic algorithms, or plain heuristics. It is not easy to determine upfront which method will converge fast or perform the best.

While multiple iterative update methods can be run in an embarrassingly parallel manner, it is unclear if iterates from multiple algorithms for the same problem can be meaningfully combined to guarantee good optimization performance. Ideally, one would like the combined iterates to outperform the best algorithm in the pool, noting that the best algorithm may be different for different problem settings and domains. Such a desideratum is related to ensemble methods for prediction problems, where one expects the ensemble prediction to outperform the single best predictor in the pool [52, 25, 26]. In this chapter, we investigate a related question in the contest of iterative optimization: Can iterates from multiple iterative update algorithms for the same problem be combined in a way so as to outperform the single best algorithm in the pool in terms of optimization performance? Related questions have been investigated

in certain other contexts, including online learning [32, 87] and genetic programming [111, 122].

The setting we consider is fairly general: Given a canonical convex optimization problem $\min_{x \in P} \phi(x)$ and a set of k different base algorithms which generate an iterate $x_{t,h} \in P, 1 \leq h \leq k$ in every iteration, can we form an adaptive convex combination of the iterates $x_t^{w_t} = \sum_{h=1}^{k} w_{t,h} x_{t,h}$ whose performance is at least as good as the best single algorithm. There is no requirement from the base algorithms other than producing a feasible $x_{t,h} \in P$ in every iteration. In particular, the base algorithms need not guarantee monotonic improvements in the objective function, and may be based on a heuristic without any guarantees. To make our analysis general, we even allow the convex function to change over time. Using advances in online learning and online convex optimization [31, 87, 76, 32, 57], we develop two algorithms for adaptively combining iterates which are guaranteed to be as good as the best convex combination of iterates, and hence the best algorithm.

We extensively evaluate the proposed methodology in an important problem in financial data mining—portfolio selection [91, 36, 61, 2]. The goal is to adaptively update a portfolio over a set of stocks so that the returns over time are maximized. The problem can be posed as an online convex optimization problem, where the convex function gets determined by market movements on each day [2, 61, 36]. Due to its importance, the portfolio selection problem has been widely studied for six decades [91, 75, 36, 32], and numerous algorithms and heuristics exist on how to pick the next days portfolio which forms the iterate $x_{t,h}$ in our setting. We use a pool of these existing algorithms for portfolio selection, and focus on creating a portfolio by adaptively combining the portfolios suggested by the base algorithms. Through our analysis and algorithms, we establish theoretical results and illustrate strong empirical performance. In particular, we show that the meta algorithms for portfolio selection will be universal, i.e., competitive with the best constant rebalanced portfolio (CRP) chosen in hindsight [36, 74, 16], if any base algorithm in the pool is universal. Note that universal portfolios are guaranteed to be as good as the best stock even in adversarial settings. Our experiments show that the meta algorithms outperform all existing universal algorithms by orders of magnitude, by suitably leveraging good heuristics in the pool. For example, trading on S&P500 stocks over the past 21 years (1990-2010), the meta algorithms multiply the

starting wealth by 10^3 times even with two major financial meltdowns. Further, the proposed meta algorithms clearly outperform other simplistic approaches to combining portfolios.

The rest of the chapter is organized as follows. We present a general framework and two algorithms for meta optimization in Section 5.2. In Section 5.3, we specialize the analysis and algorithms to the problem of portfolio selection. We present comprehensive experimental results in Section 5.4, and conclude in Section 5.5.

5.2 Online Meta Optimization

Consider the following generic convex optimization problem which shows up while building models for a variety of data mining tasks [113]:

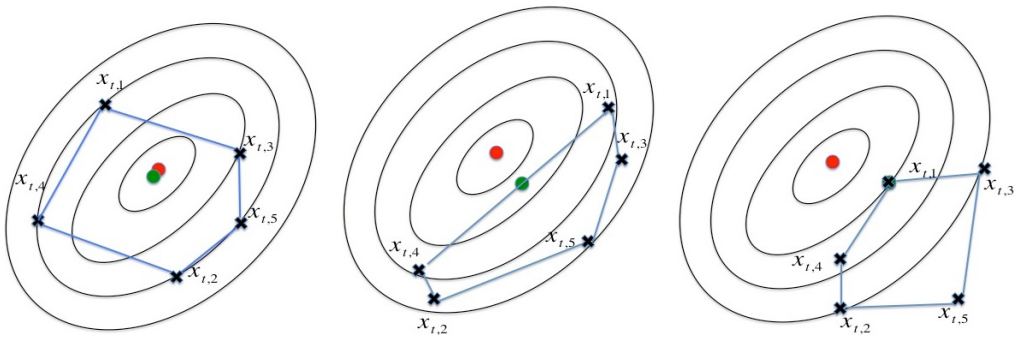
$$\min_{x \in P} \ \phi(x) \ , \tag{5.1}$$

where ϕ is a convex function and $P \in \mathbb{R}^d$ determines the convex feasible set. For the meta-optimization setting, we assume access to k different iterative algorithms $A_1, \ldots, A_k$, referred to as *base algorithms*, which attempt to solve the above problem. In particular, A_h is assumed to generate a feasible $x_{t,h} \in P$ at every iteration. The analysis we present does not depend on any other properties of the base algorithms or the iterates. The iterates may be coming from a iterative convex optimization routines based on gradient or Newton methods, from domain specific heuristics, or even entirely arbitrary guesses. The proposed meta-algorithm picks a suitable iterate from the convex hull of the iterates at any time, given by:

$$\mathrm{Co}(X_t) = \left\{ X_t w = \sum_{h=1}^{k} w_h x_{t,h} \,\middle|\, \sum_{h=1}^{k} w_h = 1, w_h \geq 0 \right\} \ ,$$

where $X_t = [x_{t,1} \ \cdots \ x_{t,k}] \in \mathbb{R}^{d \times k}$ is the matrix of iterates. Let Δ_k denote the k-dimensional simplex. Then, it is easy to see that the best point $x_t^w = X_t w = \sum_h w_h x_{t,h} \in \mathrm{Co}(X_t)$ will always achieve a lower (better) objective function value than any of the individual iterates, i.e.,

$$\min_{w \in \Delta_k} \ \phi\left(\sum_{h=1}^{k} w_h x_{t,h} \right) \leq \phi(x_{t,h}), \quad \forall h \ .$$



(a) (b) (c)

Figure 5.1: The best convex combination x_t^w of the iterates from the base algorithms is always better than individual iterates $x_{t,h}$(the red dot is the global minimum and the green dot is the best point in the convex hull of iterates): (a) x_t^w achieves the global minimum, (b) x_t^w is on an edge of the hull, and (c) x_t^w overlaps with the best iterate.

Figure 5.1 shows examples to illustrate the above point. In Figure 5.1(a), the best point in the convex hull of the iterates achieves the global minimum of the function; in Figure 5.1(b), it is nearest to the global minimum; and in Figure 5.1(c), the best point in the convex hull is an iterate itself, i.e., a corner of the hull.

In general, the best point $x_t^w = X_t w \in \mathrm{Co}(X_t)$ inside the convex hull or equivalently the best convex combination $w \in \Delta_k$ cannot be obtained in closed form. One can design optimization algorithms to find the best point inside the convex hull. Note that such computations have to be repeated at every iteration, since corners of the hull, determined by X_t, changes in every iteration. In this section, we develop algorithms which adaptively pick $w_t \in \Delta_k$ based on X_{t-1}, and show that the iterates $x_t^{w_t} = X_t w_t = \sum_h w_{t,h} x_{t,h}$ of the meta-algorithm are competitive with any fixed convex combination $w \in \Delta_k$ used over iterations, i.e., $\forall w \in \Delta_k$ we have

$$\sum_{t=1}^{T} \phi(X_t w_t) \leq \sum_{t=1}^{T} \phi\left(X_t w\right) + o(T) \ . \tag{5.2}$$

In particular, if any $w^* \in \Delta_k$ achieves the global minimum, the adaptive approach will find the global minimum as well. Indeed, instead of simply being competitive with the single best iterate, the adaptive $x_t^{w_t}$ will be competitive with any convex combinations of them (Figure 5.1). To present our analysis in its full generality, we consider the online convex optimization (OCO) setting [127], where the convex function itself can change

over time. We denote the convex function at time t to be ϕ_t. Note that we can recover the batch case analysis for a fixed ϕ as a special case by simply setting $\phi_t = \phi, \forall t$. In the OCO setting, we intend to get a set of adaptive iterates $x_t^{w_t}$ such that the following form of *regret bounds* are satisfied:

$$\sum_{t=1}^{T} \phi_t(X_t w_t) \leq \min_{w \in \Delta_k} \sum_{t=1}^{T} \phi_t \left(X_t w_t\right) + o(T) \ . \tag{5.3}$$

5.2.1 Online Gradient Updates

Our meta algorithm and analysis for Online Gradient Updates (OGU) involves suitably reducing the Online Meta Optimization (OMO) problem to an online learning problem over k experts [87, 32], where each expert corresponds to a corner for meta-optimization. We start by recalling a standard result from the online learning literature [87, 50, 9]:

Lemma 1. *Let $\ell_t \in [0,1]^k, t = 1, \ldots, T$, be an arbitrary sequence of loss vectors over the k experts. If one maintains an adaptive distribution over the experts using multiplicative updates given by $p_{t+1}(h) = p_t(h) \exp(-\eta \ell_t(h))/Z_t$, where $\eta > 0$ and Z_t is the partition function, then for any $w \in \Delta_k$, the following inequality holds:*

$$\sum_{t=1}^{T} p_t^T \ell_t \leq \frac{\eta \sum_{t=1}^{T} w^T \ell_t + \log k}{1 - \exp(-\eta)} \ . \tag{5.4}$$

Variants of the above result form the basis of much work in online learning, boosting, game theory, and numerous other developments in the past two decades [87, 50, 49, 6, 9, 32]. We now outline a transformation of the OMO problem to the above online learning setting.

For our analysis, we assume that the sequence of convex functions ϕ_t can be arbitrary, but satisfies $\|\nabla \phi_t(x)\|_\infty \leq g_\infty$ for $x \in P$. Further, we assume $x \in P$ satisfies $\|x\|_1 \leq c$. For the portfolio selection application in Section 5.3, we will obtain specific values for g_∞ and c. Let

$$f_t(w) = \phi_t \left(X_t w\right) \ . \tag{5.5}$$

Since $\phi_t : P \mapsto \mathbb{R}$, where $P \subseteq \mathbb{R}^d$, is a convex function, the function $f_t : \Delta_k \mapsto \mathbb{R}$ is also convex. To see this, first note that the Hessian $\nabla^2 f_t(w) = X_t^T \nabla^2 \phi_t(X_t w) X_t$. Since ϕ_t

Algorithm 5 Online Gradient Update (OGU) for Meta Optimization

Initialize $w_{1,h} = \frac{1}{k}, h = 1, \ldots, k$
For $t = 1, \ldots, T$
    Receive $X_t = [x_{t,1} \ \cdots \ x_{t,k}]$ from base algorithms
    Compute $x_t^{w_t} = \sum_{h=1}^{k} w_{t,h} x_{t,h}$
    Receive convex function ϕ_t from nature
    Update distribution

$$w_{t+1,h} = w_{t,h} \exp(-\eta \ell_t(h))/Z_t$$

Regret bound: $2cg_\infty \left(\sqrt{2T \log k} + \log k\right) = o(T)$.

is convex, $\nabla^2 \phi(X_t w)$ is positive semi-definite. Hence, $\nabla^2 f_t(w)$ is positive semi-definite, implying convexity of f_t. Define loss vector

$$\ell_t = \frac{1}{2}\left(\frac{\nabla f_t(w_t)}{cg_\infty} + e\right) \in \mathbb{R}^k \ , \tag{5.6}$$

where e is the all ones vector. Based on this definition of loss, Algorithm 5 presents an adaptive algorithm for Online gradient update for meta optimization. We establish the following regret bound for OGO for this algorithm:

Theorem 7. *For any sequence of convex functions ϕ_t such that $\|\nabla \phi_t(x)\|_\infty \leq g_\infty$, and any sequence of iterates $X_t = [x_{t,1} \ \cdots \ \cdots \ x_{t,k}]$ such that $\|x_{t,h}\|_1 \leq c$, for $\eta = \log\left(1 + \sqrt{\frac{2 \log k}{T}}\right)$ in Algorithm 5, we have*

$$\begin{aligned} &\sum_{t=1}^{T} f_t(w_t) - \min_{w \in \Delta_k} \sum_{t=1}^{T} f_t(w) \\ &\qquad \leq 2cg_\infty \left(\sqrt{2T \log k} + \log k\right) \ . \end{aligned} \tag{5.7}$$

Proof. Since $\nabla f_t(w_t) = X_t^T \nabla \phi_t(X_t w_t)$, $\|\nabla f_t(w_t)\|_\infty = \max_h |x_{t,h}^T \nabla \phi_t(x_t^{w_t})|$. From Hölder's inequality [121, 54, 77],

$$|x_{t,h}^T \nabla \phi_t(X_t w_t)| \leq \|x_{t,h}\|_1 \|\nabla \phi_t(X_t w_t)\|_\infty \leq cg_\infty \ .$$

Hence $\frac{\nabla f_t(w_t)}{cg_\infty} \in [-1,1]^k$, so that $\ell_t \in [0,1]^k$. From Lemma 1, Algorithm 5 will satisfy (7.11). Let $\epsilon = 1 - \exp(-\eta)$ so that from Lemma 1 we have

$$\sum_{t=1}^{T} w_t^T \ell_t - \sum_{t=1}^{T} w^T \ell_t \leq \epsilon T + \frac{1}{\epsilon} \log k \ ,$$

where we have used $\sum_{t=1}^{t} w^T \ell_t \leq T$. Choosing $\epsilon = \frac{\sqrt{2 \log k}}{\sqrt{2 \log k} + \sqrt{T}}$, a direct calculation shows

$$\sum_{t=1}^{T} \ell_t^T (w_t - w) \leq \sqrt{2T \log k} + \log k \ . \tag{5.8}$$

Now, since f_t is convex, we have

$$f_t(w_t) - f_t(w) \leq \nabla f_t(w_t)^T (w_t - w) = 2cg_\infty \ell_t^T (w_t - w) \ ,$$

where the last equality follows since $e^T(w_t - w) = 0$ as $w_t, w \in \Delta_k$. Adding over all t and using (5.8), we have

$$\begin{aligned} \sum_{t=1}^{T} f_t(w_t) - \sum_{t=1}^{T} f_t(w) &\leq 2cg_\infty \sum_{t=1}^{T} \ell_t^T (w_t - w) \\ &\leq 2cg_\infty \left(\sqrt{2T \log k} + \log k \right) \ . \end{aligned}$$

Noting that the above inequality holds for any $w \in \Delta_k$ completes the proof. □

Since $2cg_\infty \left(\sqrt{2T \log k} + \log k \right) = o(T)$, we have a desired form of the bound. Further, assuming $\phi_t = \phi$ gives the corresponding bound for the batch optimization case.

5.2.2 Online Newton Updates

Our analysis for Online Newton Updates (ONU) build on recent advances in Online Convex Optimization (OCO) [57, 2]. The analysis of ONU differs from the standard analysis of online Newton step [57] due to two reasons: first, our analysis focuses on the derived convex function $f_t : \Delta_k \mapsto \mathbb{R}$ instead of the original convex function $\phi_t : P \mapsto \mathbb{R}$, and second, our bounds are based on the L_∞ norm of ϕ_t instead of the L_2 norm, which can be substantially larger for high-dimensional problems.

Following [57], we consider convex functions ϕ_t which satisfy the α-exp-concavity property: there is a $\alpha > 0$ such that for $x \in P$, $\exp(-\alpha \phi_t(x))$ is a concave function . Note that α-exp-concave functions ϕ_t are more general than ones which have bounded

Algorithm 6 Online Newton Update (ONU) for Meta Optimization

Initialize $w_1 \in \Delta_k$, let $\beta = \min\left\{\frac{1}{8cg_\infty}, \alpha\right\}$
For $t = 1, \ldots, T$
  Receive $X_t = [x_{t,1} \ \cdots \ x_{t,k}]$ from base algorithms
  Compute $x_t^{w_t} = \sum_{h=1}^{k} w_{t,h} x_{t,h}$
  Receive convex function ϕ_t from nature
  Update distribution

$$w_{t+1,h} = \prod_{\Delta_k}^{A_t}\left(w_t - \frac{2}{\beta}A_t^{-1}\nabla f_t\right) \ ,$$

Regret bound: $k\left(8cg_\infty + \frac{1}{\alpha}\right)\log\frac{eT}{a}$.

gradients and Hessians which are strictly bounded away from 0, i.e., $\nabla^2\phi_t \succeq H\mathbb{I}$ for some constant $H > 0$. As before, we assume that L_∞ norm of the gradient of ϕ_t are bounded above, i.e., $\|\nabla\phi_t\|_\infty \leq g_\infty$.

With these assumptions, Algorithm 6 presents the Online Newton Update (ONU) algorithm for Online Meta Optimization [57]. In essence, the algorithm takes a Newton-like step from the current iterate w_t, and then projects the vector to the feasible set Δ_k to obtain w_{t+1}. Note that the algorithm does not use the actual Hessian of f_t, but a matrix based on the outer product of the gradients defined as:

$$A_t = \sum_{\tau=1}^{t} \nabla f_t \nabla f_t^T + \epsilon\mathbb{I} \ , \tag{5.9}$$

where $\epsilon = \frac{k}{\beta^2 c^2}$. Further $\prod_{\Delta_k}^{A_t}$ is the projection onto Δ_k using the Mahalanobis distance induced by A_t, i.e.,

$$\prod_{\Delta_k}^{A_t}(\tilde{w}) = \operatorname*{argmin}_{w\in\Delta_k} (w-\tilde{w})^T A_t^{-1}(w-\tilde{w}) \ . \tag{5.10}$$

We start our analysis by showing that if ϕ_t is α-exp-concave for $x \in P$, then f_t is α-exp-concave for $w \in \Delta_k$ for the same (set of) α.

Lemma 2. *If ϕ_t is α-exp-concave for some $\alpha > 0$, then f_t as defined in (5.5) is also α-exp-concave.*

Proof. Let $h_t(w) = \exp(-\alpha f_t(w))$. The Hessian is given by

$$\begin{aligned}\nabla^2 h_t(w) &= [\alpha^2(\nabla f_t)(\nabla f_t)^T - \alpha\nabla^2 f_t]h_t(w) \\ &= X_t^T[\alpha^2(\nabla\phi_t)(\nabla\phi_t)^T - \alpha\nabla^2\phi_t]X_t h_t(w)\end{aligned}$$

Let $\psi_t(x) = \exp(-\alpha\phi_t(x))$. Since ϕ_t is α-exp-concave, the Hessian $\nabla^2\psi_t \preceq 0$, so that

$$[\alpha^2(\nabla\phi_t)(\nabla\phi_t)^T - \alpha\nabla^2\phi_t]\psi_t(x) \preceq 0 \ .$$

Let $B_t = [\alpha^2(\nabla\phi_t)(\nabla\phi_t)^T - \alpha\nabla^2\phi_t]$. Since $\psi_t(x) \geq 0$, we have

$$B_t \preceq 0 \quad \Rightarrow \quad X_t^T B_t X_t \preceq 0 \ ,$$

so that $\nabla^2 h_t \preceq 0$ since $h_t(w) \geq 0$, implying h_t is α-exp-concave. □

We now establish a result, similar to Lemma 3 in [57], but using the L_∞ bound g_∞ and the fact that $\|x\|_1 \leq c$ for $x \in P$.

Lemma 3. *For $\beta \leq \min\{\frac{1}{8cg_\infty}, \alpha\}$, for any $w, w_t \in \Delta_k$, we have*

$$\begin{aligned}f_t(w) \geq f_t(w_t) &+ \nabla f_t(w_t)^T(w - w_t) \\ &+ \frac{\beta}{4}(w - w_t)^T\nabla f_t(w_t)\nabla f_t(w_t)^T(w - w_t) \ .\end{aligned} \tag{5.11}$$

Proof. Since $\beta \leq \alpha$, following the proof of Lemma 3 in [57] we have

$$f_t(w) \geq f_t(w_t) - \frac{1}{\beta}\log[1 - \beta\nabla f_t(w_t)^T(w - w_t)] \ .$$

Now, by Hölder's inequality,

$$|\beta\nabla f_t(w_t)^T(w - w_t)| \leq \beta\|\nabla f_t(w_t)\|_\infty\|w - w_t\|_1 \leq 2\beta c g_\infty \leq \frac{1}{4} \ .$$

Since $-\log(1 - z) \geq z + \frac{1}{4}z^2$ for $|z| \leq \frac{1}{4}$, using it for $z = \beta\nabla f_t(w_t)^T(w - w_t)$ completes the proof. □

We now present the main result for ONU:

Theorem 8. *For any sequence of α-exp-concave functions ϕ_t such that $\|\nabla\phi_t\|_\infty \le g_\infty$ for $x \in P$ where $\|x\|_1 \le c$, for $T \ge 2a$ where $a = \frac{32g_\infty}{c^2}$, we have the following regret bound:*

$$\sum_{t=1}^{T} f_t(w_t) - \min_{w\in\Delta_k} \sum_{t=1}^{T} f_t(w) \le k\left(8cg_\infty + \frac{1}{\alpha}\right)\log\frac{eT}{a} \ . \tag{5.12}$$

Proof. Using Lemma 3 and using the proof of Theorem 2 in [57], we have

$$\sum_{t=1}^{T} R_t \le \frac{1}{\beta}\sum_{t=1}^{T} \nabla_t^T A_t^{-1}\nabla_t + \frac{\beta}{4}(w_t - w)^T(A_1 - \nabla_1\nabla_1^T)(w_1 - w) \ ,$$

where $R_t = f_t(w_t) - f_t(w)$ for any $w \in \Delta_k$, $\nabla_t = \nabla f_t$, and $A_t = \sum_{\tau=1}^{t} \nabla_\tau\nabla_\tau^T + \epsilon\mathbb{I}$ as in (5.9). Since $A_1 - \nabla_1\nabla_1^T = \epsilon\mathbb{I}$, $\|w_1 - w\|_2^2 \le 4c^2$, and $\epsilon = \frac{k}{\beta^2c^2}$, we have

$$\begin{aligned}\sum_{t=1}^{T} R_t &\le \frac{1}{\beta}\sum_{t=1}^{T} \nabla_t^T A_t^{-1}\nabla_t + \frac{\beta}{4}\epsilon\|w_1 - w\|_2^2 \\ &\le \frac{1}{\beta}\sum_{t=1}^{T} \nabla_t^T A_t^{-1}\nabla_t + \frac{k}{\beta} \ .\end{aligned}$$

Since $\|\nabla f_t\| \le \sqrt{k}\|\nabla f_t\|_\infty \le \sqrt{k}cg_\infty$, from Lemma 11 in [57], we have

$$\sum_{t=1}^{T} \nabla_t^T A_t^{-1}\nabla_t \le k\log\left(\frac{kc^2g_\infty^2T}{\epsilon} + 1\right) \le k\log\left(\frac{T}{2a} + 1\right) \ ,$$

where we have used $\epsilon = \frac{k}{\beta^2c^2}, \beta \le \frac{1}{8cg_\infty}$, and $a = \frac{32g_\infty}{c^2}$. For $T \ge 2a$, $\frac{T}{2a} + 1 \le \frac{T}{a}$. Plugging everything back, we have

$$\sum_{t=1}^{T} R_t \le \frac{k}{\beta}\left(\log\frac{T}{a} + 1\right) = \frac{k}{\beta}\log\frac{eT}{a} \ .$$

Since $\beta = \min\{\frac{1}{8cg_\infty}, \alpha\}$, we have

$$\frac{1}{\beta} = \max\left\{8cg_\infty, \frac{1}{\alpha}\right\} \le 8cg_\infty + \frac{1}{\alpha} \ .$$

Plugging this upper bound back completes the proof. □

5.3 Meta Optimization for Portfolio Selection

We recap the online portfolio selection framework here. We use a slightly different notation here compared to the previous chapters. We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over a span of T periods. Let $r_t(i)$ denote the *price relative* of stock s_i in day t, i.e., the multiplicative factor by which the price of s_i changes in day t. Hence, $r_t(i) > 1$ implies a gain, $r_t(i) < 1$ implies a loss, and $r_t(i) = 1$ implies the price remained unchanged. We assume $r_t(i) > 0$ for all i, t. Let $r_t = \langle r_t(1), \ldots, r_t(n) \rangle$ denote the vector of price relatives for day t, and let $r_{1:t}$ denote the collection of such price relative vectors upto and including day t. A portfolio $x_t = \langle x_t(1), \ldots, x_t(n) \rangle$ on day t can be viewed as a probability distribution over the stocks that prescribes investing $x_t(i)$ fraction of the current wealth in stock $s_t(i)$. Note that the portfolio x_t has to be decided before knowing r_t which will be revealed only at the end of the day. The multiplicative gain in wealth at the end of day t, is then simply $r_t^T x_t = \sum_{i=1}^{n} r_t(i) x_t(i)$. Given a sequence of price relatives $r_{1:t-1} = \{r_1, \ldots, r_{t-1}\}$ upto day $(t-1)$, the sequential portfolio selection problem in day t is to determine a portfolio x_t based on past performance of the stocks. At the end of day t, r_t is revealed and the actual performance of x_t gets determined by $r_t^T x_t$. Over a period of T days, for a sequence of portfolios $x_{1:T} = \{x_1, \ldots, x_t\}$, the multiplicative gain in wealth is then

$$S(x_{1:T}, r_{1:T}) = \prod_{t=1}^{T} \left(r_t^T x_t \right) \ . \tag{5.13}$$

The above problem can be viewed as an Online Convex Optimization (OCO), where the convex function $\phi_t(x_t) = -\log(r_t^T x_t)$, and the cumulative loss over T iterations is

$$\sum_{t=1}^{T} \phi_t(x_t) = -\sum_{t=1}^{T} \log(r_t^T x_t) = -\log S(x_{1:T}, r_{1:T}) \ . \tag{5.14}$$

There are numerous algorithms in the literature for picking the portfolio x_t on a given day based on past information $r_{1:(t-1)}$ [36, 61, 32, 2, 18]. Instead of proposing new algorithms for the task, we focus on meta optimization to combine the portfolios from a pool of base algorithms from the literature. We now specialize the general case results and algorithms of Section 5.2 to the task of portfolio selection.

Consider k base algorithms $\{A_1, \ldots, A_k\}$ for portfolio selection where algorithm A_h generates a portfolio $x_{t,h} \in \Delta_n$ based on the past information $r_{1:(t-1)}$. Recall that

our analysis does not impose any other constraints on the base algorithms and so they can be based on theoretically well grounded ideas [36, 61, 2] or good heuristics [18]. Given the set of base portfolios $X_t = [x_{t,1} \ \cdots \ x_{t,k}]$, the goal of the meta algorithm is to choose $w_t \in \Delta_k$ to construct the portfolio $x_t^{w_t} = X_t w_t$ and subsequently incur loss $f_t(w_t) = \phi_t(x_t^{w_t}) = -\log(r_t^T x_t^{w_t}) = -\log(r_t^T X_t w_t)$. Since a portfolio $x \in \Delta_n$, we have $c = \|x\|_1 = 1$. Among all price relatives over all stocks, let $r_{\min} = \min_{i,t} r_t(i) > 0$ and let $r_{\max} = \max_{i,t} r_t(i)$. Since $\nabla\phi_t(x) = -\frac{r_t}{r_t^T x}$, $\|\nabla\phi_t(x)\|_\infty \leq \frac{r_{\max}}{r_{\min}} = g_\infty$. For convenience, we use $\bar{u} = \frac{r_{\max}}{r_{\min}}$. For our subsequent analysis, we note that

$$\nabla f_t(w_t) = -\frac{X_t^T r_t}{r_t^T X_t w_t} \ . \tag{5.15}$$

Gradient Updates: Since $\nabla\phi_t(x)$ for portfolio selection is a positive vector, one can define the loss vector for OGU in Algorithm 5 as follows:

$$\ell_t = \frac{\nabla f_t(w_t)}{c g_\infty} = -\frac{1}{2\bar{u}} \frac{X_t^T r_t}{r_t^T X_t w_t} + e \ . \tag{5.16}$$

With this modification, the OGU in Algorithm 5 has the following guarantee:

Corollary 1. *For any sequence of price relatives $r_{1:T}$ and any sequence of base portfolios $X_{1:T}$, the log-wealth accumulated by Algorithm 5 choosing adaptive w_t satisfies the following regret bound:*

$$\begin{aligned} \max_{w \in \Delta_k} \sum_{t=1}^{T} \log(r_t^T X_t w) - \sum_{t=1}^{T} \log(r_t^T X_t w_t) \\ \leq \bar{u} \left(\sqrt{2T \log k} + \log k \right) \ . \end{aligned} \tag{5.17}$$

The proof follows from a direct application of Theorem 7. We briefly discuss the implication of the fact that the wealth accumulated by the adaptive meta algorithm will be competitive with any fixed combination strategy chosen in hindsight. If one of the base algorithms is *universal* [36, 61, 16, 73, 32], i.e., competitive with best constant rebalanced portfolio (CRP) [36, 16] in hindsight so that

$$\max_{x \in \Delta_n} \sum_{t=1}^{T} \log(r_t^T x) - \sum_{t=1}^{T} \log(r_t^T x_t) = o(T) \ , \tag{5.18}$$

then our meta algorithm will also be universal. Also, since the best CRP would outperform the best stock, having an universal algorithm in the pool is sufficient to ensure

the meta algorithm will be competitive with the single best stock in hindsight. More generally, the meta algorithm will be competitive with the best convex combination of the base algorithms, which is guaranteed to be better than the best base algorithm in the pool (Figure 5.1).

Newton Updates: We start our analysis with the following result:

Lemma 4. *$\phi_t(x) = -\log(r_t^T x)$ is a α-exp-concave function for $\alpha \in (0, 1]$.*

Proof. Let $\psi_t(x) = \exp(-\alpha\phi_t(x)) = (r_t^T x)^\alpha$. A direct calculation shows the Hessian to be

$$\nabla^2\psi_t(x) = \alpha(\alpha - 1)\frac{r_t r_t^T}{r_t^T x}\psi_t(x) \ ,$$

which is negative semi-definite for $\alpha > 0$ if $\alpha \in (0, 1]$. □

As a result, Algorithm 6 can be applied as a meta algorithm for the portfolio selection problem. As before, $c = 1, g_\infty = \bar{u}$. Choosing $\alpha = 1$, $\beta = \min\{\frac{1}{8\bar{u}}, \alpha\} = \frac{1}{8\bar{u}}$ since $\bar{u} = \frac{r_{\max}}{r_{\min}} \geq 1$. Hence, $\frac{1}{\beta} = 8\bar{u}$. Further, $a = \frac{32 g_\infty}{c^2} = 32\bar{u}$, so that $\frac{a}{e} \leq 12\bar{u}$. Using the above values in Algorithm 6, from Theorem 8 we have the following result:

Corollary 2. *For any sequence of price relatives $r_{1:T}$ and any sequence of base portfolios $X_{1:t}$, the log-wealth accumulated by Algorithm 6 choosing adaptive w_t satisfies the following regret bound:*

$$\max_{w \in \Delta_k} \sum_{t=1}^{T} \log(r_t^T X_t w) - \sum_{t=1}^{T} \log(r_t^T X_t w_t) \leq 8k\bar{u} \log \frac{T}{12\bar{u}} \ . \tag{5.19}$$

As before, the proof follows from a direct application of Theorem 8. The bound has the same optimality properties as discussed in the context of OGU above. In fact, the worst case regret of ONU grows as $O(\log T)$ as opposed to $O(\sqrt{T})$ for OGU. The bound for OGU can in fact be sharpened in this setting by suitably modifying the OGU algorithm and analysis using the fact that the Hessian $\nabla^2 f_t$ is bounded away from 0 under the assumption $\min_{i,t} r_{i,t} > 0$.

5.4 Experimental Results

We conducted extensive experiments on two financial data-sets to establish how effective Online Meta Optimization can be carried out by OGU and ONU. In this section, we describe the datasets that were chosen for the experiments, the algorithms, the parameter choices and most importantly the results of our experiments.

Datasets: The experiments were conducted on two major datasets: the New York Stock Exchange dataset (NYSE) [36] and a Standard & Poor's 500 (S&P 500) dataset. The NYSE dataset consists of 36 stocks with data accumulated over a period of 22 years from July 3, 1962 to Dec 31 1984. The dataset captures the bear market that lasted between January 1973 and December 1974. However, all of the 36 stocks increase in value in the 22-year run.

The S&P500 dataset that we used for our experiments consists of 263 stocks which were present in the S&P500 index in December 2010 and were alive since January 1990. This period of 21 years from 1990 to 2010 covers bear and bull markets of recent times.

Methodology: We ran a pool of base portfolio selection algorithms and the Meta Algorithms on the datasets (NYSE and S&P500). For our experiments, this pool included universal and non-universal algorithms. We start by briefly describing the base algorithms and the Meta Algorithms.

5.4.1 Base Algorithms

Of the base algorithms that we used for our experiments UP, EG and ONS are *universal* while Anticor and its variant are heuristics.

Universal Portfolios (UP): The key idea behind Cover's [36] UP is to maintain a distribution over all Constant Rebalanced Portfolios (CRPs) and perform a Bayesian update after observing every r_t. Each CRP q is a distribution over n stocks and hence lies in the n-simplex, one uses a distribution $\mu(q)$ over the n-simplex. The universal portfolio x_t is defined as:

$$x_t(i) = \frac{\int_q q(i) S_{t-1}(q, r_{1:t-1}) \mu(q) dq}{\int_q S_{t-1}(q, r_{1:t-1}) \mu(q) dq} \ . \tag{5.20}$$

UP has a regret of $O(\log T)$ with respect to the best CRP in hindsight. However, the updates for UP are computationally prohibitive.

Exponentiated Gradient (EG): Exponentiated Gradient (EG) [61] scales linearly with the number of stocks but is weaker in regret than UP. The EG investment strategy was introduced and analyzed by [61]. At the start of day t, the algorithm computes its new portfolio vector x_t such that it stays close to x_{t-1} and does well on the price relatives r_{t-1} for the previous day. The updated portfolio turns out to be

$$x_t(i) = \frac{x_{t-1}(i)\exp(\eta r_{t-1}(i)/x_{t-1}^T r_{t-1})}{\sum_{i'=1}^{n} x_{t-1}(i')\exp(\eta r_{t-1}(i')/x_{t-1}^T r_{t-1})} \ . \tag{5.21}$$

where $\eta > 0$ is a parameter called the learning rate.

Online Newton Step (ONS): ONS uses a Newton step based method to compute the portfolio for the next iteration [2]. The Online Newton Step method can be shown to achieve sublinear regret and hence is an universal strategy.

The ONS algorithm uses following portfolio update method for round $t > 1$:

$$x_t = \prod_{\Delta_n}^{\mathbf{A_t - 1}} \left(x_{t-1} - \frac{1}{\beta}\mathbf{A_{t-1}^{-1}}\nabla_{t-1} \right) \tag{5.22}$$

where $\nabla_t = \nabla[\log(x_t \cdot r_t)]$, $\mathbf{A}_t = \sum_{\tau=1}^{t} \nabla_\tau \nabla_\tau + \mathbb{I}$, β is a non-negative constant, and $\prod_{\triangle_\mathbf{n}}^{\mathbf{A_t - 1}}$ is the projection onto the n-simplex Δ_n.

Anticor: Anticor is a heuristic based method which does not confirm to the *universal* property for portfolio selection algorithms [18]. Here learning the best stocks (to invest money in) is done by exploiting the volatility of the market and the statistical relationship between the stocks. It implements the 'reversal to the mean' market phenomenon rather aggressively. One of the most important parameters for Anticor is the window length *win*. The version of Anticor implemented, works with two most recent windows of length *win*. The strategy is to move money from a stock i to stock j if the growth rate of stock i is greater than the growth rate of j in the most recent window. An additional condition that requires to be satisfied is the existence of a positive correlation between stock i in the second last window and stock j in the last window. For more details on the Anticor algorithm please refer to [18]. The experiments with different variations of the Anticor algorithm in [18], brought to the fore the exceptional empirical performance improvement that this heuristic-based approach can achieve over theoretically motivated approaches.

The performance of Anticor is sensitive to the window size *win* [18]. One way to address this issue is to adaptively learn the weights and invest in a weighted version of all

Anticor$_{win}$s where $win \leqslant W$. We consider a variant BAH(Anticor$_W$), which maintains a uniform buy-and-hold investment on the Anticor$_{win}$, $win \in [2, W]$.

5.4.2 Meta Algorithms

Meta Algorithms (MAs) are constructed by combining a pool of base algorithms. Meta algorithm MA$_{EG}$ uses the gradient updates and this follows from OGU in Algorithm 5 and Meta Algorithm MA$_{ONS}$ uses Newton updates and follows from ONU in Algorithm 6. Meta Algorithms MA$_{EG}$ and MA$_{ONS}$ are universal if at least one of the base algorithms in their pool is universal.

Additionaly, we used two other versions of Meta Algorithms for our experiments: MA$_{Anticor}$ and MA$_{BAH}$. MA$_{Anticor}$ is a Meta Algorithm version of Anticor. Like Anticor it works with different window lengths over the pool of base algorithms. MA$_{BAH}$ does a uniform buy-and-hold over the base algorithms and does not move money between the algorithms. Unlike MA$_{EG}$ and MA$_{ONS}$, MA$_{Anticor}$ and MA$_{BAH}$ have no performance guarantees.

5.4.3 Results

The experimental setup can be broadly categorized into three subcategories: (a) Universal Pool, (b) Mixed Pool 1 and (c) Mixed Pool 2 based on the pool of base algorithms that were used by the MAs.

Universal Pool: In the Universal Pool setup, the MAs worked with universal base algorithms UP, EG, and ONS. Figure 5.2 shows the wealth accumulated by the universal base algorithms on the NYSE and S&P500 datasets. We see that ONS performs best, followed by EG and UP. Figure 5.2 also shows that the two Meta Algorithms MA$_{EG}$ and MA$_{ONS}$ are able to catch up with the performance of ONS, the best performing algorithm in the pool.

Mixed Pool 1: The Mixed Pool 1 is formed by adding Anticor to the Universal Pool of base algorithms. In Figure 5.3, we see that there is a stark difference in the wealth garnered by Anticor as compared to the Universal Pool of base algorithms. While for the NYSE dataset, Anticor's wealth is of the order of 10^6 (almost 10^4 times the wealth gathered by ONS), for S&P500, the wealth reaches the order of 10^3. MA$_{EG}$ is able to

catch up with the performance of Anticor for both the datasets. MA_{ONS} is very slightly behind Anticor and MA_{EG} on the NYSE dataset. On S&P500, the base algorithm ONS outperforms MA_{ONS} (which is still better than UP and EG by a sybstantial margin).

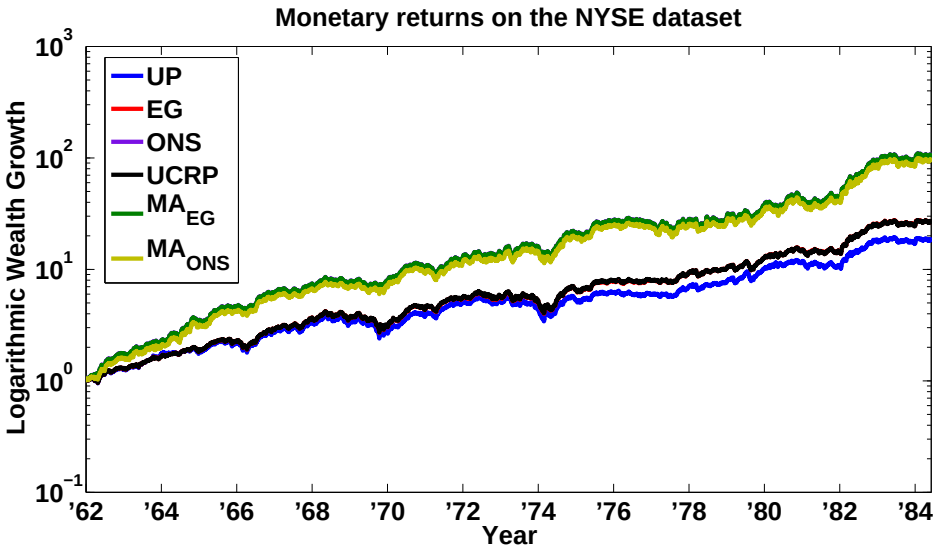
Mixed Pool 2: To further emphasize the strength of the MAs we formed Mixed Pool 2 of base algorithms by adding $\text{BAH}(\text{Anticor}_W)$ to Mixed Pool 1. $\text{BAH}(\text{Anticor}_W)$ outperforms Anticor_{win} (for $win \leqslant W$), EG, UP, and ONS. Figure 5.4 shows that the wealth achieved by MA_{EG} and MA_{ONS} with $\text{BAH}(\text{Anticor}_W)$ in the pool, is almost as much as $\text{BAH}(\text{Anticor}_W)$ itself. Thus we see that Meta Algorithms, MA_{EG} and MA_{ONS} are competitive with the best base algorithm in all the three experimental setups.

Figure 5.5, shows the performance of $\text{MA}_{Anticor}$ and MA_{BAH} with the Mixed Pool 2 of base algorithms. The performance of the $\text{MA}_{Anticor}$ is inferior than both MA_{EG} and MA_{ONS}. We could attribute this inferior performance to the inherent nature of Anticor which will tend to move money away from the base algorithms which are performing well. With a buy-and-hold version of MA called MA_{BAH}, the wealth gained is more than Anticor_w, but less than that of $\text{BAH}(\text{Anticor}_W)$ and MA_{EG} and MA_{ONS}.

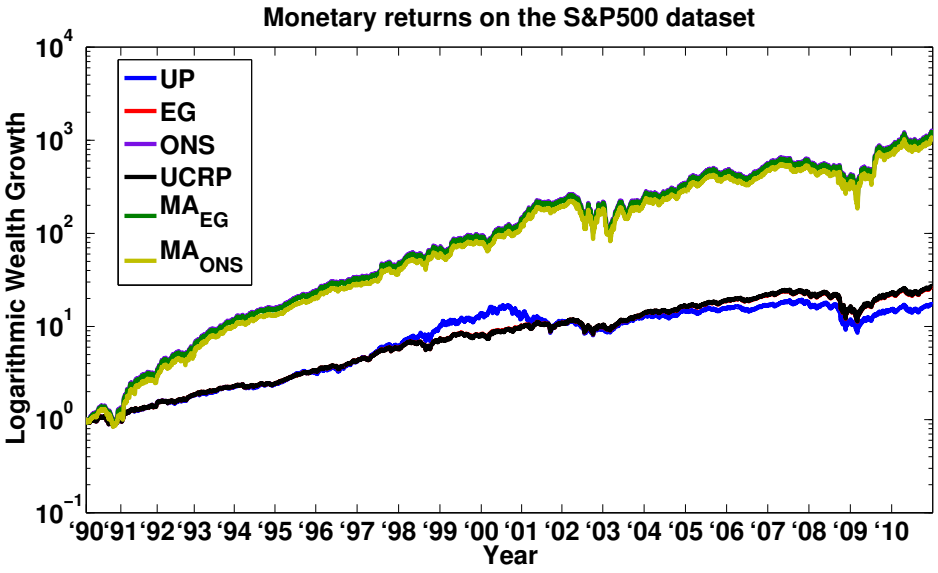
Parameter choices: Parameter choices had to be made for the universal as well as non-universal base algorithms. For EG, we experimented with different learning rate (η) values and found $\eta = 0.05$ to be a good choice, validating the observations in [61]. For ONS, β value was chosen as 1. This gave better results than when β was chosen as a function of the market-variability (refer to [2] for details).

The window size w for Anticor_{win} was taken to be 30. The performance of Anticor_{win} is a function of win as demonstrated in [18]. $\text{BAH}(\text{Anticor}_W)$ combines multiple Anticor_{win} algorithms, $win \in [2, W]$ to harness the strength of these different versions. We choose $W = 30$ for our experiments which might not be the optimal value for the NYSE and S&P500 datasets. Hoever, it is observed that $\text{BAH}(\text{Anticor}_{30})$ surpasses all the base universal algorithms and Anticor_{30} in terms of empirical performance and helps us establish that MA_{EG} and MA_{ONS} can always manage to track the best strategy even if it is a heuristic.

The rate at which MA_{EG} caught up with Anticor in terms of wealth increased as we increased the value of η. However, for very high values of η ($\eta \geqslant 50$), the increase in the wealth gathered by MA_{EG} changed by a small amount. For MA_{ONS}, the β value was chosen according to Lemma 3. $\text{MA}_{Anticor}$ was run with different window lengths.

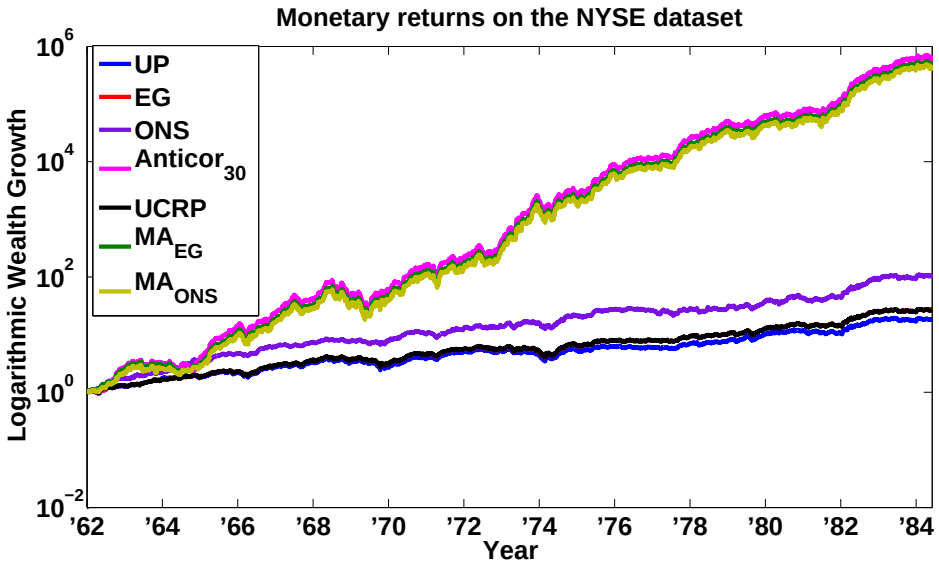


(a) Monetary returns on NYSE.

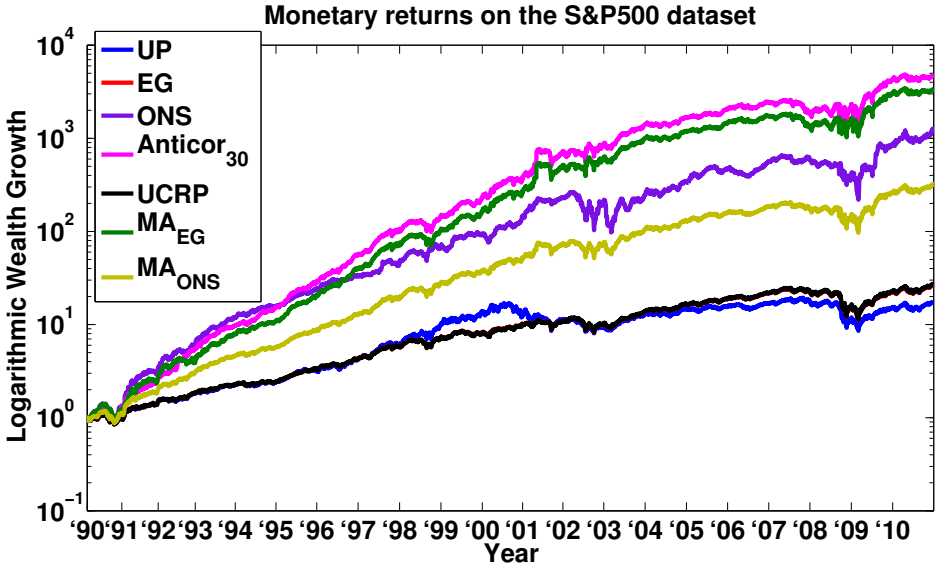


(b) Monetary returns on S&P500.

Figure 5.2: Monetary returns of the Meta Algorithms, MA $_{EG}$ and MA $_{ONS}$ for \$1 investment, is competitive with the best performing base algorithm ONS in this case(best viewed in color).

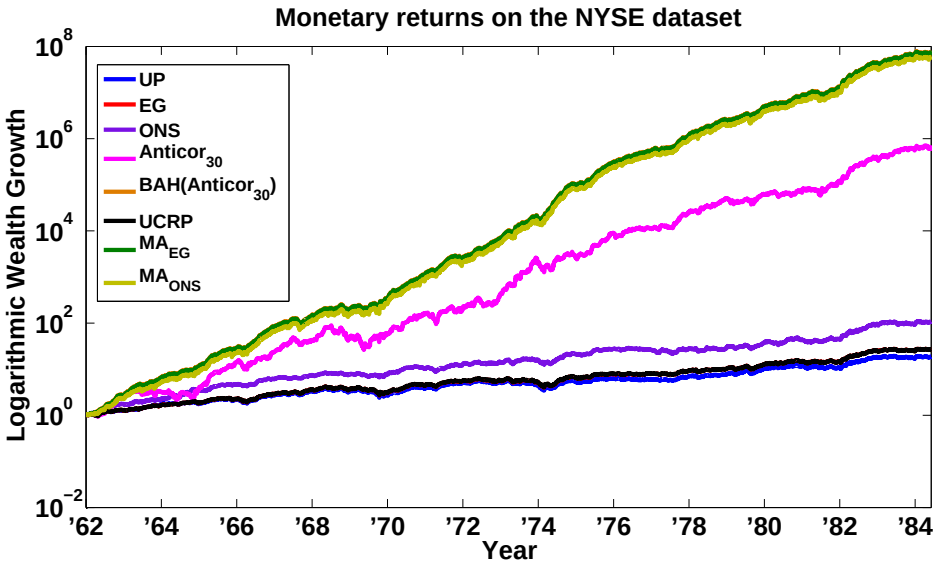


(a) Monetary returns on NYSE.

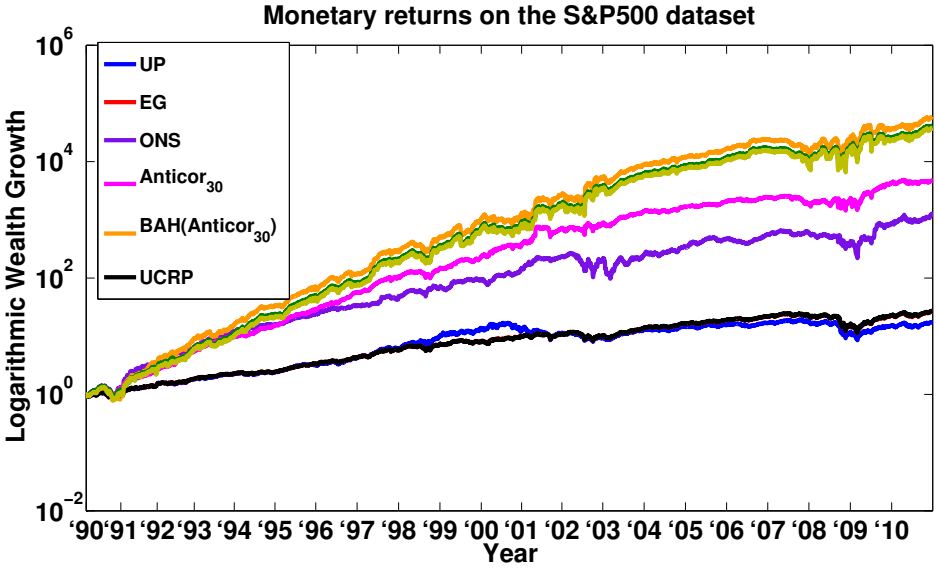


(b) Monetary returns on S&P500.

Figure 5.3: Monetary returns of the meta algorithms(MA_{EG}, MA_{ONS}) when Anticor_{30} is added to the pool of base algorithms. Anticor_{30} performs best and particularly MA_{EG} is able to track Anticor_{30} (best viewed in color).

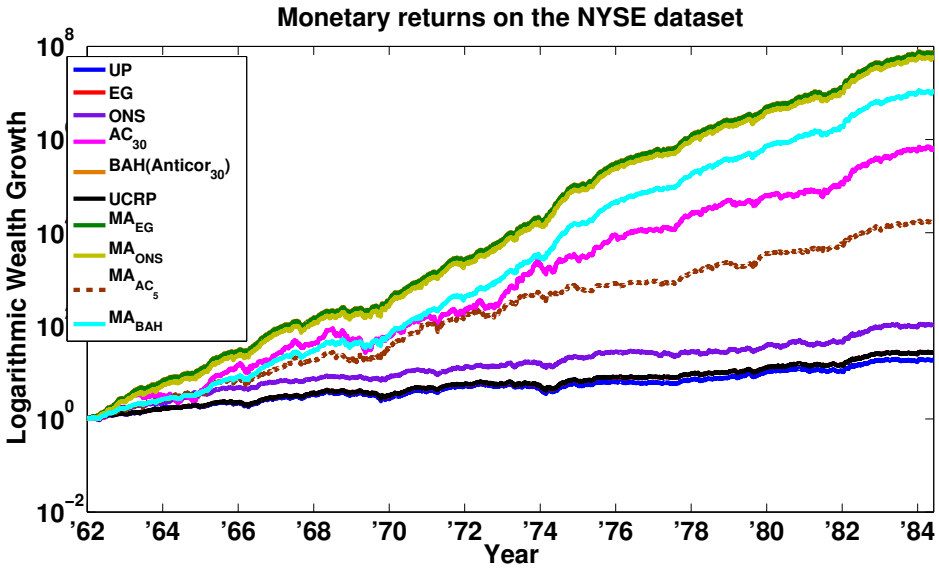


(a) Monetary returns on NYSE.

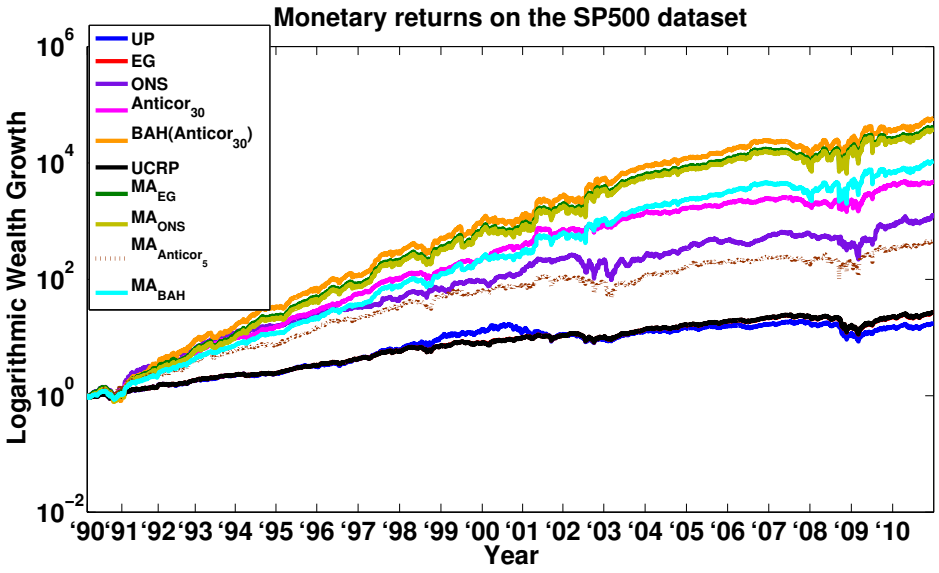


(b) Monetary returns on S&P500.

Figure 5.4: Monetary returns of the meta algorithms(MA_{EG}, MA_{ONS}). $\text{BAH}(\text{Anticor}_{30})$is added to the pool of base algorithms. The Meta Algorithms continue to track the best algorithm in the pool.(best viewed in color).



(a) Monetary returns on NYSE.



(b) Monetary returns on S&P500.

Figure 5.5: Monetary returns of the meta algorithms(MA_{EG}, MA_{ONS}, $\text{MA}_{Anticor}$, MA_{BAH}). MA_{EG} performs best while $\text{MA}_{Anticor}$ doesn't fare well.(best viewed in color).

We plotted the results with a window length of 5, as this version was observed to

perform reasonably well. It was observed that the performance of $\text{MA}_{Anticor}$ decreased as the window length was increased beyond 10 days. **APY and Volatility:** Table C.1 presents the monetary returns in dollars, APY and volatility of the universal and non-universal algorithms on the two datasets.

The wealth for the algorithms has been expressed as the final return on an initial investment of \$1. The values given for MA_{EG} and MA_{ONS} are with Mixed Pool 2 of base algorithms. The top three final returns appear in bold-face.

APY: The Annual Percentage Yield (APY) of the algorithms were calculated based on the following formula:

$$\textbf{APY} = \left[(\textbf{Final}/\textbf{Initial})^{\frac{1}{\mathbf{T_{years}}}} - 1\right] \times 100$$

where **Final** and **Initial** are the final return and initial investment respectively for $\mathbf{T_{years}}$. BAH(Anticor_{30}, MA_{EG} and MA_{ONS} have the top three (in the order mentioned) APY for both the datasets.

Volatility: Volatility of the Algorithms were calculated by taking the standard deviation of the sequence of daily wealth relatives (${x_t}^T r_t$) over $\mathbf{T_{years}}$. Anticor_{30} and $\text{BAH}(\text{Anticor}_{30})$ have more than twice the volatility of the base universal algorithms. MA_{EG} and MA_{ONS} have almost the same volatility as BAH(Anti-cor_{30}). This could be explained by the fact that the MAs try to track $\text{BAH}(\text{Anticor}_{30})$.

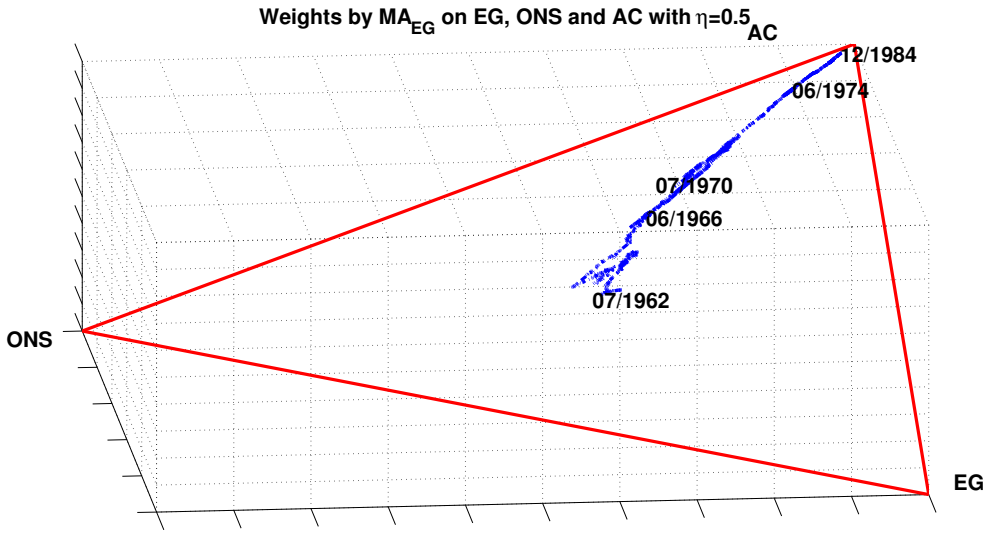
Figure 5.6 renders the path traced by MA_{EG} as a weighted combination of EG, ONS, and Anticor. Our experiments show that Anticor outperforms EG and ONS in terms of accumulated wealth by an overwhelming margin. Hence, we see that MA_{EG} converges towards the Anticor corner and the rate of convergence depends on the learning rate η. We also conducted experiments where we interchanged the weights of AC and ONS (switched corners) once every $t^{'}$ days. The value of $t^{'}$ was taken to be 500, 1000 and 2000 days for our experiments. Even after switching, MA_{EG} quickly learned that Anticor is the best performing strategy. This led MA_{EG} to automatically adjust its weight distribution over the base algoothms. MA_{EG} is sensitive to the volatility. Our experiments show that MA_{EG} might prefer ONS earlier on due to its low volatility (See Figure 5.6 (b)). But as the wealth accumulated by Anticor increases, it shifts its weight from ONS to Anticor.

Table 5.1: Monetary returns in dollars (per $1 investment), APY and volatility of universal and non-universal algorithms

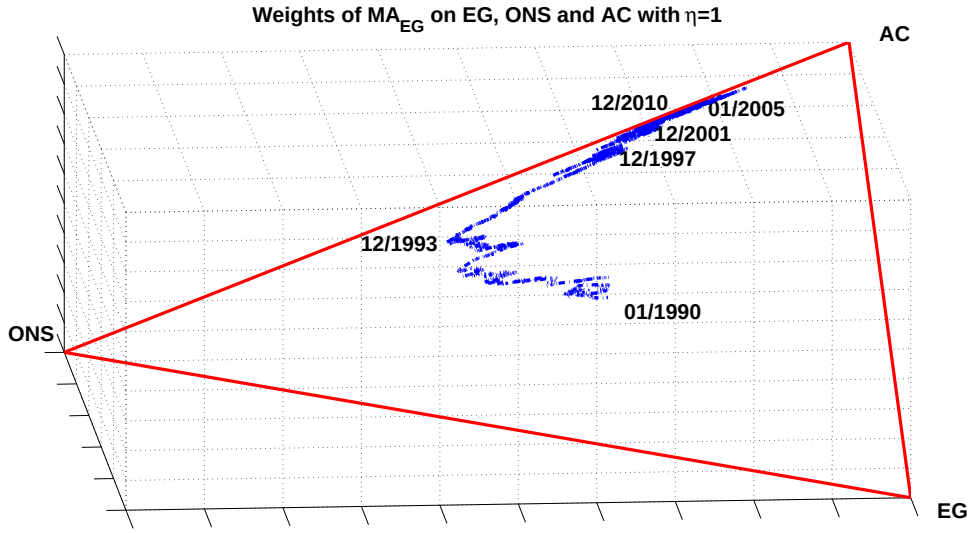
Algorithm		NYSE	SP500
UP	wealth	18.56	17.42
	APY	14.20	15.36
	volatility	0.0089	0.0139
EG	wealth	27.10	26.83
	APY	16.18	17.88
	volatility	0.0085	0.0116
ONS	wealth	109.17	1217.39
	APY	23.78	42.65
	volatility	0.0113	0.0201
Anticor_{30}	wealth	617754.61	4769.39
	APY	83.32	52.73
	volatility	0.0284	0.0191
$\text{BAH}(\text{Anticor}_{30})$	wealth	**77626129.46**	**58591.86**
	APY	**128.37**	**73.14**
	volatility	0.0195	0.03296
MA_{EG}	wealth	**68381688.71**	**41678.69**
	APY	**127.06**	**70.21**
	volatility	0.0194	0.0287
MA_{ONS}	wealth	**61119978.64**	**37667.41**
	APY	**125.90**	**69.36**
	volatility	0.0194	0.0286

5.5 Conclusions

In this chapter, we have presented the idea of designing new Meta Algorithms which work with a pool of base algorithms for optimization. We have shown that solutions from multiple iterative algorithms can be combined by Meta Algorithms to outperform the single best base algorithm. We demonstrate the efficacy of the Meta Algorithms in the domain of Online Portfolio Selection. Detailed experiments over the NYSE and S&P500 datasets show that the Meta Algorithms MA_{EG} and MA_{ONS} beat the universal algorithms in terms of empirical performance but are still competitive with the best CRP in hindsight.



(a) NYSE: weights of MA_{EG} with η=0.5.



(b) S&P500: weights of MA_{EG} with η=1.

Figure 5.6: Traces the weights maintained by MA_{EG} on the base algorithms EG, ONS and AC for NYSE and S&P500 with η values 0.5 and 1 respectively(best viewed in color).

Although, the Meta Algorithms have exceptional performance, they do not take into account the commission one has to pay while trading. This is also a shortcoming with the existing universal and non-universal portfolio selection algorithms [36] [61] [2] [18]. Most of these algorithms trade every stock every day which is not practical as one can incur huge amount of commission costs. As an extension of this chapter, we would like to recommend investigation of a sparse version of the meta algorithm that can take care of commissions and yet achieve good empirical performance. Also, the current models for on-line portfolio selection do not model risk. Modeling risk and taking account of volatility of stocks is an interesting direction, which we will explore in the very next chapter.

Chapter 6

Online Convex Optimization with Constraints

6.1 Motivation

Online convex optimization (OCO) and stochastic gradient descent (SGD) have emerged as powerful large scale optimization approaches for learning models from huge amounts of data [127, 57, 19, 20, 12, 13]. Instead of handling the full dataset at once, such approaches update the model incrementally based on one data point at a time. SGD is applicable for batch problems where the method makes multiple passes or epochs over the data. OCO is applicable to online settings, where the convex function to be optimized can itself change over time [127]. Such approaches have strong theoretical performance guarantees and have demonstrated remarkable empirical performance [79, 12].

For constrained optimization problems, which forms the basis for learning most widely used models in data mining, the literature on OCO (and SGD) is not as mature. The literature suggests using projected gradient descent (or Newton) updates, where one takes a gradient step based on the current data point and projects the solution to the feasible set. In recent literature, theoretical guarantees have been established for such updates [127, 57, 72]. In practice, however, performing a projection onto a given feasible set is an excrutiatingly time consuming process. Standard approaches, such as Bregman's algorithm which perform cyclic projections onto constituent convex

sets followed by nonlinear corrections [29], are inherently sequential and hence can be rather slow in practice especially if the number of constraints is large.

In this chapter, we consider online quadratically constrained convex optimization (QCCO) problems. The feasible set F for such problems is determined by a collection of quadratic and linear constraints. In other words, the feasible set is the intersection of ellipsoids, hyperplanes, and half-spaces. As in online convex optimization, learning proceeds in rounds. In round t, the algorithm picks $\mathbf{x}_t \in F$, then nature reveals a convex function f_t, and the algorithm incurs a cost of $f_t(\mathbf{x}_t)$. We illustrate that for any sequence of convex functions, regret guarantees can be obtained by solving a suitable quadratically constrained quadratic program (QCQP) at each step. We propose an efficient primal-dual algorithm for solving QCQPs based on the alternating directions method [22]. The advantage of the proposed approach that it can handle the constraints in parallel using auxiliary variables, so the number of constraints is not an issue in itself.

A key practical motivation behind considering online QCCO problems is portfolio selection under risk constraints. While considerable work has been done in online portfolio selection over the past two decades [38, 61, 2, 36], almost all of them choose not to model the risk associated with the portfolios. Since risk models play a central role in finance [91, 109], online algorithms which disregard risk have had only limited impact, if any, in practice.

Using the general online QCCO framework, we propose Risk Adjusted Meta Portfolio (RAMP), an online portfolio selection algorithm which attempts to maximize wealth under risk constraints. There is extensive literature in finance dwelling on the balance between risk and return and methods for doing the same [17] chapter 4. Our work is motivated by Markowitz's mean-variance portfolio theory [91] which has had profound influence in economic modeling in finance domain [112]. Despite the model's theoretical success, the instable nature of the posed optimization problem has hindered its practical application. [43] found that none of the existing work based on Markowitz's framework is able to outperform the naive equally weighted portfolio.

We model risk in terms of the variance of the portfolio, which is incorporated as a quadratic constraint in the optimization framework. By posing risk as a constraint as opposed to minimizing it as suggested by Markowitz's framework, we also avoid the empirical instability that is inherent of that framework. Further, instead of constructing

the portfolio from scratch, we focus on building a meta-portfolio by suitably combining portfolios recommended by a set of existing algorithms [40, 61, 2, 36]. While such algorithms may not have any control on the risk of the portfolios suggested, the meta-portfolio we infer is gauranteed to satisfy the pre-specified risk constraints, possibly at the cost of accumulating less wealth.

Our experiments on the NYSE and S&P500 dataset show that for a fixed risk, RAMP always outperforms the existing online portfolio selection algorithms [61, 2, 36]. For higher risk, RAMP competes with heuristics like Anticor and Meta Portfolios like MP_{GD} and multiplies its starting wealth by 10^7 times for NYSE (10^6 for S&P500) but still has lower risk than any of the former.

The rest of the chapter is organized as follows. Section 6.2 introduces online QCCO, along with regret analysis and the primal-dual algorithm. Section 6.3 introduces RAMP as a special case of the online QCCO framework. We present details of the extensive experiments we have performed in section 6.4, and conclude in section 6.5.

6.2 Online QCQO

Consider the following general form of a *Quadratically Constrained Convex Optimization* (QCCO) problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \frac{1}{2}\mathbf{x}^T P_i x + q_i^T x + r_i \leq 0 \ , \quad i = 1, \ldots, m \\ & \mathbf{a}_j^T \mathbf{x} = b_j \ , \quad j = 1, \ldots, k \ , \end{aligned} \tag{6.1}$$

where $P_i \in \mathbb{R}^{n \times n}, i = 1, \ldots, m$ are positive definite matrices. A special case of In QCCO problems, one minimizes a (strictly) convex function over a feasible set K determined by the intersection of ellipsoids (when $P_i \succ 0$) and hyperplanes. Quadratically Constrained Quadratic Programs (QCQPs) form an important special case of QCCOs [55, 23]. More widely studied special cases include quadratic programs (QPs) and linear programs (LPs) which are somewhat simpler since the constraint set does not include any ellipsoids.

We focus on an online setting for QCCOs where the convex objective function keeps changing over time while the feasible set stays fixed. The optimization proceeds in

round where in round t, the algorithm has to first pick $\mathbf{x}_t \in K$; then, nature reveals f_t and the value of the objective function $f_t(x_t)$ is determined. The strictly convex function f_t chosen by nature can be arbitrary, even adversarial, as long as it satisfies some minimal regularity conditions, which we discuss shortly. The goal is to design an algorithm which picks the sequence of $\mathbf{x}_t$ such that the cumulative objective function value of the adaptive algorithm is competitive with that of the single best $x \in F$ chosen in hindsight [32, 127]. More precisely, we want a sequence $\mathbf{x}_t$ such that

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) \leq \min_{x \in F} \sum_{t=1}^{T} f_t(\mathbf{x}) + o(T) \ . \tag{6.2}$$

With the regret defined as $R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{x \in F} \sum_{t=1}^{T} f_t(\mathbf{x})$, we simply want the regret to be sublinear [57, 72].

The above formulation can be viewed as a special case of the online convex optimization (OCO) framework [127]. While OCO has been widely studied over the past few years from a theoretical perspective, the main algorithm analyzed in this context is projected gradient descent [127, 13], which needs to perform a projection to the given convex feasible set at every step. While such a projection may be efficiently doable for simple feasible sets, such as the unit L_1 or L_2 ball [46] the projection will undoubtedly be the most time consuming step for more complex feasible sets, including the one considered in QCCO in (6.1). Assuming the feasible set F to be an intersection of simpler convex sets, Bregman's algorithm can provide a fairly general purpose approach for solving such projection problems by cyclic projections into individual convex sets. However, Bregman's algorithm can become rather computationally restrictive especially when the feasible set is defined as the intersection between a number of nonlinear constraints. : (i) For anything other than linear equality constraints, each projection has to be followed by a correction by solving a nonlinear equation which can be time consuming [29], and (ii) The algorithm is inherently sequential so that only one constraint can be worked on at any given time. In particular, while online QCQO problems can in principle be solved by projected gradient methods, one has to perform sequential cyclic projections on ellipsoids in each step [29], with suitable corrections by solving nonlinear equations, leading to a rather inefficient algorithm.

In this chapter, we propose a primal-dual approach based on the alternating directions method of multipliers (ADMMs) [22]. Our ADMM algorithm can work with the

nonlinear ellipsoidal constraints in parallel. Further, the only nonlinear aspect of the solution is to essentially find the zero-crossing of a monotonic function in a bounded region, which can be done efficiently using standard methods, such as bisection search or Newton-Raphson [28].

We first show that the update for projected gradient descent method, for which regret guarantees already exist [13, 127], can be viewed as the solution of a QCQP, where the quadratic objective gets suitably determined at every step and the quadratic constraints are the same as the original problem. Then, we present an efficient ADMM algorithm to solve general QCQPs, which can be readily applied to obtain $\mathbf{x}_{t+1}$ at each stage of the online algorithm.

6.2.1 Regret Analysis with Modified QCQP

The projected gradient descent update for OCO with a sequence of strongly convex functions f_t is given by:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in F}{\operatorname{argmin}} \left\{ \mathbf{x}^T \nabla f_t(\mathbf{x}_t) + \frac{1}{2\eta_t} \|\mathbf{x} - \mathbf{x}_t\|^2 \right\} . \tag{6.3}$$

Assuming that the functions are β-strongly convex and gradients are bounded, i.e., $\|\nabla f_t\| \leq G$, with step sizes $\eta_t = \frac{1}{\beta t}$, existing approaches to OCO regret analysis yields:

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in F} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq \frac{G^2}{2\beta}(1 + \log T) . \tag{6.4}$$

For the update (6.3), when the convex set F is determined by an intersection of quadratic constraints as in (6.1), the optimization problem is a QCQP. Next we outline an efficient algorithm for solving general QCQPs which can be used for computing $\mathbf{x}_{t+1}$ in the context of online QCCO.

6.2.2 ADMM Algorithm for QCCOs

Following (6.1) and (6.3), the optimization problem to be solved at each stage of online QCCO can be posed as a QCQP given by:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^T P\mathbf{x} + \mathbf{q}^T\mathbf{x} + r \\ \text{s.t.} \quad & \frac{1}{2}\mathbf{x}^T P_i\mathbf{x} + q_i^T\mathbf{x} + r_i \leq 0 \ , \quad i = 1, \ldots, m \\ & a_j^T\mathbf{x} = b_j \ , \quad j = 1, \ldots, k \ . \end{aligned} \tag{6.5}$$

For online QCCO, the parameters in the objective function are given by $P = \frac{1}{\eta_t}\mathbb{I}$, $q = \nabla f_t(x_t) + \mathbf{x}_t$, and $r = \frac{1}{2\eta_t}\|\mathbf{x}_t\|^2$. We present our analysis in this section for general $\{P, \mathbf{q}, r\}$ to illustrate the fact that the proposed ADMM algorithm can be used to solve any QCQPs efficiently.

We start by introducing a set of auxiliary variables—$\mathbf{y}_i$ for each ellipsoidal constraint and $\mathbf{z}_i$ for each linear constraint with the additional constraint set $\mathbf{y}_i = \mathbf{x}$ and $\mathbf{z}_j = \mathbf{x}$. Further, we rewrite the quadratic objective function as a sum of quadratic objectives, one corresponding to each $\mathbf{y}_i$. The modified problem stated below is exactly equivalent to the original problem in (6.5):

$$\begin{aligned} \min_{\mathbf{x},\mathbf{y}_i,\mathbf{z}_j} \quad & \frac{1}{2m}\sum_{i=1}^{m}\left\{\mathbf{y}_i^T P\mathbf{y}_i + \mathbf{q}^T\mathbf{y}_i + r\right\} \\ \text{s.t.} \quad & \frac{1}{2}\mathbf{y}_i^T P_i\mathbf{y}_i + q_i^T\mathbf{y}_i + r_i \leq 0 \ , \quad i = 1, \ldots, m \\ & a_j^T\mathbf{z}_j = b_j \ , \quad j = 1, \ldots, k \\ & \mathbf{x} = \mathbf{y}_i \ , \quad i = 1, \ldots, m \\ & \mathbf{x} = \mathbf{z}_j \ , \quad j = 1, \ldots, k \ . \end{aligned} \tag{6.6}$$

Disregarding both sets of original constraints, the augmented Lagrangian for the problem, based on the equality constraints $\mathbf{y}_i = \mathbf{x}$ and $\mathbf{z}_j = \mathbf{x}$, is given by

$$\begin{aligned} L_\beta(\mathbf{x}, \mathbf{y}_i, \mathbf{z}_j, \sigma_i, \gamma_j) = \frac{1}{2m} \sum_{i=1}^{m} \mathbf{y}_i^T P \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + r \\ + \sum_{i=1}^{m} \sigma_i (\mathbf{y}_i - \mathbf{x}) + \sum_{j=1}^{k} \gamma_j (\mathbf{z}_j - \mathbf{x}) \\ + \frac{\beta}{2} \left\{ \sum_{i=1}^{m} \|\mathbf{y}_i - \mathbf{x}\|^2 + \sum_{j=1}^{k} \|\mathbf{z}_j - \mathbf{x}\|^2 \right\} . \end{aligned} \tag{6.7}$$

The quadratic constraints cannot be directly handled by the ADMM. Hence, we ensure the constraint on each $\mathbf{y}_i$ while performing the corresponding iterative update in the ADMM. Similarly, the individual linear constraints ${a^{(j)}}^T \mathbf{z}_j = b_j$ are satisfied while performing the ADMM update on $\mathbf{z}_j$. In particular, the updates for the primal variables $\{\mathbf{y}_i, \mathbf{z}_j, \mathbf{x}\}$ and the dual variables $\{\sigma_i, \gamma_j\}$ are given in Algorithm 7.

The update for the dual variables in (6.11) and (6.12) are already in closed form. Note that the update for $\mathbf{x}^{(t+1)}$ in (6.10) can be obtained in closed form:

$$\mathbf{x}^{(t+1)} = \frac{1}{m+k} \left\{ \sum_{i=1}^{m} \left(\mathbf{y}_i^{(t+1)} + \frac{\sigma_i^{(t)}}{\beta} \right) + \sum_{j=1}^{k} \left(\mathbf{z}_j^{(t+1)} + \frac{\gamma_j^{(t)}}{\beta} \right) \right\}. \tag{6.13}$$

Further, the update for $\mathbf{z}_j^{(t+1)}$ in (6.9) can also be obtained in closed form by projection the unconstrained solution onto the hyperplane constraint:

$$\mathbf{z}_j^{(t+1)} = \left(\mathbf{x}^{(t)} - \frac{\gamma_j^{(t)}}{\beta} \right) + \frac{1}{\|\mathbf{a}_j\|^2} \left(b_j - \mathbf{a}_j^T \left(\mathbf{x}^{(t)} - \frac{\gamma_j^{(t)}}{\beta} \right) \right), j = 1, \ldots, k \, . \tag{6.14}$$

y-update: The update for $\mathbf{y}_i^{(t+1)}$ in (6.8) is a QCQP with only one quadratic constraint. Denoting $\mathbf{y} = \mathbf{y}_i$, the problem is given by

$$\min_{\mathbf{y}} \ \frac{1}{2} \mathbf{y}^T \Gamma \mathbf{y} + g^T \mathbf{y} + d_0 \quad \text{s.t.} \frac{1}{2} \mathbf{y}^T P_i \mathbf{y} + q_i^T \mathbf{y} + r_i \leq 0 \tag{6.15}$$

where $\Gamma = \frac{1}{m} P + \beta \mathbb{I}$, $g = \frac{1}{m} q + \beta \mathbf{x}^{(t)} + \sigma_i^{(t)} \mathbf{e}$ where $\mathbf{e}$ is the all ones vector, and $d_0 = \frac{\beta}{2} \|\mathbf{x}^{(t)}\|^2 - \sigma_i^{(t)} \mathbf{x}^{(t)}$. Let

$$\mathbf{v} = \frac{1}{\rho_i} (P_i^{1/2} \mathbf{y} + P_i^{-1/2} \mathbf{q}_i) \, , \tag{6.16}$$

Algorithm 7 ADMM updates for QCQP

1: Input $P, q, \{P_i, q_i, r_i\}_1^m, \{a_j, b_j\}_1^k$

2: ADMM iterations

$$\mathbf{y}_i^{(t+1)} = \underset{\mathbf{y}_i:\frac{1}{2}\mathbf{y}_i^T P_i \mathbf{y}_i + q_i^T \mathbf{y}_i + r_i \leq 0}{\operatorname{argmin}} \left\{ \frac{1}{2m}\mathbf{y}_i^T P \mathbf{y}_i + \mathbf{q}^T \mathbf{y}_i + \sigma_i^{(t)}(\mathbf{y}_i - \mathbf{x}^{(t)}) + \frac{\beta}{2}\|\mathbf{y}_i - \mathbf{x}^{(t)}\|^2 \right\} , \; i = 1, \ldots, m \tag{6.8}$$

$$\mathbf{z}_j^{(t+1)} = \underset{\mathbf{z}_j:a_j^T \mathbf{z}_j = b_j}{\operatorname{argmin}} \left\{ \gamma_j^{(t)}(\mathbf{z}_j - \mathbf{x}^{(t)}) + \frac{\beta}{2}\|\mathbf{z}_j - \mathbf{x}^{(t)}\|^2 \right\}, j = 1, \ldots, k \tag{6.9}$$

$$\mathbf{x}^{(t+1)} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \sum_{i=1}^m \sigma_i^{(t)}(\mathbf{y}_i^{(t+1)} - \mathbf{x}) + \sum_{j=1}^k \gamma_j^{(t)}(\mathbf{z}_j^{(t+1)} - \mathbf{x}) + \frac{\beta}{2}\left\{ \sum_{i=1}^m \|\mathbf{y}_i^{(t+1)} - \mathbf{x}\|^2 + \sum_{j=1}^k \|\mathbf{z}_j^{(t+1)} - \mathbf{x}\|^2 \right\} \right\} \tag{6.10}$$

$$\sigma_i^{(t+1)} = \sigma_i^{(t)} + (\mathbf{y}_i^{(t+1)} - \mathbf{x}^{(t+1)}) \;\; i = 1, \ldots, m \tag{6.11}$$

$$\gamma_j^{(t+1)} = \gamma_j^{(t)} + (\mathbf{z}_j^{(t+1)} - \mathbf{x}^{(t+1)}) \;\; j = 1, \ldots, k \; . \tag{6.12}$$

3: Continue until **Stopping Criteria** is satisfied

where $\rho_i = (2\mathbf{q}_i^T P_i^{-1} \mathbf{q}_i)^{1/2}$. With this change of variables, the original problem can be written as:

$$\min_{\mathbf{v}} \; \frac{1}{2}\mathbf{v}^T A \mathbf{v} + h^T \mathbf{v} + d \quad \text{s.t.} \mathbf{v}^T \mathbf{v} \leq 1 \; , \tag{6.17}$$

where $A = \rho_i^2 P_i^{-1/2} \Gamma P_i^{-1/2}$, $h = b_i P_i^{-1/2} g - P_i^{-1/2} \Gamma P_i^{-1} q_i$, and $d = d_0 - \frac{1}{2} q_i^T P_i^{-1} \Gamma P_{-i}^{-1} q_i - g^T P_i^{-1} q_i$. The Lagrangian for (6.17) is given by

$$L(\mathbf{v}, \lambda) = \frac{1}{2}\mathbf{v}^T A \mathbf{v} + h^T \mathbf{v} + d + \lambda(\mathbf{v}^T \mathbf{v} - 1) \; , \tag{6.18}$$

where $\lambda \geq 0$. Setting gradient w.r.t. $\mathbf{v}$ to zero, we obtain

$$\mathbf{v} = -(A + \lambda \mathbb{I})^{-1} h \; . \tag{6.19}$$

From the complimentary slackness condition, we have $\lambda(\mathbf{v}^T \mathbf{v} - 1) = 0 \Rightarrow \lambda = 0$ or $\mathbf{v}^T \mathbf{v} - 1 = 0$. For the latter condition to be true, from (6.19) we must have

$$f(\lambda) = h^T (A + \lambda \mathbb{I})^{-2} h - 1 = 0 \; . \tag{6.20}$$

Let $\alpha_h, h = 1, \ldots, p$ denote the (positive) eigenvalues of A. Then, the eigenvalues of $(A + \lambda\mathbb{I})$ are $(\alpha_h + \lambda), h = 1, \ldots, p$, which are also positive for $\lambda \geq 0$. If $\phi_h, h = 1, \ldots, p$ denote the eigenvectors of $(A + \lambda\mathbb{I})$, then

$$f(\lambda) = h^T(A + \lambda\mathbb{I})^{-2}h - 1 = \sum_{h=1}^{p} \frac{\|h^T\phi_h\|^2}{(\alpha_h + \lambda)^2} - 1 \ .$$

It is easy to verify that $f(\lambda)$ is a decreasing function of λ for $\lambda \geq 0$, e.g., the gradient is negative. Thus, if $f(0) \geq 0$, then $f(\lambda)$ has a zero-crossing in $\lambda \geq 0$ which can be found efficiently using a root finding method, such as bisection search or Newton-Raphson [28]. If $f(0) < 0$, then $\mathbf{v}^T\mathbf{v} - 1 = 0$ cannot be satisfied implying $\lambda = 0$ from the complimentary slackness condition. Then, λ yields the optimal $\mathbf{v}$ from (6.19), which from (6.16) gives the solution to the original QCQP in (6.15). Thus, (6.8) can also be solved efficiently since the only computation involved is root finding of a monotonic decreasing function.

We highlight two additional advantages of the ADMM approach considered. The updates for the $\mathbf{y}_i$ corresponding to different quadratic constraints for $i = 1, \ldots, m$ can be done in parallel, since there are no interaction terms. This is in sharp contrast with a projected gradient approach based on Bregman's algorithm which uses cyclic projections where at any point only one constraint can be considered. Further, if the original problem has linear inequality constraints of the form $C\mathbf{x} \leq d$, they can be handled similar to the equality constraints since the projection to a single half-space constraint is also closed form [23].

6.2.3 Online QCCO with Varying Constraints

We consider a generalization of the QCCO problem where the constraints themselves can change over time. In particular, we assume that the quadratic constraints have parameters $\{P_i^t, \mathbf{q}_i^t, r_i^t\}$ which are also chosen by nature and revealed before the algorithm picks $\mathbf{x}_t$. As before, the convex function f_t is revealed by nature after $\mathbf{x}_t$ is chosen. If F_t denotes the constraint set in step t, let $F_{(T)} = \cap_{t=1}^{T} F_t$. We assume that $F_{(T)}$, the intersection of the quadratic constraints across all time steps, is non-empty. The QCQP to be solved at each step of the online QCCO is similar to (6.3) with the constraint set

being F_{t+1}, i.e.,

$$\mathbf{x}_{t+1} = \underset{x \in F_{t+1}}{\operatorname{argmin}} \left\{ \mathbf{x}^T \nabla f_t(\mathbf{x}_t) + \frac{1}{2\eta_t} \|\mathbf{x} - \mathbf{x}_t\|^2 \right\} . \tag{6.21}$$

The above updates can be shown to have logarithmic regret w.r.t. the best solution $\mathbf{x} \in F_{(T)}$ (proof in appendix).

Theorem 9. *The projected gradient descent algorithm with step sizes $\eta_t = \frac{1}{\alpha t}$ achieves the following regret:*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in F_{(T)}} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq \frac{G^2}{\alpha\beta}(1 + \log T) . \tag{6.22}$$

Proof: The proof follows from a modification of the standard argument for projected gradient descent [127]. Let $\mathbf{x}^* = \operatorname{argmin}_{x \in F_{(T)}} \sum_{t=1}^{T} f_t(\mathbf{x}_t)$. Since f_t is α-strong convex, we have

$$\begin{aligned} f_t(x^*) &\geq f_t(x_t) + \nabla f_t(\mathbf{x}_t)^T (x^* - \mathbf{x}_t) + \frac{\beta}{2} \|\mathbf{x}^* - \mathbf{x}_t\|^2 \\ 2(f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*)) &\leq 2\nabla f_t(\mathbf{x}_t)^T (\mathbf{x}_t - \mathbf{x}^*) - \beta \|\mathbf{x}^* - \mathbf{x}_t\|^2 . \end{aligned} \tag{6.23}$$

Since $\mathbf{x}_{t+1}$ is a projection of $(\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t))$ onto F_{t+1}, and $\mathbf{x}^* \in F_{(T)} \subseteq F_t$, from the generalized Pythagoras theorem [90],

$$\begin{aligned} \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2 &\leq \|\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t) - \mathbf{x}^*\|^2 \\ &= \|\mathbf{x}_t - \mathbf{x}^*\|^2 + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2 \\ &\quad - 2\eta_t \nabla f_t(\mathbf{x}_t)(\mathbf{x}_t - \mathbf{x}^*) \end{aligned}$$

which implies

$$2\nabla f_t(\mathbf{x}_t)^T (\mathbf{x}_t - \mathbf{x}^*) \leq \frac{1}{\eta_t} \left\{ \|\mathbf{x}_t - \mathbf{x}^*\|^2 - \|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2 \right\} + \eta_t G^2. \tag{6.24}$$

Plugging (6.24) in (6.23), summing over all t, and using the fact that $\eta_t = \frac{1}{\beta t}$ yields the regret bound (6.22). □

6.3 Risk Adjusted Meta Portfolio

An important application of OCO, online portfolio selection [36, 61, 32, 2, 40] has attracted increasing interest in machine learning and AI communities over the past

two decades. In a well defined technical sense, online portfolio selection algorithms are guaranteed to perform competitively with certain families of adaptive portfolios even in an adversarial market without making any statistical assumptions regarding the movement of the stocks. However, none of the existing algorithms model risk in any form. This omission of risk is particularly glaring since volatility and uncertainty measures are given equal footing as the returns themselves in finance.

6.3.1 Risk in Portfolio Selection

The financial literature on the proposed metrics for risk is extensive and beyond the scope of this chapter [27, 109, 110, 107, 71, 68, 95, 8, 103]. Markowitz was the first to formalize protfolio risk [91] and in his seminal work proposed a portfolio's risk is equal to the *variance* of the portfolio's returns.

We adopt Markowitz's framework and use variance of a portfolio (a convex measure), to compute its risk and avoid any theoretical and computation challenges that we might face in modeling other risk measures [98],We pose the problem as one of maximizing returns using the risk as constraint. The constraint considers relative risk w.r.t. a given strategy (or benchmark) such as the uniform portfolio u over the stocks (or standard market index) . If Σ_t is the (estimated) covariance among returns in time t, for a portfolio p_t, the risk constraint is given by:

$$p_t^T \Sigma_t p_t \leq \alpha_{\mathbf{risk}}\ u^T \Sigma_t u\ , \quad p_t \in \triangle_n \tag{6.25}$$

where $\alpha_{\mathbf{risk}} \in \mathbb{R}_{++}$ is the relative risk factor which determines the allowable risk w.r.t. the uniform portfolio in time t. The advantage of the relative formulation, as opposed to an absolute formulation, is that the natural variability of the market is already taken into account. Note that the above quadratic constraint varies over time. This quadratic constraint along with the simplex constraint in (6.25), sets up the constraints of our Online QCCO framework. However, given the framework, other definitions of risk can be considered as long as they are convex functions of the portfolio, or can be relaxed/approximated by a convex function.

6.3.2 Online Risk Adjusted Meta Portfolio

Algorithm 8 RAMP Algorithm for Portfolio Selection

Input η, β
Initialize $w_{1,g} = \frac{1}{G}, g = 1, \ldots, G; w_0 = w_1; S_0 = 1$
For $t = 1, \ldots, T$
  Receive $\mathbf{r}_t$, the vector of price relatives and X_{t+1}, the matrix of base portfolios
  Compute cumulative wealth: $S_t = S_{t-1} \times ((X_t w_t)^T r_t)$
  Initialize $w, y, \nu \in 0^n, l = 0$
  Update portfolio:
  ADMM iterations

$$w_{t+1}^{(k+1)} = \min_{w^T X_{t+1}^T \Sigma_{t+1} X_{t+1} w \leq b_{t+1}} - r_t^T X_t w / r_t^T X_t w_t + \frac{1}{2\eta_t}\|w - w_t\|_2^2 + \frac{\beta}{2}\|w - y^{(k)} + \nu^{(k)}\|_2^2 \tag{6.26}$$

$$y^{(k+1)} = \min_{y \in \Delta_n} \|y - (w^{(k+1)} + \nu^{(k)})\|_2^2 \tag{6.27}$$

$$\nu^{(k+1)} = \nu^k + (w^{(k+1)} - y^{(k+1)}) \ . \tag{6.28}$$

  Continue until **Stopping Criteria** is satisfied
  Compute true γ_{risk}.
End For

Recent work on meta portfolios (MPs), which adaptively combine different online portfolio selection algorithms, have demonstrated strong empirical performance in terms of the overall wealth return, while preserving theoretical guarantees [40]. In order to harness the strength of such meta portfolios, we pose our portfolio selection problem as Risk Adjusted Meta Portfolio selection. We formalize our model below.

We consider a stock market consisting of n stocks $\{s_1, \ldots, s_n\}$ over a span of T periods. For ease of exposition, we will consider a period to be a day, but the analysis presented holds for any valid definition of a 'period', such as an hour or a month. Let $r_i(t) = \frac{\text{closing price of } s_i \text{on day } t}{\text{opening price of } s_1 \text{on day } t}$ denote the *price relative* of stock s_i in day t, i.e., the multiplicative factor by which the price of s_i changes in day t. We assume, $r_t(i) > 0 \ \forall \ i, t$. Let $\mathbf{r}_t = \langle r_t(1), \ldots, r_t(n) \rangle$ denote the vector of price relatives for day t. A portfolio $\mathbf{p}_t = \langle p_t(1), \ldots, p_t(n) \rangle$ on day t can be viewed as a probability distribution over the stocks that prescribes investing $p_t(i)$ fraction of the current wealth in stock $r_t(i)$. Note that the portfolio $\mathbf{p}_t$ has to be decided before knowing $\mathbf{r}_t$ which will be revealed only at

the end of the day.

On day t, let $\Sigma_{t+1} \in \mathbb{R}^{n\times n}$ be the (estimated) covariance matrix of price relatives, $X_{t+1} = [x_{t+1,1} \cdots x_{t+1,G}] \in \mathbb{R}^{n\times G}$ be the matrix of base portfolios from k base algorithms, and w_t be the probability distribution on the algorithm iterates on day t, yielding a meta portfolio $p_t = X_t w_t$. Further, the covariance Σ_{t+1} is estimated using a smoothed kernel estimator [126].

Then based on (6.25), the constraint set at time $(t+1)$ is given by $F_{t+1} = \{w \in \mathbb{R}^G | w^T X_{t+1}^T \Sigma_{t+1} X_{t+1} w \leq \alpha_{\mathbf{risk}} u^T \Sigma_{t+1} u, w^T e = 1, w_i \geq 0\}$. Further, the intersection of all such constraints is denoted as $F_{(T)} = \cap_{t=1}^T F_t$. Note that $F_{(T)}$ is non-empty by design since the uniform portfolio $u \in F_t, \forall t$. The goal is then to select a sequence of weights $w_{t+1} \in F_{t+1}$ over portfolios X_{t+1} from base algorithms such that in terms of the logarithmic wealth accumulated over time, the cumulative regret w.r.t. any $w \in F_{(T)}$ is sublinear, i.e.,

$$\max_{w\in F_{(T)}} \sum_{t=1}^{T} \log(r_t^T X_t w) - \sum_{t=1}^{T} \log(r_t^T X_t w_t) \leq o(T) \ . \tag{6.29}$$

6.3.2.1 RAMP Algorithm

We now introduce our Risk Adjusted Meta Portfolio (RAMP) algorithm. The problem of picking the sequence of w_t is a special case of the online QCCO framework in Section 6.2 with convex function $f_t(w) = -\log(r_t^T X_t w)$. Following (6.21), the update is given by

$$w_{t+1} = \underset{\mathbf{x}\in F_t}{} \left\{ -\frac{r_t^T X_t w}{r_t^T X_t w_t} + \frac{1}{2\eta_t}\|w - w_t\|^2 \right\} \ . \tag{6.30}$$

By introducing auxiliary variables for the linear constraints in F_{t+1}, and denoting $\alpha_{\mathbf{risk}} u^T \Sigma_{t+1} u = b_{t+1}$, the minimization problem can be equivalently written as:

$$\begin{aligned} \min_{w,y} \ & -\frac{r_t^T X_t w}{r_t^T X_t w_t} + \frac{1}{2\eta_t}\|w - w_t\|^2 \\ \text{s.t. } & w^T X_{t+1}^T \Sigma_{t+1} X_{t+1} w \leq b_{t+1}, \quad y^T e = 1, y_i \geq 0, \quad w = y \ . \end{aligned} \tag{6.31}$$

Ignoring all constraints from F_{t+1} for now, with $\nu = \frac{1}{\beta}\lambda$ where λ is the Lagrangian multiplier for the constraint $w = y$, the augmented Lagrangian is given by

$$L(w, y, u) = -\frac{r_t^T X_t w}{r_t^T X_t w_t} + \frac{1}{2\eta_t}\|w - w_t\|^2 \frac{\beta}{2}\|w - y + \nu\|^2 \ . \tag{6.32}$$

The ADMM updates are given as follows:

$$w^{k+1} = \min_{w^T X_{t+1}^T \Sigma_{t+1} X_{t+1} w \le b_{t+1}} - r_t^T X_t w / r_t^T X_t w_t + \frac{1}{2\eta_t}\|w - w_t\|_2^2 + \frac{\beta}{2}\|w - y^k + \nu^k\|_2^2 \tag{6.33}$$

$$y^{k+1} = \min_{y \in \Delta_n} \|y - (w^{k+1} + \nu^k)\|_2^2 \tag{6.34}$$

$$\nu^{k+1} = \nu^k + (w^{k+1} - y^{k+1}) \ . \tag{6.35}$$

The update for w^{k+1} solves a QCQP with a single quadratic constraint, which was discussed in Section 6.2. The update for y^{k+1} does an Euclidean projection into the probability simplex, for which there are efficient algorithms [46]. Thus, the updates are efficient, and are run iteratively till convergence. To keep track of progress, we compute the primal and dual residuals respectively given by $r^{k+1} = w^{k+1} - y^{k+1}$, and $s^{k+1} = \beta(y^{k+1} - y^k)$. Here b_{t+1} is α times the risk of a uniform portfolio $\mathbf{u}$, i.e. $b_{t+1} = \alpha \times \mathbf{u}^T \Sigma_{t+1} \mathbf{u}$ We can write the w-update step as

$$\min_{(X_t w)^T \Sigma_{t+1} (X_t w) \le b_{t+1}} \frac{1}{2} w^T P w + q^T w \ . \tag{6.36}$$

where $P = (\frac{1}{\eta} + \beta)I$ and $q = (-\frac{r_t^T X_t}{r_t^T X_t w_t} - \rho w_0 - \beta y^k + \beta \nu^k)$.

Transformation to QCCO: Putting $Z = X_t^T \Sigma X_t$ and $v = \frac{1}{\sqrt{b_{t+1}}} Z^{\frac{1}{2}} w$, we have

$$\min_{v^T v \le 1} \frac{1}{2} v^T \hat{P} v + \hat{q}^T v \ . \tag{6.37}$$

where $\hat{P} = b_{t+1}(Z^{-\frac{1}{2}} P Z^{-\frac{1}{2}})$ and $\hat{q} = q^T Z^{-\frac{1}{2}} \sqrt{b_{t+1}}$

Primal and Dual residual calculation

$$r^{k+1} = w^{k+1} - y^{k+1}$$

$$s^{k+1} = \beta(y^{k+1} - y^k)$$

Stopping Criteria: $\|r^{k+1}\| \le \epsilon^{pri}$ and $\|s^{k+1}\| \le \epsilon^{dual}$

$\epsilon^{pri} = \sqrt{k}\epsilon^{abs} + \epsilon^{rel} \max\{\|w^{k+1}\|, \|y^{k+1}\|\}$

$\epsilon^{dual} = \sqrt{k}\epsilon^{abs} + \epsilon^{rel}\|\lambda^{k+1}\|$

For our experiments, we used $\epsilon^{pri} = \epsilon^{dual} \simeq 10^{-4}$. It is import ant to note that the risk adjusted versions of the base algorithms [61, 2, 36], can each be be posed as a special case of the online

6.4 Experimental Results

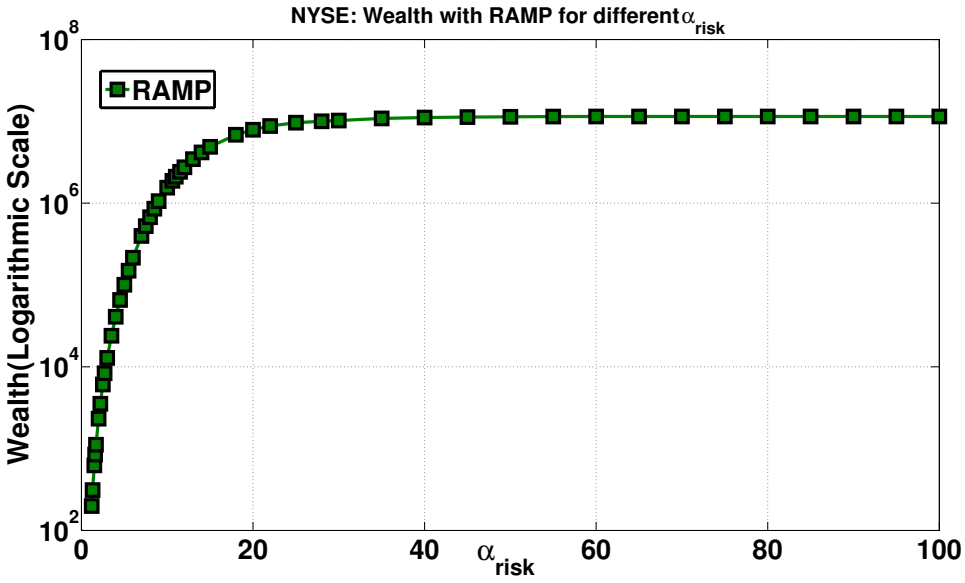


Figure 6.1: Monetary returns of RAMP on the NYSE dataset, for \$1 investment, with different values of α_{risk} used for Risk Minimization. The wealth accumulated grows with increase of α_{risk}. When the permissible risk is very high, the total wealth becomes less sensitive to the change in α_{risk} and this is observed for $\alpha_{risk} > 20$ (best viewed in color).

We now present the results of our experiments with RAMP on different datasets and compare its performance with other algorithms.

Datasets: The experiments were conducted on two real-world datasets: the New York Stock Exchange dataset (NYSE) [36] and a Standard & Poor's 500 (S&P 500) dataset. The NYSE dataset consists of 36 stocks with data accumulated over a period of 22 years from July 3, 1962 to Dec 31 1984. The dataset captures the bear market that lasted between January 1973 and December 1974. The S&P500 dataset that we used for our experiments consists of 258 stocks which were present in the S&P500 index in 2011 and were alive since January 1990. This period of 22 years from 1990 to 2011 covers bear and bull markets of recent times.

Methodology: We chose a pool of popular Base Portfolio selection algorithms (BP)

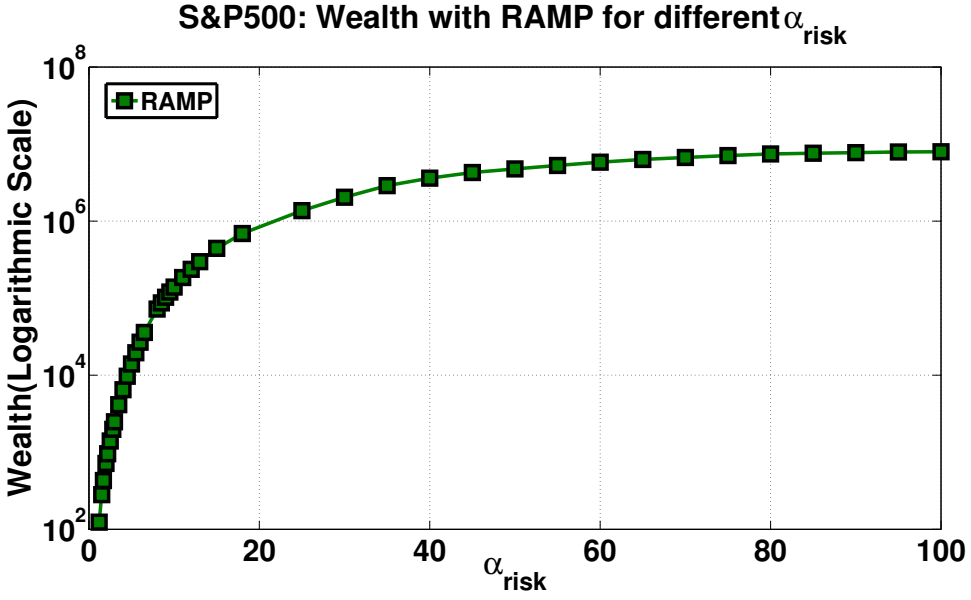


Figure 6.2: Monetary returns of the RAMP on the S&P500 dataset, for $1 investment, with different values of α_{risk} used for Risk Minimization. The wealth accumulated grows with increase of α_{risk}. When the permissible risk is very high, the total wealth becomes less sensitive to the change in α_{risk} and for $\alpha_{risk} > 60$, the change in α_{risk} hardly changes the wealth accumulated (best viewed in color).

which use price relatives as their choice of momentum and run them to generate a sequence of portfolios for each dataset. Next, we run RAMP with a varied range for α_{risk} values and use the portfolios from BPs as input to generate our sequential meta portfolios. At the beginning of every day we estimate the covariance matrix of the price relatives. Additionally we run a couple of non-risk adjusted meta portfolio algorithms with the same BPs to compare their performance with RAMP.

Base Portfolios (BP): We have used Universal Potfolios(UP) [36], Exponentiated-Gradient(EG) [61], Online Newton Step method(ONS) [2] as our base portfolio selection algorithms. We also have a heuristic Anticor(AC) in our base pool because of their exceptional empirical performance seen in [18, 40]. **Meta Portfolios (MP):** Additionally we run a *non-risk adjusted* Meta Portfolio, MP_{GD} algorithm (unconstrained RAMP) with the same BPs to compare their performance with RAMP. MP_{GD} (RAMP without its risk constraint) uses gradient based update as seen in Section 3) [40]. We

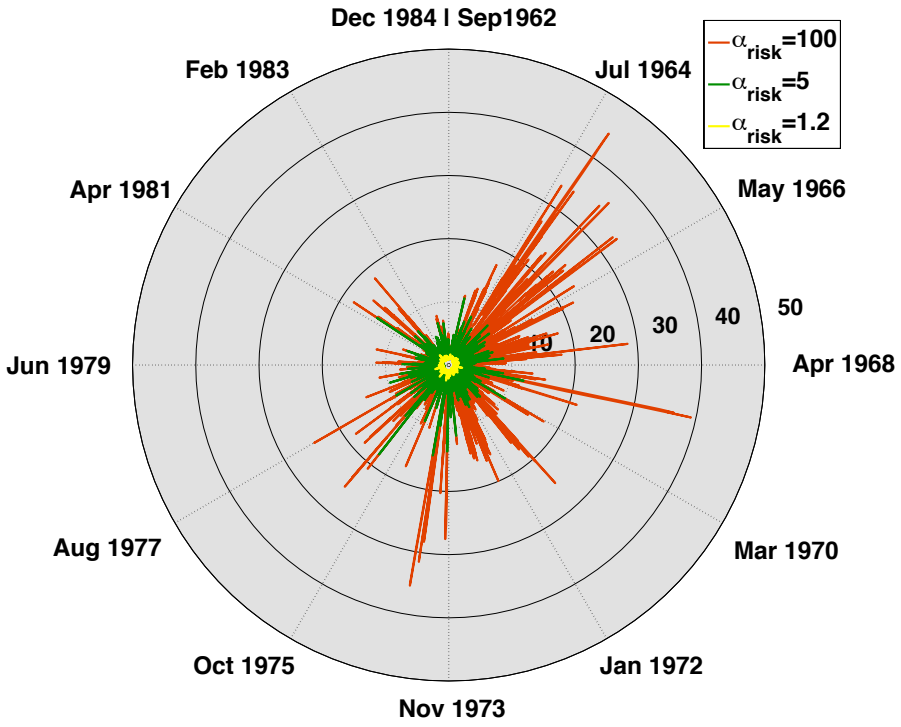


Figure 6.3: RAMP for NYSE: Comparison of γ_{risk} with their corresponding α_{risk} values. Setting permissible risk: α_{risk} to appropriate values, it is possible to achieve small values of true risk: γ_{risk} for our portfolios. During the 1973-74 market crash, by setting permissible risk ($\alpha_{risk} = 1.2$) it is possible to achieve small values of true risk for the portfolio (best viewed in color).

observed that RAMP was quite robust to the choice of the learning rate.

Risk: In order to evaluate the risk of a portfolio, we estimated the covariance matrix of the price relatives using formulation in Section 2 and chose an RBF kernel to decide the weights. We used the Uniform Constant Rebalanced Portfolio [32] (U-CRP also referred to as uniform portfolio) as our reference portfolio u for risk minimization. Using U-CRP is similar to using EWI (Equally Weighted Index) which is a popular weighting schemes for S&P500 and equally weighs the constituent S&P500 stocks. However, it is possible to use any other reference portfolio eg. Buy-and-Hold portfolio. We call α_{risk}, the 'permissible risk' with which we run RAMP and we call $\gamma_{risk} = p_t^T \Sigma_t p_t / u^T \Sigma_t u$ the 'true risk' of the portfolio p on day t. We ran RAMP with a wide range of α_{risk} values starting from 1 to 100 for both the datasets. For details on parameter choices for the base and

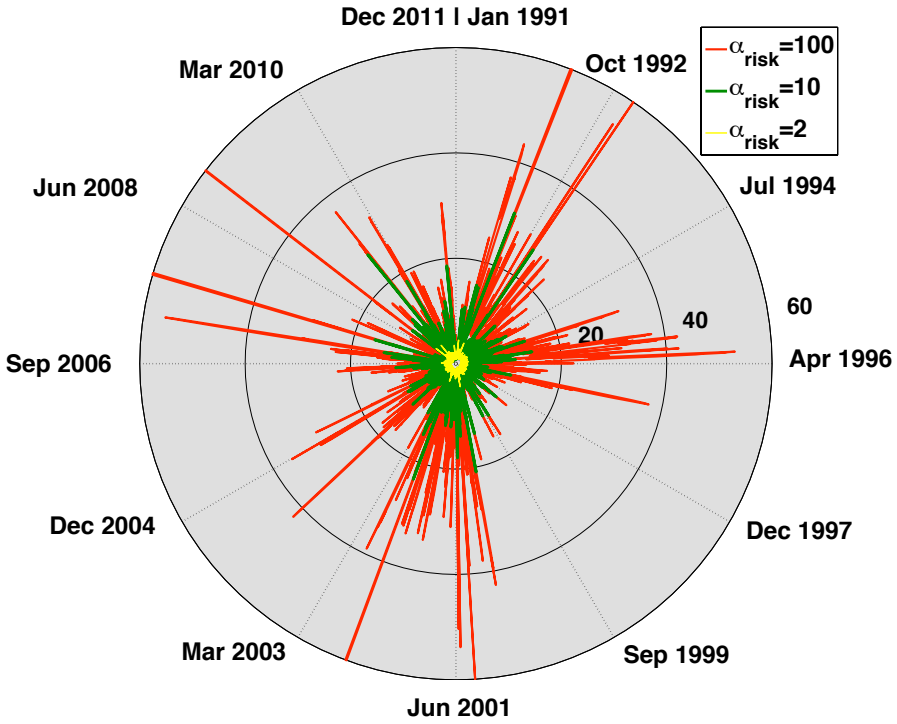


Figure 6.4: RAMP for S&P500: Comparison of γ_{risk} with their corresponding α_{risk} values. Setting permissible risk: α_{risk} to appropriate values, it is possible to achieve small values of true risk: γ_{risk} for our portfolios. During the major market movements between 2000-03 and between 2006-07 which coincides with the dot-com and the housing bubble, by setting permissible risk ($\alpha_{risk} = 2$) it is possible to achieve small values of true risk for the portfolio (best viewed in color).

meta portfolio algorithms, we refer the reader to the supplement.

No Risk No Gain: Investors usually prefer to be aware of the risk of their investment and it is somewhat expected that risk averse investors might end up with low returns than their less conservative counterparts.The key distinctive feature of our RAMP algorithm is that it enables an investor to specify his/her risk tolerance which is crucial in financial investment. RAMP then goes on to recommend the optimum meta portfolio for a given risk level. Figures 6.1 and 6.2 show how permissible risk controls the total the wealth accumulated by RAMP for the NYSE and S&P500 datasets respectively. Figures 6.1 and 6.2 clearly depicts a low risk low gain trend. We particularly observe

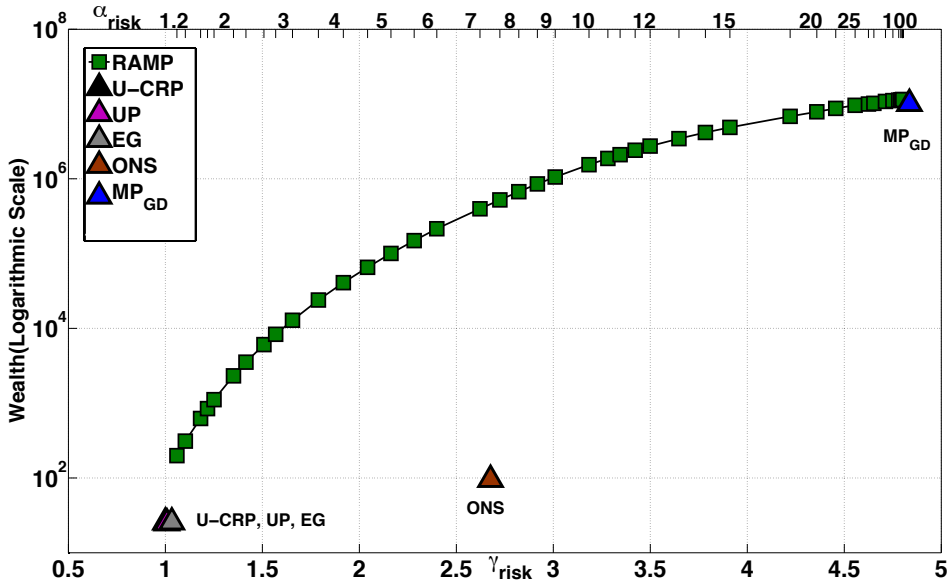


Figure 6.5: NYSE: For equivalent γ_{risk}, RAMP achieves greater multiplicative wealth (for \$1 investment shown here) compared to the BPs: UP, EG, ONS and the benchmark U-CRP. With high values of γ_{risk}, its behavior is equivalent to the non-risk adjusted MP_{GD} (best viewed in color).

that as the permissible risk is allowed to grow, the wealth also grows with other conditions remaining unchanged. For both the datasets, U-CRP makes the least amount of money while RAMP with α_{risk} set to 1.2, outperforms U-CRP by a notable margin. This is because RAMP is guaranteed to produce the optimal portfolio for a given risk level. We also observe that for high values of α_{risk}, marked increase in permissible risk does not change the total wealth accumulated significantly. The general trend can be summarized as follows: for low values of α_{risk}, change in the value of the permissible risk has a marked influence on the wealth accumulated. When the permissible risk is very high, the total wealth becomes less sensitive to the change in α_{risk}. For NYSE this dampening is observed around α_{risk} 20-25, as seen in Figure 6.1, while for S&P500 the dampening is much slower and is observed around values of α_{risk} between 60-70 as seen in 6.2 .

Permissible and True Risk: Figure 6.3 and 6.4 compares the true risk (γ_{risk}) with the permissible risk (α_{risk}) for the NYSE and S&P500 dataset respectively. Firstly, we

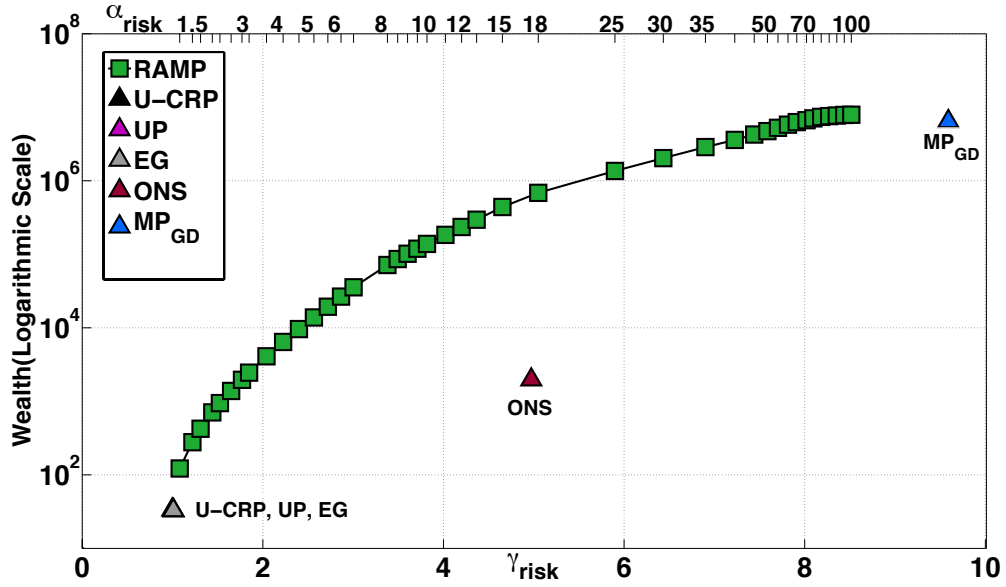


Figure 6.6: S&P500: For equivalent γ_{risk}, RAMP achieves greater multiplicative wealth (for \$1 investment shown here) compared to the BPs: UP, EG, ONS and the benchmark U-CRP. It achieves the same wealth as that of non-risk adjusted MP$_{GD}$ with significantly lower γ_{risk} (best viewed in color).

observe that by controlling the permissible risk it is possible to control the true risk of our portfolios. Figures 6.3 and 6.4 show that for α_{risk} set to high values (100 in the figure), the γ_{risk} is typically unstable and can peak to large values on certain days (going up to 100 for S&P500). It is interesting that the peaks of γ_{risk} for $\alpha_{risk} = 100$ in Figure 6.4 between 2000-03 and between 2006-07 coincides with the dot-com and the housing bubble. We observe that by being conservative in choosing the permissible risk it might possible to control the true risk of the portfolio even during drastic market movements.

Comparison with BPs and MP$_{GD}$: Figures 6.5 and 6.6 compares the wealth accumulated by the BPs, MP$_{GD}$ and RAMP and their corresponding mean *true risk* (γ_{risk}) for NYSE and S&P500 respectively. The wealth (in logarithmic scale) is plotted on the Y-axis. The mean γ_{risk} of each algorithm is plotted on the bottom X-axis and the α_{risk} values for RAMP can be found on the top X-axis. Table 6.1 supplements this information with the APY and γ-risk values of the the algorithms for both the datasets.

Table 6.1: APY and mean γ_{risk} of RAMP, BPs and MP_{GD} on NYSE and S&P500. With equivalent γ_{risk}, RAMP achieves greater APY than any of the BPs and MP_{GD} .

NYSE	EG	RAMP	ONS	RAMP	AC	RAMP	MP_{GD}
APY	15.89	**27.19**	22.99	**62.03**	83.25	**109.35**	108.17
γ_{risk}	1.03	1.05	2.67	2.62	12.66	4.80	4.83
S&P500	EG	RAMP	ONS	RAMP	AC	RAMP	MP_{GD}
APY	18.14	**25.79**	43.58	**82.24**	93.99	**113.09**	111.01
γ_{risk}	1.01	1.08	4.96	4.64	29.16	8.50	9.57

We see the following trend amongst the BPs: EG and UP and U-CRP accumulate the least wealth with very small risk. RAMP clearly makes more money than either of them with the same risk. ONS fares slightly better than EG and UP and U-CRP, in terms of wealth but at the cost of significantly higher risk. However, note that RAMP with γ_{risk} close to that of ONS has almost triple the APY of ONS for NYSE and almost double in case of S&P500 (as seen in Figure 6.6 and Table 1). Amongst the BPs, Anticor gathers the highest wealth of all the BPs, however with much higher γ_{risk} than the rest as seen in Table 6.1 (left out of Figures 6.5 and 6.6 because of not being able to accommodate very high γ_{risk} in the interest of space). RAMP even outperforms Anticor and can achieve greater wealth with lower risk as seen in Table 6.1. As expected RAMP with high α_{risk} replicates the performance of MP_{GD} (non-risk adjusted RAMP). Hence, RAMP can operate on an entire spectrum of risk values which can be pre-specified by an investor and for any given risk level, it outperforms the BPs and MP_{GD} in terms of the wealth.

RAMP's meta weight on BPs: Figure 6.7 shows how RAMP's distribution of wealth over the 3 BPs (EG, ONS and Anticor) changes with α_{risk} for the NYSE dataset. We have already observed that for the NYSE dataset, EG has the least risk, followed by ONS and Anticor realized the highest risk. The wealth accumulated by these BPs however follows the reverse ordering. Thus with very low α_{risk}, RAMP concentrates its weight on EG. As we increase α_{risk}, to 4 and 6, RAMP tends to favor ONS which has medium risk and medium wealth gain amongst all the BPs. As we further relax the risk constraint and set α_{risk} to 95 and 100, RAMP shifts almost all its wealth to Anticor. This is because for high values of α_{risk}, RAMP is oblivious to the risk and chooses the

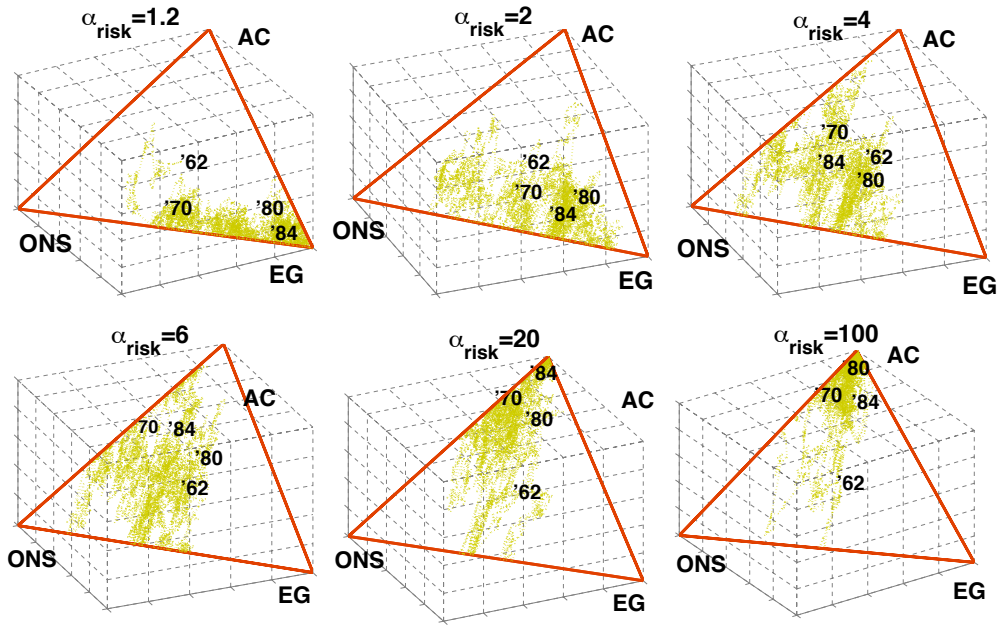


Figure 6.7: Temporal trajectory of RAMP's weight distribution on the BPs: EG, ONS and Anticor. With low α_{risk} values, most of RAMP's weight is concentrated on EG (has inherent low γ_{risk}) , as α_{risk} increases, the weight shifts first to ONS (moderate wealth and moderate γ_{risk}) and with high values of α_{risk}, weight shifts entirely on to Anticor (achieves greatest wealth amongst the BPs with large γ_{risk}(best viewed in color).

portfolio which makes the most money (which is Anticor in this case). Hence depending on the value of α_{risk} that we set and the independent performance of the BPs in terms of risk and wealth accumulated , RAMP will adaptively change its weight distribution on the BPs.

6.5 Conclusions

We have presented an efficient and scalable algorithm for solving Quadratically Constrained Convex Optimization problems in the Online setting. Using a primal-dual approach based on ADMM, our algorithm overcomes the computational bottleneck that arises in the projection step of Online Convex Optimization when faced with ellipsoidal constraints. We extend our work to portfolio selection, where the existing online algorithms do not have a way of accounting for risk. We adopt Markowitz's mean-variance

framework for risk and propose RAMP, the risk adjusted meta portfolio which combines base portfolios and is adept in satisfying the risk constraint. Through extensive experiments over the NYSE and S&P500 datasets, we observe that RAMP for a given risk level outperforms existing online portfolio selection algorithms. RAMP is also competitive with the best heuristic in its pool of base algorithms and has lower risk. For our future work we plan to look at other existing notions of risk in finance as additional constraints.

Chapter 7

Dynamic Gaussian Networks

7.1 Motivation

A network is defined as a collection of variables which are inter-dependent through a complex dependency structure. Examples include social networks, protein-protein interaction networks, communication systems between operators, dependency between stock prices, user interaction in social media [104], etc. Often, network data is large, noisy, dynamic and even unobservable, thus creating unique challenges for developing methods of analysis [5].

Analysis of networks has mostly focused on analyzing the structural properties, and flow behavior, which rely on knowledge of the network structure. However, it is now increasingly becoming apparent that obtaining the structure can be challenging. For example, prices of stocks in a stock market interact and influence each other through complex financial and economic dependencies [48]. In a social network, human users and their friends share dependencies for sharing information and influencing online behavior [104].

In many scenarios, the structure of the network is not known beforehand and the problem of interest lies in estimation of the structure based on sample data from the network variables. For example, in social media, we can often observe people (or media sites) talking about a new piece of information without explicitly observing the path it took in the information diffusion network to reach the particular node of interest [104]. Therefore, estimating the network of information flow can enable prediction of

user preferences or sentiments, by analyzing information propagation from a subset of the users to their neighbors.

Certain statistical methods exist for estimating the structure of the network, under time-invariant scenarios [15]. For example, one can compute correlations between the nodes from observed samples and construct the network based on the correlation strength. However, for most real world networks, we have limited data samples, often corrupted by noise, and such simple statistical tests fail to accurately determine the dependencies between nodes.

In many domains, such as bioinformatics, social media, climate, etc., it is believed that the underlying network is *sparse*, i.e., there exists only a few edges between the nodes [88]. Under such assumptions, it is possible to estimate the structure with high accuracy, even if the sample size is less than the number of nodes (dimensionality) of the network. Recent work has established methods for estimation of the network structure in the case when some of the variables in the network are unobserved [33, 35].

Time-invariance of the network structure is not an appropriate assumption for many domains. For example, the user interactions in social media evolve constantly over time, when new "information pathways" are formed, while old ones become obsolete [104]. Similarly, the relationship patterns in stocks may change significantly over years depending on the companies' financial developments and international economic policies. Under such scenarios, we need novel methods to estimate the network structure as it changes over time, from temporal samples obtained from the network variables.

In this chapter, we describe a method for estimation of the dynamic network, when the network variables follow a multivariate Gaussian distribution. Under this assumption, the conditional independence structure of the distribution is encoded in the network structure [80]. Our method *DyGN* (Dynamic Gaussian Network) can recover the network structure in an "online" fashion, i.e., without knowledge of the entire time-series of data samples, thus providing significant computational benefits over batch estimation methods.

We present an efficient primal dual based alternating directions method of multipliers (ADMM) algorithm to track time-varying networks and we provide theoretical guarantees to show that our online algorithm is competitive with its the batch counterpart that has the benefit of hindsight. Experiments on synthetic data illustrate that

DyGN can accurately capture an abruptly changing structure, even with a small number of samples. Further experiments on data from the stock market shows that the network estimated by DyGN captures some of the complex dynamics that exist within financial institutions.

Our chapter is organized as follows. In Section 7.2, we briefly review some popular network estimation methods and motivate our time varying network estimation framework. We describe the general Gaussian network structure estimation methods in Section 7.3. We introduce our dynamic network framework and go on to describe our algorithm and analysis in Section 7.4. We present extensive experimental results with synthetic and real world data in Section 7.5. We conclude with additional remarks and discussions in Section 7.6.

7.2 Network Estimation

We consider some popular approaches for network estimation and illustrate that they can run into issues, and may give incorrect estimates. In the following, we assume that the network has p variables and denote the variables in the network as $Y = (y_1, \ldots, y_p)$.

Covariance Thresholding: One of the classical methods for estimation is to compute the sample covariance matrix, $\hat{\Sigma}$, obtained from n samples $Y_1, \ldots, Y_n$, and consider that a dependency exists between y_i and y_j, if and only if $|\hat{\Sigma}_{i,j}| > a_{i,j}$, where $a_{i,j}$ is a pre-determined threshold. However, it is well known [15] that when the number of variables p is much larger than n, the sample covariance matrix is ill conditioned and analytical choice of threshold is difficult [15].

Pairwise Mutual Information: A more robust method of estimating the structure is to compute the pairwise mutual information (MI) between the nodes of the network [100], and discarding edges between pairs which have insignificant MI (computed through some measure of statistical significance). However, both pairwise correlation and MI do not capture conditional independence. For example, let us consider three binary random variables 'Rain' (RN), 'Wet Grass' (WG) and 'Wet Road' (WR) shown in Figure 7.1. Both probabilities $p(WG = 1|RN = 1)$ and $p(WR = 1|RN = 1)$ are large. Then, the correlation and MI between WG and WR will be high, although, WG and WR are conditionally independent, given RN.

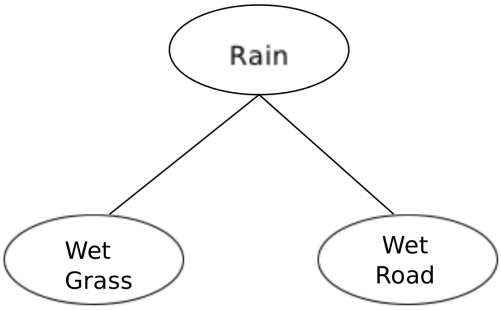


Figure 7.1: 'Wet Grass' and 'Wet Road' are conditionally independent, given 'Rain', but have high correlation and Mutual Information

Conditional MI: In order to estimate the *conditional independence structure* between variables in a network, we need to compute the conditional mutual information (MI) between each pair of variables $MI(y_i, y_j|Y_{-i,-j})$. For general distributions, computation cost of this quantity for all pairs is very high, exponential in the dimensionality p [34] .

If the underlying distribution P is Gaussian, then the pairwise conditional independence between y_i and y_j corresponds to the $(i,j)^{th}$ entry of the *inverse of the covariance matrix* being 0 [80], and we can obtain efficient algorithms for computing this quantity and hence obtain the network structure. We now provide a brief background on Gaussian networks and methods for structure estimation.

7.3 Gaussian Graphical Models

As before, let us consider a p-variate random vector $\mathbf{Y} = (Y_1, \ldots, Y_p)$ with distribution P. This distribution can be characterized by an undirected graph $G = (V, E)$, which represents the conditional independence relations between the components of $\mathbf{Y}$. The vertex set V has p components $Y_1, \ldots, Y_p$ and by the Hammersley-Clifford theorem [80], if Y_i is conditionally independent of Y_j given the other variables, then the edge (i, j) is not in E. If $\mathbf{Y} \sim \mathcal{N}(0, \Sigma)$, the missing edges correspond to zeros in the inverse covariance matrix or *precision* matrix given by $\Sigma^{-1} = \Theta$, i.e., $(\Sigma^{-1})_{ij} = 0 \ \forall (i,j) \notin E$.

The problem of estimating precision matrices with elements set to zeros is known as covariance selection [10] . Classical approaches of estimating the correct set of non-zero elements and then estimating the non-zero entries [44] work well when p is small. Given,

that we have n i.i.d. samples of data $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ from the distribution, we compute the empirical covariance $\hat{\Sigma} = \frac{1}{n-1}\sum_{k=1}^{n} \mathbf{Y}_k \mathbf{Y}_k^T$. When $(p \gg n)$, $\hat{\Sigma}$ is rank-deficient and its inverse cannot be used to estimate the precision matrix Θ. However, for a sparse graph, i.e. most of the entries in the precision matrix are zero, several methods exist that can estimate Θ [93, 51].

In the i.i.d. case, a popular line of work considers the ℓ_1-penalized maximum likelihood estimators [10, 51] which involves solving a convex objective and has theoretical guarantees for the estimation. The log-likelihood of the i.i.d. data takes the form:

$$\log\left\{\prod_{k=1}^{n} f(Y_k)\right\} = -\frac{np}{2}\log(2\pi) + \frac{n}{2}\log\det(\Theta) - \frac{n}{2}\mathrm{Tr}(\Theta\hat{\Sigma}). \tag{7.1}$$

where $\hat{\Sigma}$ is the empirical covariance matrix. The ℓ_1-penalized convex optimization problem then takes the form:

$$\hat{\Theta} = \underset{\Theta \succ 0}{\operatorname{argmin}} \left\{ -\log\det(\Theta) + \mathrm{Tr}\left(\Theta\hat{\Sigma}\right) + \lambda\|\Theta\|_1 \right\}, \tag{7.2}$$

where $\|\Theta\|_1 = \sum_{i,j} |\Theta_{ij}|$, is added to the objective function to induce sparsity. Most of the previous work in this domain deals with estimating a single static precision matrix from i.i.d. samples [118, 51, 10]. Under certain statistical assumptions, one can show statistical consistency of estimation of the precision matrix [118].

7.4 Dynamic Network

The methods we have discussed so far, assume that the graphical structure is stable or time invariant. But it is easy to cite scenarios when this time invariant assumption might be violated. An example could be the conditional independence structure between stock prices of different companies which could be used for financial analysis and stock price prediction. It is reasonable to suppose that the dependency structure between the stocks changes over time to reflect market movements in the financial sector. These changes could be sudden or could be smooth and spread out over a certain period of time.

We consider the scenario where the conditional independence structure of a graph can change over time, i.e., we let $\mathbf{Y}_t \sim \mathcal{N}(0, \Sigma_t)$ at each time step t. We obtain

an independent sequence of p-dimensional observations $(\mathbf{Y}_{t,1}, \ldots, \mathbf{Y}_{t,n})$ according to the multivariate normal distribution with covariance matrix Σ_t. The undirected graph $G_t := (V, E_t)$ is associated with each $\mathbf{Y}_t$ at time step t. Our problem can then be formulated as estimating the sequence of precision matrices $(\Theta_1, \ldots, \Theta_T)$, at every time step t.

7.4.1 Related Work

A few methods have been proposed in recent years for estimating the associated graph of a time varying multivariate normal distribution. These expositions have been motivated by problems arising in biology, astronomy, social sciences and the stock market. [126] provided a kernel-smoothing framework for this model. They proposed to solve the problem (7.2) using a smoothened empirical covariance matrix $\hat{\Sigma}_{kernel}(t) = \frac{\sum_s w_{st}\hat{\Sigma}_s}{\sum_s w_s}$, where $w_{st} := K(s,t)$ is a kernel function for temporal smoothing between time steps s and t. Our model closely resembles their approach in that we assume that a suitable norm of precision matrices at every time step is bounded by a positive constant. It translates to assuming that the network changes smoothly over time. [78] extended the model proposed by [126] by using sparse regression following [93] and solved the resulting convex optimization problem using coordinate descent method [116] . By introducing a strict assumption on the Fisher information matrix $\Sigma \bigotimes \Sigma$, where $\bigotimes$ denotes the Kronecker product, they proved rates of statistical convergence of the recovery of the support set of the precision matrix.

All the above mentioned approaches require that the ratio $\frac{n}{p}$ be a strictly positive constant, in order to prove consistency guarantees of estimation. The optimality guarantees of our model do not require any assumption on the ratio $\frac{n}{p}$.

7.4.2 Online Learning for Dynamic Network

We focus on an online convex optimization setting for estimating the sequence $\Theta_1, \ldots, \Theta_T$. In the traditional online setting, optimization proceeds in rounds, where in round t the online algorithm has to pick Θ_t; then nature reveals the f_t and the value of the objective function $f_t(\Theta_t)$ is revealed. The strictly convex function f_t can be arbitrary, even adversarial, as long as it satisfies some minimal regularity conditions. Our goal is then

to design an algorithm to give us a sequence $\Theta_1, \ldots, \Theta_T$, such that the following *regret* is sublinear in T

$$R_T = \sum_{t=1}^{T} f_t(\Theta_t) - \min_{\Theta_1^*, \cdots, \Theta_T^*} \sum_{t=1}^{T} f_t(\Theta_t^*) \leq o(T) + \sum_{t=2}^{T} \|\Theta_t^* - \Theta_{t-1}^*\|_q \ , \tag{7.3}$$

where $\|\cdot\|_q$ is a suitable matrix norm. The sequence $\{\Theta_1^*, \cdots, \Theta_T^*\}$ is the minimizer of $\sum_{t=1}^{T} f_t$ as it has the power of hindsight. The bound in (7.3) is a *shifting bound* [62] which allows the $\{\Theta_1^*, \cdots, \Theta_T^*\}$ to also change over time. If the sum $\sum_{t=2}^{T} \|\Theta_t^* - \Theta_{t-1}^*\|_q$ is bounded by $o(T)$, then "on the average" the algorithm performs as well as the best sequence in hindsight.

We now pose our problem of learning the conditional independence structures in a dynamic setting as an online convex optimization problem. We adopt the framework of the ℓ_1 penalized maximum likelihood estimators and set

$$f_t(\Theta) = \left\{ -\log\det(\Theta) + \mathbf{Tr}\left(\Theta \hat{\Sigma}_t\right) + \lambda \|\Theta\|_1 \right\} \ , \tag{7.4}$$

where **Tr** denotes the trace. We now note that a batch algorithm which has the benefit of hindsight and generates a sequence to minimize$\sum_{t=1}^{T} f_t$, hence solves a total variation (TV) regularization problem [117].

In a real world scenario, $\mathbf{Y}_t$ arrives at time step t and it is not possible to know the entire sequence ($\mathbf{Y}_{1:T}$) before hand. Our online learning algorithm can dynamically estimate Θ_{t+1} without observing $\mathbf{Y}_{t+1}$ as follows,

$$\begin{aligned} \Theta_{t+1} = \min_{\Theta \succ 0} \ & -\text{logdet}(\Theta) + \text{Tr}(\Theta \hat{\Sigma}_t) + \lambda ||\Theta||_1 \\ & + \beta ||\Theta - \Theta_t||_1 + \gamma ||\Theta - \Theta_t||_F^2 \ , \end{aligned} \tag{7.5}$$

where γ is a suitable learning rate. We refer to our algorithm, which sequentially solves (7.5), as Dynamic Gaussian Network estimation (DyGN). We first propose an efficient algorithm to solve (7.5) and thereforth show that our algorithm has regret sublinear in T.

7.4.3 ADMM algorithm for DyGN

We propose an alternating direction method of multipliers (ADMM) [21] based primal dual algorithm for estimating Θ_{t+1} by solving (7.5). ADMM is an efficient distributed

Algorithm 9 ADMM Updates for DyGN

1: ADMM iterations

$$\Theta^{(k+1)} = \underset{\Theta \succ 0}{\operatorname{argmin}} \; -\text{logdet}(\Theta) + \text{Tr}(\Theta\hat{\Sigma}) + \frac{\rho}{2}||\Theta - X^{(k)} + U^{(k)}||_F^2 + \frac{\rho}{2}||\Theta - \Theta_t - Z^{(k)} + V^{(k)}||_F^2 \tag{7.6}$$

$$X^{(k+1)} = \underset{X}{\operatorname{argmin}} \quad \lambda||X||_1 + \frac{\rho}{2}||\Theta^{(k+1)} - X + U^{(k)}||_F^2 \tag{7.7}$$

$$Z^{(k+1)} = \underset{Z}{\operatorname{argmin}} \quad \beta||Z||_1 + \gamma||Z||_F^2 + \frac{\rho}{2}||\Theta^{(k+1)} - \Theta_t - Z + V^{(k)}||_F^2 \tag{7.8}$$

$$U^{(k+1)} = U^{(k)} + (\Theta^{(k+1)} - X^{(k+1)}) \tag{7.9}$$

$$V^{(k+1)} = V^{(k)} + (\Theta^{(k+1)} - \Theta_t - Z^{(k+1)}) \; . \tag{7.10}$$

2: Continue until Stopping Criterion is met.

optimization method, closely related to Bregman iterative algorithms for ℓ_1 problems and proximal point methods. It has been applied in many large scale problems in statistics and machine learning beacuse of its computational benefits and fast convergence in practice.

We first rewrite (7.5) by introducing auxillary variables X and Z as follows:

$$\min_{\substack{\Theta \succ 0; \Theta = X \\ \Theta - \Theta_t = Z}} \; -\text{logdet}(\Theta) + \text{Tr}(\Theta\hat{\Sigma}_t) + \lambda||X||_1 + \beta||Z||_1 + \gamma||Z||_F^2.$$

The ADMM formulation naturally lets us decouple the ℓ_1 term from the smooth terms which is computationally advantageous.The augmented lagrangian for (7.6) can be written as: $L_\rho(\Theta, X, Z, U, V) = -\text{logdet}(\Theta) + \text{Tr}(\Theta\hat{\Sigma}_t) + \lambda||X||_1 + \beta||Z||_1 + \gamma||Z||_F^2 + \frac{\rho}{2}||\Theta - X^k + U^k||_F^2 + \frac{\rho}{2}||\Theta - \Theta_t - Y^k + V^k||_F^2$, where U and V are the scaled dual variables.

The detailed ADMM updates are provided in Algorithm 9 . The update for $\Theta^{(k+1)}$ has a closed form solution while we can use the shrinkage operator [114] to perform the updates for $X^{(k+1)}$ and $Z^{(k+1)}$. We use the primal and dual residuals to keep track of the convergence of the algorithm and use the stopping criterion specified in [21].

7.4.4 Analysis

Let $\{\hat{\Sigma}_t \in \mathbb{R}^{p\times p}\ , t = 1, \ldots, T\}$ be a series of covariance matrices obtained sequentially at time steps t, $t = 1, \ldots, T$. We predict the precision matrix Θ_t obtained from Algorithm 9, after observing the series of covariance matrices $\{\Sigma_1, \ldots, \Sigma_{t-1}\}$, and suffer a loss $f_t(\Theta_t)$ given by $f_t(\Theta) = -\log\det\Theta + \langle\Sigma_t, \Theta\rangle + \alpha\|\Theta\|_1 + \beta\|\Theta - \Theta t - 1\|_1$, where $\langle\Sigma_t, \Theta\rangle = \mathrm{Tr}(\Sigma_t\Theta)$. Our goal is to minimize the *regret* with respect to the least loss in hindsight, given by the sequence of precision matrices $\Theta_1^*, \ldots, \Theta_T^*$ s. Then, the regret we want to upper bound is

$$R_T := \sum_{t=1}^{T} \left[f_t(\Theta_t) - f_t(\Theta_t^*)\right] \ . \tag{7.11}$$

Our proof of the regret bound relies on the following lemma:

Lemma 5. *Let us define* $G := \sup_{g\in\partial f_t(\Theta)} \|g\|_F$ *, where* $\partial f_t(\Theta)$ *denotes the subgradient set of* f_t *at* Θ*, and* $\Theta \succ 0$*. Then, at each step* t *of the algorithm,*

$$\eta_t[f_t(\Theta_t) - f_t(\Theta)] \leq \frac{\eta_t^2}{2}G^2 + \frac{1}{2}\left[\|\Theta - \Theta_t\|_F^2 - (1 + \eta_t\nu)\|\Theta - \Theta_{t+1}\|_F^2\right] \ , \tag{7.12}$$

where ν *is a strong convexity parameter of* f_t *,*$\forall t$ *.*

We assume that a suitable norm of Θ is bounded from above and hence the eigenvalues of Θ are also bounded. A brief proof sketch is provided in the appendix.

Finally we prove the following regret bound:

Theorem 10. *Let* $\{\Theta_1^*, \ldots, \Theta_T^*\}$ *be the sequence of precision matrices. Then, for* $\eta_t = \frac{1}{\nu t}$ *, the regret given by* (7.11) *is bounded as*

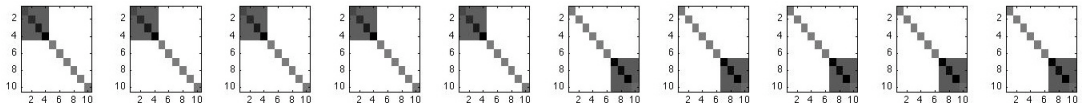
$$R_T \leq \frac{G^2}{2\nu}(1 + \log T) + \nu\nu_1 \sum_{t=1}^{T-1} \|\Theta_t^* - \Theta_{t-1}^*\|_q \ , \tag{7.13}$$

where $\|\cdot\|_p$ *and* $\|\cdot\|q$ *are Schatten matrix norms such that* $\frac{1}{p} + \frac{1}{q} = 1$ *and* $\|\Theta_t\|_p \leq \frac{\nu_1}{2}$ *,*$\forall t$ *.*

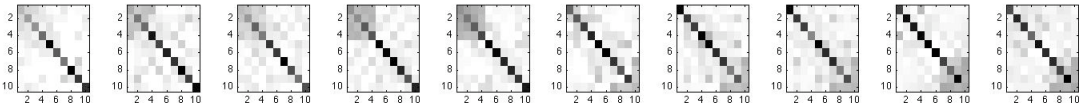
We defer the proof of the theorem to the appendix.

7.5 Experiments

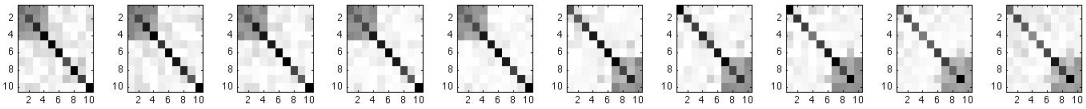
We now present our experimental results on a synthetic and real dataset with DyGN. The real world dataset we use consists of International Stock Market data.



(a) Actual sequence precision matrix structure



(b) Online algorithm: DyGN.



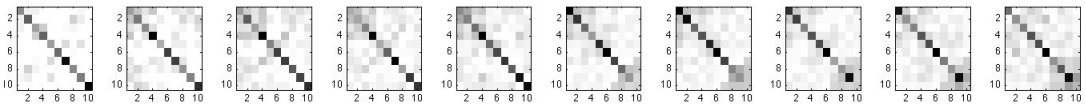
(c) Batch Algorithm

Figure 7.2: Case1: $n > p$. DyGN can successfully detect change in the structure of time varying precision matrices when sample size is greater than dimension and it is competitive with its batch counterpart.

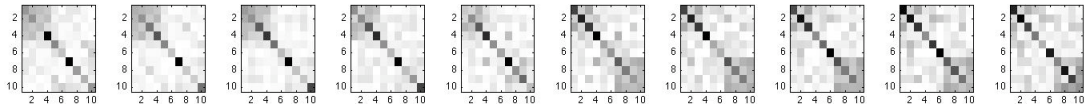
7.5.1 Synthetic Data

Figures 7.2 and 7.3 illustrate an example with a synthetic dataset where the true precision matrix changes over time. We generated a 10-dimensional dataset, with true precision structures for 10 time steps, as shown in Figure 7.2(a). It is to be noted that the true precision structure remains unchanged for the first 5 time steps and changes abruptly and drastically in step 6. We generated samples from a multivariate Gaussian using the true covariance matrices (corresponding to Figure 7.2(a)) at each time step. We implemented a batch algorithm which produces the best sequence of precision matrices $(\Theta_1^*, \ldots, \Theta_T^*)$ with the power of hindsight. We look at two particular secnarios **Case 1:** $n > p$ (Figure 7.2) and **Case 2:** $n < p$ (Figure 7.3)for evaluating the performance of DyGN for tracking dynamic networks.

Detects non-smooth change: As mentioned earlier, the true precision changes in time step 6. We observe that for both **Case 1**, shown in Figure 7.2 and **Case 2** shown in Figure 7.3, DyGN is able to detect this change in precision matrix almost immediately and converges to this new structure in a couple of time steps. Although, DyGN fares better in **Case 1** in terms of the individual values of the precision matrix, it certainly



(a) Online Algorithm: DyGN



(b) Batch Algorithm

Figure 7.3: Case1: $n < p$. Even when the sample size is smaller than the number of dimensions, DyGN can detect change in the structure of time varying precision matrices and it is competitive with its batch counterpart.

is able detect the change and recover the true structure in both cases.

Competitive with the Batch Algorithm: The batch algorithm can see the entire dataset and come up with the best changing/shifting sequence of precision matrices in hindsight. Hence, this is a difficult baseline for comparison. We observe in Figures 7.2 and 7.3, that while the batch algorithm can exactly recover the non-smooth change in precision matrix accurately, DyGN is not far behind and can quickly detect the true structure in a couple of time steps.

Sparsity increases with increasing λ and β: Figure 7.4 shows how the average sparsity of the recovered matrices depend on the values of the parameters λ and β. The x-axis plots the different λ values and the y-axis shows the average fraction of edges (over t time steps) in the precision matrix obtained by DyGN. Every curve in Figure 7.4 represents a particular β value. We show results by fixing γ. It is clearly evident from the Figure 7.4, that for any β value, the sparsity increases as we increase λ. Also, for identical λ values, the sparsity increases as we increase the β. Overall, by choosing λ and β, we can control the sparsity of the recovered precision matrix.

Table 7.1: Country names, their abbreviations and the Index used.

Country	Austria	Switzerland	Spain	United Kingdom	Ireland	United States	Canada	Hong Kong
Abbrv.	AT	SZ	ES	UK	IE	US	CAN	HK
Index.	ATX	SMI	IBEX 35	FTSE 100	ISEQ	S&P500	TSE 300	HSI

Country	Japan	Austria	India	Brazil	Netherlands	Belgium	France	Germany
Abbrv.	JP	AU	IN	BR	NL	BE	FR	GER
Index.	Nikkei 225	ASX	NSEI	IBOV	AEX	BEL 20	CAC 40	DAX 30

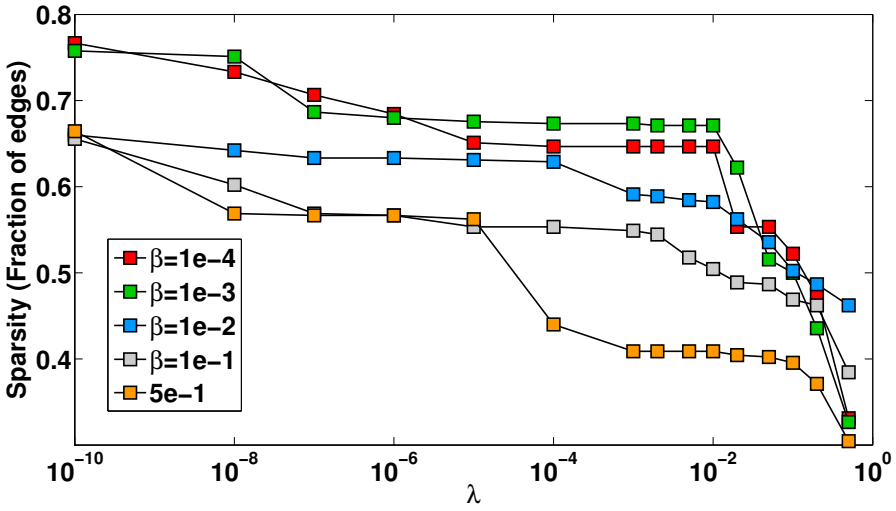


Figure 7.4: Sparsity increases as we increase λ and β.

7.5.2 International Stock Market Data

We work with international stock market indices and our aim is to observe how the structure of the precision matrix between these international markets evolve temporally. Such an analysis can facilitate studies of the comovement of world exchange indices which have long been an active area in finance [65, 47]. Studies have explored if the extent of financial and economical integration between a country-pair may indeed be reflected by the degree of stock markets co-movement that they exhibit [48]. Investigating the propensity of one country to be affected by global shocks have enormous value for preventing future crises.

Dataset and Experimental setup: We consider a multivariate time series of 16 stock market indices: the S&P 5000 composite index (U.S.), Toronto stock exchange 300 index (Canada), the All ordinary composite stock index(Australia), the Nikkei 225 stock index (Japan), the Hang Seng stock composite index (Hong Kong), the FTSE 100 share index (United Kingdom), the Frankfurt DAX 30 composite index (German), the CAC 40 stock composite index (France), the Zurich Swiss Market composite index (Switzerland), the Amsterdam exchange index (Netherlands), the Austrian traded index (Austria), IBEX 35 (Spain), BEL 20 (Belgium), the Irish stock exchange index (Ireland). We also include the market indices from two emerging countries National Exchange

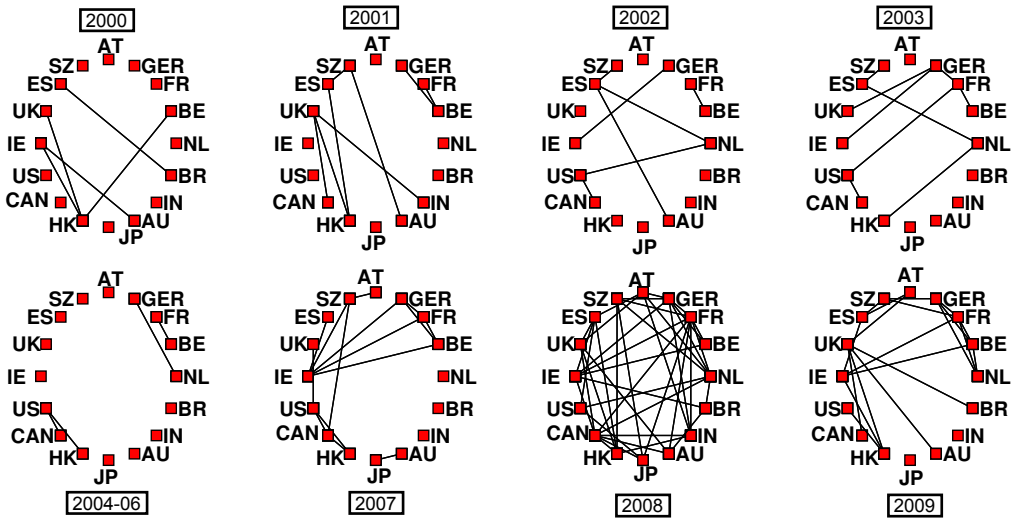


Figure 7.5: Changing dependancy between the international market indices from 2000-09. We observe that there is significant change in dependancy structure in 2008, which coincides with the global financial crisis. Almost all the international markets seem to get interconnected at this time.

Index of India (India) and Bovespa Index (Brazil). The data constitutes 13 years of the daily stock index closing prices recorded from Jan 1, 1999 to December 30, 2012, obtained from www.yahoo.com/finance. Table 7.1 contains comprehensive information about the countries, their representative abbreviations used in the figures and the names of the indices respectively.

Since, the stock indices were reported in different currencies (dollars, euro, real etc.), the data first was converted to US dollars. For each market we make use as variable the logarithm of the price relative on trading day t, defined as $r_t = \log \frac{s_t}{s_{t-1}}$, where s_t is the closing price of a market on day t. We take the z-score values of r_t and Gaussianize the data. We ran experiments with 13 years of data but updated our precision matrix using our online algorithm every quarter of a year (Jan-March being the first quarter of every year).

Overview of Results: We present a snapshot of how the dependancy structure between the different indices changes between 200 and 2009 in Figure 7.5, but have conducted extensive experiments with a range of parameter values for λ, β and γ. One general observation from our experiments is that countries which are spatially adjacent

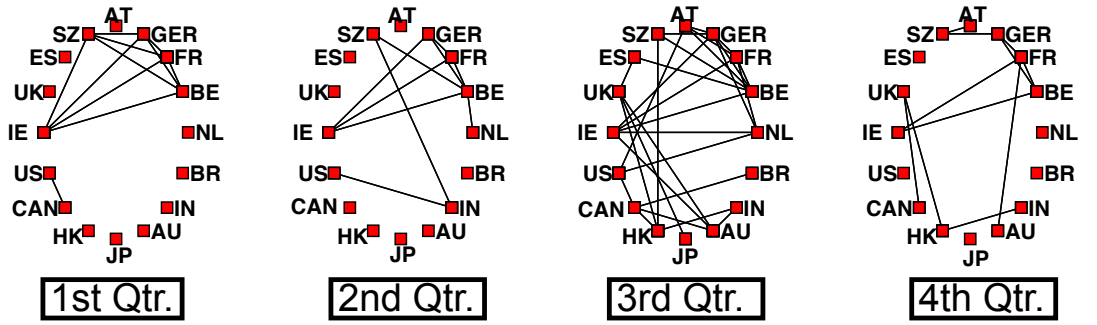


Figure 7.6: Analysis of dependency structure between the international stock markets in 2008. The first quarter corresponds to the first three months an so on. The structure for the 3rd quarter captures the major market movement in Sept. 2008.

often tend to be consistently connected over time. For example, we see in Figure 7.5, that more often that not US and Canada which are neighboring countries are connected. The same is true for France, Belgium and Germany and many of the European countries. Note that, UK and Hong Kong seem to be connected in the early 2000s. This is interesting especially in the light of the fact that Hong Kong was a part of the British colony till the late 20th century.

World Financial Crisis and Index comovement: Perhaps the most interesting part of Figure 7.5 is how the dependancy structure between the international stock indices undergo a drastic change in 2008. Although dependancy graph structure was sparse till 2006, almost all the indices seem to get interconnected in the year 2008. which coincides with great recession otherwise known as the Global Financial crisis of 2007-2008. Figure 7.6, shows the quarterly change of the precision matrix structure particularly for 2008, and hence gives us more insight about the market movements. It clearly shows how the graph structure becomes dense for the months of July, August and September, cumulatively represented as the 3rd quarter here. Incidentally, it is now widely known that for the great recession, although the decline began in December 2007, it took a particularly sharp downward turn in September 2008. Our results, clearly illustrate the potential of our online dynamic networks algorithm: DyGN in structure analysis and prediction for the financial sector. Moreover, we note that such analysis is also beneficial for investors with internationally diverse portfolios which depend on the fact that comovement between international markets is usually low under normal

market conditions

7.6 Conclusions

We present a framework for learning the structure of networks which can vary over time. We pose our problem as an online convex optimization problem and propose an efficient primal dual algorithm based on ADMM to track time varying networks which is theoretically competitive with a batch algorithm which has the power of hindsight. Through extensive experiments over synthetic and a real world dataset in stock market, we illustrate the power of our framework in detecting changes in dynamic networks. A proposed future direction could be to conduct a detailed study on partially observable networks which can evolve over time.

Chapter 8

Conclusions

In this thesis we contribute to the development of the OCO literature by studying various problems that are motivated by real world applications. We present algorithms to tackle these problems and used tools from convex optimization to propose efficient solutions favorable for large scale and dynamic data. Also, we present theoretical analyses of our algorithms and analyze their performance with respect to solutions which have the benefit of hindsight. The key results in our thesis can be summarized as follows:

1. In Chapter 3, we introduced *lazy updates* which makes sparse updates to iterative solutions with the help of a non-smooth ℓ_1 penalty term on the difference of the consecutive solutions (iterates). The resulting convex optimization problem is non-smooth with a feasible set which lies in the probability simplex. This is handled efficiently by ADMM. Moreover, we go on to show that our algorithm has sublinear regret with respect to the best fixed expert in hindsight. We also present analyses with respect to the best sequence of solutions in hindsight.

2. In Chapter 4, we extend the lazy updates framework to composite objectives, i.e, the objective consists of two parts: a convex function which changes over time and a fixed regularizer. We particularly focus on the case when the regularizer is the group sparsity inducing group-lasso. Similar to Chapter 3, we propose an ADMM algorithm which handles non-smooth terms and projection onto a feasible set with unique variable splitting. Our analyses for composite objectives with lazy updates is more general and we show that the analyses in Chapter 3 is a special

case.

3. In Chapter 5, we propose two new Meta Algorithms which can combines solutions from a pool of base algorithms for optimization. The solutions from multiple iterative base algorithms can be combined by Meta Algorithms to outperform the single best base algorithm. We propose a first order algorithm which results in gradient updates and a second order algorithm which results in Newton updates.

4. In Chapter 6, we have presented a framework for handling constraints in the OCO setting (quadratic and linear) and when the constraints can also vary over time. In particular, we present an efficient and scalable algorithm for solving Quadratically Constrained Convex Optimization problems in the Online setting. Using a primal-dual approach based on ADMM, our algorithm overcomes the computational bottleneck that arises in the projection step of Online Convex Optimization when faced with ellipsoidal constraints.

5. In Chapter 7, we present a framework for learning the structure of networks which can vary over time. We pose our problem as an online convex optimization problem and propose an efficient primal dual algorithm based on ADMM to track time varying networks which is theoretically competitive with a batch algorithm which has the power of hindsight.

In addition to contributing in terms of algorithm development and theoretical analyses, there are several practical contributions of this thesis. Although, the solutions presented and the framework is general for any problem in the OCO setting, we illustrate and summarize our results mainly in computational finance with focus on online portfolio selection.

1. In Chapter 3, we have developed a framework and an online algorithm (OLU) to allow for lazy updates for the problem of portfolio selection with transaction costs. Our analysis shows that transaction cost adjusted OLU is competitive with reasonable fixed strategies which have the power of hindsight. Our experimental results describe the behavior of such lazy updates and show that OLU is able to outperform exponentiated gradient (EG) and Buy-and-Hold with reasonable transaction costs.

2. In Chapter 4, we used our composite objective with lazy updates, in particular using the group lasso as a group sparsity regularizer to incorporate sector information in portfolio selection. Our experimental results illustrate the behavior of group sparsity and lazy updates and show that our algorithm OLU-GS is able to outperform baseline strategies with reasonable transaction costs. Finally, we demonstrate that OLU-GS is able to select the best performing sectors during different economic conditions.

3. In Chapter 5, motivated by the fact that heuristic often outperform the theoretically motivated (universal algorithms) online portfolio selection algorithms, we use our two new meta algorithms to combine several baselines for portfolio selection. As expected, we show that our meta algorithms MA_{EG} and MA_{ONS} beat the universal algorithms in terms of empirical performance by harnessing the empirical performance of heuristics but are still competitive with the best CRP in hindsight.

4. In Chapter 6, we use our Online QCCO framework to incorporate risk with meta optimization in portfolio selection. The existing online algorithms do not have a way of accounting for risk. We adopt Markowitz's mean-variance framework for risk and propose RAMP, the risk adjusted meta portfolio which combines base portfolios and is adept in satisfying the risk constraint. Through extensive experiments over two datasets, we observe that RAMP for a given risk level outperforms existing online portfolio selection algorithms.

5. In Chapter 7, we illustrate the strength of our framework in capturing the time varying dependencies between the international market indices. Our experimental analysis provides evidence that our framework can capture economic co-movements in the international market and can detect drastic market changes.

With the ever growing amount of data in recent times, and the need for scalable machine learning algorithms, online convex optimization is becoming an increasingly important area for investigation. In this thesis, we provide theoretical results, algorithms which provide solutions to previously unanswered, but important technical questions in the OCO framework. Moreover, we provide empirical results for an important application

of OCO: the online portfolio selection problem. However, the thesis suggests that our proposed framework is fairly general and can be applied to a variety of different problems which can be cast in the OCO framework, for computational and practical benefits.

References

[1] J. Abernathy and A. Rakhlin. Competing in the dark: An efficient algorithm for bandit linear optimization. In *Proceedings of the Conference on Learning Theory (COLT)*, pages 263–274, 2008.

[2] A. Agarwal, E. Hazan, S. Kale, and R. Schapire. Algorithms for portfolio management based on the newton method. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 9–16, 2006.

[3] A. Agarwal, E. Hazan, S. Kale, and R. Schapire. Algorithms for portfolio management based on the newton method. *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 9–16, 2006.

[4] A. Agarwal, S. Negahban, and M.J. Wainwright. Stochastic optimization and sparse statistical recovery: Optimal algorithms for high dimensions. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 1547–1555, 2012.

[5] E. Airoldi, D. Blei, S. Fienberg, A. Goldenberg, E. Xing, and A. Zheng. *Statistical Network Analysis: Models, Issues, and New Directions*, volume 4503. Springer, 2007.

[6] S. Arora, E. Hazan, and S. Kale. The multiplicative update algorithm: A meta algorithm and applications. Technical report, Dept of Computer Science, Princeton University, 2005.

[7] M. Arouri and D. Nguyen. Oil prices, stock markets and portfolio investment: Evidence from sector analysis in europe over the last decade. *Energy Policy*, 38(8):4528–4539, 2010.

[8] P. Artzne, F. Delbaen, J. M. Eber, and D. Health. Thinking coherently. *Risk*, 10(11):68–71, 1997.

[9] A. Banerjee. On Bayesian bounds. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[10] O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

[11] R. Baraniuk, V. Cevher, M.F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56:1982–2001, 2010.

[12] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[13] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[14] D. Bertsekas. *Convex analysis and optimization*. Athena Scientific, 2003.

[15] P. Bickel and E. Levina. Covariance regularization by thresholding. *The Annals of Statistics*, 36(6):2577–2604, 2008.

[16] A. Blum and A. Kalai. Universal portfolios with and without transaction costs. In *Proceedings of the 10th Annual Conference on Learning Theory*, 1997.

[17] Z. Bodie, A. Kane, and A.J. Marcus. *Portfolio Performance Evaluation, Investments*. Irwin McGraw-Hill, 1999.

[18] A. Borodin, R. El-Yaniv, and V. Gogan. Can we learn to beat the best stock. *Journal of Artificial Intelligence Research*, 21:579–594, 2004.

[19] L. Bottou. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nîmes*, 1991.

[20] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of the 19th International Conference on Computational Statistics*, pages 177–187, 2010.

[21] S. Boyd. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. and Trends in Machine. Learning*, 3(1):1–122, 2010.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[23] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[24] L. Bregman. The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7:200–217, 1967.

[25] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[26] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[27] J. Brodie, I. Daubechies, C. De Mol, D. Giannone, and I. Loris. Sparse and stable markowitz portfolios. *Proceedings of the National Academy of Sciences*, 2009.

[28] R.L. Burden and J. D. Faires. *Numerical Analysis*. Thompson, 1985.

[29] Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms and Applications*. Oxford University Press, 1997.

[30] N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of online learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

[31] N. Cesa-Bianchi, Y. Freund, D. P. Helmbold, D. Haussler, R. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[32] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[33] V. Chandrasekaran, P. Parrilo, and A. Willsky. Latent variable graphical model selection via convex optimization. In *Allerton Conf. on Comm., Control, and Comp.*, pages 1610–1613, 2010.

[34] H. Cheng, Z. Qin, C. Feng, Y. Wang, and F. Li. Conditional mutual information-based feature selection analyzing for synergy and redundancy. *ETRI Journal*, 33(2), 2011.

[35] M. Choi, V. Tan, A. Anandkumar, and A. Willsky. Learning latent tree graphical models. *Journal of Machine Learning Research*, 12, 2011.

[36] T. Cover. Universal portfolios. *Mathematical Finance*, 1:1–29, 1991.

[37] T. Cover and E. Ordentlich. Universal portfolios with side information. *IEEE Transactions of Information Theory*, 42:348–363, 1996.

[38] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[39] V. Dani, T. Hayes, and S. Kakade. The price of bandit information for online optimization. *Advances in Neural Information Processing Systems 20, year = 2008*.

[40] P. Das and A. Banerjee. Meta optimization and its application to portfolio selection. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011.

[41] P. Das, N. Johnson, and A. Banerjee. Online lazy updates for portfolio selection with transaction costs. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 2013.

[42] M. Davis and A. Norman. Portfolio selection with transaction costs. *Mathematics of Operations Research*, 15(4):676–713, 1990.

[43] V. DeMiguel, L. Garlappi, F. J. Nogales, and R. Uppal. A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. *Management Science*, 55(5):798–812, 2009.

[44] Arthur P Dempster. Covariance selection. *Biometrics*, pages 157–175, 1972.

[45] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In *Proceedings of the Conference on Learning Theory (COLT)*, pages 14–26, 2010.

[46] J. Duchi, Y. Singer, and T. Chandra. Efficient projections onto the L1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

[47] C.S. Eun and S.Shim. International transmission of stock market movements. *Journal of Finance and Quantitative Analysis*, 24:241–256, 1989.

[48] K.J. Forbes and R. Rigobon. No contagion, only interdependence, measuring stock market comovements. *Journal of Finance*, 57:2223–2261, 2001.

[49] Y. Freund and R. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999.

[50] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[51] J. Friedman, T. Hastie, and R. Tibshiran. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9 3:432–441, 2008.

[52] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 2000.

[53] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *Arxiv preprint arXiv:1001.0736*, 2010.

[54] B. Fristedt and L. Gray. *A Modern Approach to Probability Theory*. Birkhauser Verlag, 1997.

[55] D. Goldfarb and G. Iyengar. Robust convex quadratically constrained programs. *Mathematical Programming*, 97:495–515, 2002.

[56] T. Goldstein, B. Donoghue, and S. Setzer. Fast alternating direction optimization methods. *CAM report*, pages 12–35, 2012.

[57] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[58] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[59] E. Hazan, A. Kalai, Satyen Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. *Proceedings of the 19th Annual Conference on Learning Theory*, pages 499–513, 2006.

[60] B. He and X. Yuan. On the $o(1/n)$ convergence rate of the douglas-rachford alternating direction method. *SIAM*, 50:700–709, 2012.

[61] D. Helmbold, E. Scahpire, Y. Singer, and M. Warmuth. Online portfolio selection using multiplicative weights. *Mathematical Finance*, 8(4):325–347, 1998.

[62] M. Herbster and M. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

[63] M. Herbster and M. K. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

[64] M. Herbster and M. K. Warmuth. Tracking the best regressor. In *Proc. of the 11th Annual Conference on Computational Learing Theory (COLT)*, pages 24–31, 1998.

[65] J.E. Hillard. The relationship between equity indices on world excahnges. *Journal of Finance*, 34:103–114, 1979.

[66] J. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of Convex Analysis*. Springer Verlag, 2001.

[67] J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. In *International Conference of Machine Learning (ICML)*, 2009.

[68] G. Huber. The generalized treynor ratio. *Review of Finance*, 9:415–435, 2005.

[69] L. Jacob, G. Obozinski, and J. Vert. Group lasso with overlap and graph lasso. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[70] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, pages 2297–2334, 2010.

[71] M. C. Jensen. The performance of mutual funds in the period 1945-64. *Journal of Finance*, 23:389–416, 1968.

[72] S. Kakade and S. Shalev-Shwartz. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *Proceedings of the Conference on Neural Information Processing Systems*, 2008.

[73] A. Kalai and S. Vempala. Efficient algorithms for universal portfolios. *Journal of Machine Learning Research*, 3(3):423–440, 2002.

[74] A. Kalai and S. Vempala. Efficient algorithms for on-line optimization. *Journal of Computer and System Sciences*, 713:291–307, 2005.

[75] J. L. Kelly. A new interpretation of information rate. *Bell Systems Technical Journal*, 35:917–926, 1956.

[76] J. Kivinen and M. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–64, 1997.

[77] O. Knill. Probability. Course notes from Caltech, 1994.

[78] M. Kolar and E. Xing. On time varying undirected graphs. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2011.

[79] J. Langford, L. Li, and A. Strehl. Vowpal Wabbit, 2007.

[80] S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, 1996.

[81] B. Li and S. Hoi. On-line portfolio selection with moving average reversion. In *International Conference of Machine Learning*, 2012.

[82] B. Li, S. Hoi, P. Zhao, and V. Gopalkrishnan. Condence weighted mean reversion strategy for on-line portfolio selection. In *Proceedings of the International Conference of Artificial Intelligence and Statistics*, 2011.

[83] B. Li, S. Hoi, P. Zhao, and V. Gopalkrishnan. Pamr: Passive aggressive mean reversion strategy for portfolio selection. *Machine Learning Journal*, 87:221–258, 2012.

[84] B. Li, S. Hoi, P. Zhao, and V. Gopalkrishnan. Condence weighted mean reversion strategy for online portfolio selection. 7(1):4, 2013.

[85] B. Li and S. C.H. Hoi. Online portfolio selection: A survey. *ACM Computing Surveys (CSUR)*, 46(3), 2014.

[86] Q. Li, M. Vassalou, and Y. Xing. Sector investment growth rates and the cross section of equity returns*. *The Journal of Business*, 79(3):1637–1665, 2006.

[87] N. Littlestone and M. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

[88] N. Luscombe et al. Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 431(7006):308–312, 2004.

[89] M. Magill and G. Constantinides. Portfolio selection with transaction costs. *Journal of Economic Theory*, 13(2):245–263, 1976.

[90] E. Maor. *The Pythagorean theorem. A 4,000-year history.* Princeton Science Library. Princeton, NJ: Princeton University Press, 2011.

[91] H. Markowitz. Portfolio selection. *Journal of Finance*, 7:77–91, 1952.

[92] L. Meier, S. van de Geer, and P. Buhlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society, Series B*, 70(1):53–71, 2008.

[93] N. Meinshausen and P. Buhlmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34 3:1436–1462, 2006.

[94] C.A. Micchelli, J.M. Morales, and M. Pontil. A family of penalty functions for structured sparsity. In *Advances in Neural Information Processing Systems*, 2010.

[95] J. P. Morgan. Risk metrics- technical, technical document, 1994.

[96] A. Nemirovski and D. Yudin. *Problem complexity and method efficiency in optimization*. Wiley, 1983.

[97] E. Ordentlich and T. Cover. Online portfolio selection. In *Proceedings of the 9th Annual Conference on Learning Theory*, 1996.

[98] M. Pritsker. Evaluating value ar rick methodologies. *Journal od Financial Services Research*, 12(2-3):201–242, 1997.

[99] A. Quattoni, X. Carreras, M. Collins, and T. Darrell. An efficient projection for $\ell_{1,\infty}$ regularization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[100] D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, and P. Sabeti. Detecting novel associations in large data sets. *Science*, 334(6062):1518–1524, 2011.

[101] R. Rockafellar. Nonsmooth optimization. *Mathematical Programming: State of the Art*, pages 248–258, 1994.

[102] R. T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics. Princeton University Press, 1970.

[103] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2(3):21–41, 2000.

[104] M. Rodriguez, J. Leskovec, and B. Schölkopf. Structure and dynamics of information pathways in online media. In *International Conference on Web Search and Data Mining (WSDM)*, pages 23–32. ACM, 2013.

[105] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear Total Variation Based Noise Removal Algorithms. *Physica D*, 60:259–268, 1992.

[106] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *Proceedings of the Conference on Learning Theory (COLT)*, 2009.

[107] W. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *Journal of Finance*, 19(3):425–442, 1964.

[108] W. Sharpe. A theory of market equilibrium. *Journal of Finance*, 19:425–442, 1964.

[109] W. Sharpe. Mutual fund performance. *The Journal of Business*, 39(1):119–138, 1966.

[110] F. Sortino and L. Price. Performance measurement in a downside risk framework. *Journal of Investing*, 3(3):35–42, 1994.

[111] T. Soule. Voting teams: A cooperative approach to non-typical problems. *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 916–922, 1999.

[112] M. Steinbach. Markowitz revisited: Mean-variance models in financial portfolio analysis. *SIAM Review*, 43(1):31, 2001.

[113] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., 2005.

[114] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc. B*, 58:267–288, 1996.

[115] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society Series B*, pages 91–108, 2005.

[116] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. of Optim. Th. and Applcns.*, 109(3):475–494, 2001.

[117] B. Wahlberg, S. Boyd, M. Annergren, and Y. Wang. An admm algorithm for a class of total variation regularized estimation problems. In *SYSID*, 2012.

[118] M. Wainwright, P. Ravikumar, and J. Lafferty. High-dimensional graphical model selection using ℓ_1-regularized logistic regression. *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[119] H. Wang and A. Banerjee. Online alternating direction method. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[120] H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. *arXiv report*, 2013.

[121] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.

[122] W. Yan, M. Sewell, and C. D. Clack. Learning to optimize profits beats predicting returns – comparing techniques for financial portfolio optimisation. *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1681–1688, 2008.

[123] Y. Yu. Better approximation and faster algorithm using the proximal average. In *Advances in Neural Information Processing Systems*, pages 458–466, 2013.

[124] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67, 2006.

[125] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468–3497, 2009.

[126] S. Zhou, J. D. Lafferty, and L. A. Wasserman. Time varying undirected graphs. *Machine Learning*, 80(2-3):295–319, 2010.

[127] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. *Proceedings of the International Conference on Machine Learning (ICML)*, 2003.

Appendix A

Online Lazy Updates

A.1 Proofs of Lemma 1 and Lemma 2

In this section, we give a detailed proof of Lemma 1, which is used for the proof of Theorem 1. Then, we give a detailed proof of Lemma 2, which was used for Theorem 2.

A.1.1 Proof of Lemma 1

Lemma 1. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* $d_\psi(\cdot,\cdot)$ *is* λ*-strongly convex with respect to norm* $||\cdot||$*, i.e.* $d_\psi(\mathbf{p},\hat{\mathbf{p}}) \geq \frac{\lambda}{2}\|\mathbf{p}-\hat{\mathbf{p}}\|_2^2$ *and* $||\mathbf{p}-\hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p},\hat{\mathbf{p}} \in \mathcal{P}$*. Then, for any* $\mathbf{p}^* \in \mathcal{P}$*,*

$$\eta[\phi_t(\mathbf{p}_t) + \gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - \phi_t(\mathbf{p}^*)] \leq d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\gamma L + \frac{\eta^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2. \tag{3.14}$$

Proof. Let $h_{\mathbf{p}_t}(\mathbf{p}) = \|\mathbf{p}-\mathbf{p}_t\|_1$ and let $g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) \in \partial h_{\mathbf{p}_t}(\mathbf{p}_{t+1})$. Then, for any $\mathbf{p} \in \mathcal{P}$, the optimality condition for (4.21) can be written as

$$\langle \mathbf{p}-\mathbf{p}_{t+1}, \eta\nabla\phi_t\lambda(\mathbf{p}_t) + \eta\gamma g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) + \nabla\psi(\mathbf{p}_{t+1}) - \nabla\psi(\mathbf{p}_t)\rangle \geq 0\ .$$

Further, by convexity we have

$$\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) \leq \langle \mathbf{p}_t - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle \ , \tag{A.1}$$

$$\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \|\mathbf{p}^* - \mathbf{p}_t\|_1 \leq \langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \tag{A.2}$$

$$. \tag{A.3}$$

Hence, for any $\mathbf{p}^* \in \mathcal{P}$

$$\begin{aligned}
&\eta[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \gamma\|\mathbf{p}^* - \mathbf{p}_t\|_1] \\
&\quad \leq \eta\langle \mathbf{p}_t - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle + \eta\gamma\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\quad = \eta\langle \mathbf{p}_{t+1} - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle + \eta\gamma\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle \\
&\quad = \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\psi(\mathbf{p}_t) - \nabla\psi(\mathbf{p}_{t+1}) - \eta\nabla\phi_t(\mathbf{p}_t) - \eta\gamma g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\qquad\qquad + \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\psi(\mathbf{p}_{t+1}) - \nabla\psi(\mathbf{p}_t)\rangle + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle
\end{aligned}$$

First term of last equation is non-positive. Thus we have,

$$\begin{aligned}
&\eta[\phi_t(\mathbf{p}_t) + \phi_t(\mathbf{p}^*) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \gamma\|\mathbf{p}^* - \mathbf{p}_t\|_1] \\
&\quad \leq \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\psi(\mathbf{p}_{t+1}) - \nabla\psi(\mathbf{p}_t)\rangle + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle \\
&\quad = d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle \\
&\quad = d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \eta\langle \sqrt{\frac{\lambda}{\eta}}(\mathbf{p}_t - \mathbf{p}_{t+1}), \sqrt{\frac{\eta}{\lambda}}\nabla\phi_t(\mathbf{p}_t)\rangle \\
&\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \frac{\lambda}{2}||\mathbf{p}_t - \mathbf{p}_{t+1}||^2 + \frac{\eta^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2 \\
&\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \frac{\eta^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2 \ ,
\end{aligned}$$

where we have used $d_\psi(p_{t+1}, \mathbf{p}_t) \geq \frac{\lambda}{2}\|\mathbf{p}_{t+1} - \mathbf{p}_t\|^2$. Rearranging the terms, we have (4.22).

□

A.1.2 Proof of Lemma 2

Lemma 2. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* $d_\psi(\cdot, \cdot)$ *is* λ*-strongly convex with respect to norm* $||\cdot||$*, i.e.* $d_\psi(\mathbf{p}, \hat{\mathbf{p}}) \geq \frac{\lambda}{2}\|\mathbf{p} - \hat{\mathbf{p}}\|_2^2$ *and* $||\mathbf{p} - \hat{\mathbf{p}}||_1 \leq$

$L, \forall \mathbf{p}, \hat{\mathbf{p}} \in \mathcal{P}$. *Assuming* ϕ_t *are all* β*-strongly convex, for any* $\gamma < \frac{\beta}{4}$ *and any* $\mathbf{p}^* \in \mathcal{P}$*, we have*

$$\begin{aligned}&\eta_t[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1] \\ &\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2 - \eta_t\left(\frac{\beta}{2} - 2\gamma\right)\|\mathbf{p}^* - \mathbf{p}_t\|^2 \ .\end{aligned} \tag{3.17}$$

Proof. Let $h_{\mathbf{p}_t}(\mathbf{p}) = \|\mathbf{p} - \mathbf{p}_t\|_1$ and let $g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) \in \partial h_{\mathbf{p}_t}(\mathbf{p}_{t+1})$. Then, for any $\mathbf{p} \in \mathcal{P}$, the optimality condition for (4.21) can be written as

$$\langle \mathbf{p} - \mathbf{p}_{t+1}, \eta_t\nabla\phi_t(\mathbf{p}_t) + \eta_t\gamma g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) + \nabla\psi(\mathbf{p}_{t+1}) - \psi(\mathbf{p}_t)\rangle \geq 0 \ .$$

Further, by convexity we have

$$\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) + \frac{\beta}{2}\|\mathbf{p}^* - \mathbf{p}_t\|^2 \leq \langle \mathbf{p}_t - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle \ , \tag{A.4}$$

$$\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \|\mathbf{p}^* - \mathbf{p}_t\|_1 \leq \langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \tag{A.5}$$

$$. \tag{A.6}$$

Hence, for any $\mathbf{p}^* \in \mathcal{P}$

$$\begin{aligned}&\eta_t[\phi_t(\mathbf{p}_t) - \phi_t(\mathbf{p}^*) + \gamma\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \gamma\|\mathbf{p}^* - \mathbf{p}_t\|_1 + \frac{\beta}{2}\|\mathbf{p}^* - \mathbf{p}_t\|^2] \\ &\quad \leq \eta_t\langle \mathbf{p}_t - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle + \eta_t\gamma\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\ &\quad = \eta_t\langle \mathbf{p}_{t+1} - \mathbf{p}^*, \nabla\phi_t(\mathbf{p}_t)\rangle + \eta_t\gamma\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle + \eta_t\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle \\ &\quad = \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\psi(\mathbf{p}_t) - \nabla\psi(\mathbf{p}_{t+1}) - \eta_t\nabla\phi_t(\mathbf{p}_t) - \eta_t\gamma g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\ &\qquad + \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\psi(\mathbf{p}_{t+1}) - \nabla\psi(\mathbf{p}_t)\rangle + \eta_t\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla\phi_t(\mathbf{p}_t)\rangle\end{aligned}$$

First term of last equation is non-positive. Thus we have,

$$\begin{aligned}
&\eta_t[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}^*)+\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\gamma\|\mathbf{p}^*-\mathbf{p}_t\|_1+\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2]\\
&\leq \langle \mathbf{p}^*-\mathbf{p}_{t+1},\nabla\psi(\mathbf{p}_{t+1})-\nabla\psi(\mathbf{p}_t)\rangle+\eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1},\nabla\phi_t(\mathbf{p}_t)\rangle\\
&= d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1},\nabla\phi_t(\mathbf{p}_t)\rangle\\
&= d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\eta_t\langle \sqrt{\frac{\lambda}{\eta_t}}(\mathbf{p}_t-\mathbf{p}_{t+1}),\sqrt{\frac{\eta_t}{\lambda}}\nabla\phi_t(\mathbf{p}_t)\rangle\\
&\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\lambda}{2}||\mathbf{p}_t-\mathbf{p}_{t+1}||^2+\frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2\\
&\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2\ ,
\end{aligned}$$

where we have used $d_\psi(p_{t+1},\mathbf{p}_t)\geq\frac{\lambda}{2}\|\mathbf{p}_{t+1}-\mathbf{p}_t\|^2$. Rearranging terms, we have

$$\begin{aligned}
&\eta_t[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}^*)+\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1]\\
&\quad\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2+\gamma\|\mathbf{p}^*-\mathbf{p}_t\|_1-\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2\ .
\end{aligned}$$

Now, using the fact that for any $u\in\mathbb{R}$, $|u|\leq 2u^2$, we have

$$\|\mathbf{p}^*-\mathbf{p}_t\|_1=\sum_{i=1}^d|\mathbf{p}^*(i)-\mathbf{p}_t(i)|\leq 2\sum_{i=1}^d(\mathbf{p}^*(i)-\mathbf{p}_t(i))^2=2\|\mathbf{p}^*-\mathbf{p}_t\|^2.$$

Hence, for any $\gamma<\beta/4$, we have $\gamma\|\mathbf{p}^*-\mathbf{p}_t\|_1<\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2$, implying

$$\begin{aligned}
&\eta_t[\phi_t(\mathbf{p}_t)-\phi_t(\mathbf{p}^*)+\gamma\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1]\\
&\quad\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\eta_t^2}{2\lambda}||\nabla\phi_t(\mathbf{p}_t)||^2-\eta_t\left(\frac{\beta}{2}-2\gamma\right)\|\mathbf{p}^*-\mathbf{p}_t\|^2\ .
\end{aligned}$$

That completes the proof. □

A.2 Other Experimental Results

Here we present additional experimental results with the OLU algorithm, which have been omitted from the main text. Figure A.1(b), shows the histogram plot of the number of trades (computed by the ℓ_0 norm) for the NYSE and the S&P500 datasets. We observe, that as α increases the number of trades decreases monotonically for the NYSE dataset. However, the number of trades does not decrease monotonically as we increase α for the S&P500 dataset.

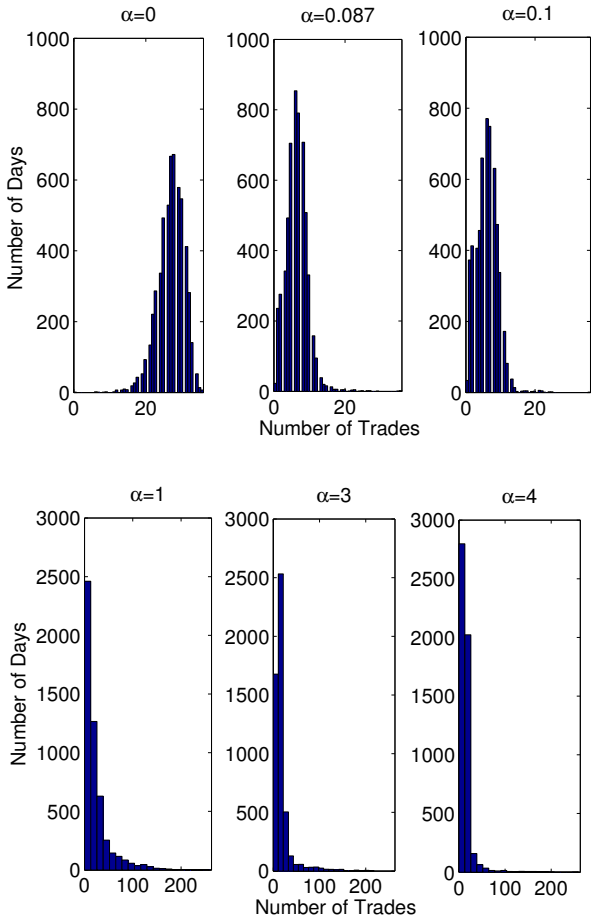


Figure A.1: Histogram plots for number of trades with varying α. As α increases the number of trades decreases monotonically for the NYSE dataset. The number of trades does not decrease monotonically as we increase α for the S&P500 dataset.

Appendix B

Online Portfolio Selection with Group Sparsity

B.1 Proof of the Lemma 3 and Lemma 4

In this section, we provide detailed proofs of Lemmas 3 and 4, which have been used in Theorems 4 and 5 respectively.

B.1.1 Proof of Lemma 3

Lemma 3. *Let the sequence of* $\{\mathbf{p}_t\}$ *be defined by the update in (4.21). Let* $d_\psi(\cdot,\cdot)$ *is* α*-strongly convex with respect to norm* $||\cdot||$*, i.e.* $d_\psi(\mathbf{p},\hat{\mathbf{p}}) \geq \frac{\alpha}{2}\|\mathbf{p}-\hat{\mathbf{p}}\|_2^2$ *and* $||\mathbf{p}-\hat{\mathbf{p}}||_1 \leq L, \forall \mathbf{p},\hat{\mathbf{p}} \in \mathcal{P}$*. Then, for any* $\mathbf{p}^* \in \mathcal{P}$*,*

$$\eta[f_t(\mathbf{p}_t) + r(\mathbf{p}_{t+1}) + \lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1] - \eta[f_t(\mathbf{p}^*) + r(\mathbf{p}^*)] \\ \leq d_\psi(\mathbf{p}^*,\mathbf{p}_t) - d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1}) + \eta\lambda_2 L + \frac{\eta^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2. \tag{4.22}$$

Proof. Let $\mathbf{r}'(\mathbf{p}_{t+1}) \in \partial\mathbf{r}(\mathbf{p}_{t+1})$ and $h_{\mathbf{p}_t}(\mathbf{p}) = \|\mathbf{p}-\mathbf{p}_t\|_1$ and let $g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) \in \partial h_{\mathbf{p}_t}(\mathbf{p}_{t+1})$. Then, for any $\mathbf{p} \in \mathcal{P}$, the optimality condition for (4.21) can be written as

$$\langle \mathbf{p}-\mathbf{p}_{t+1}, \eta\nabla f_t(\mathbf{p}_t) + \eta r'(\mathbf{p}_{t+1}) + \eta\lambda_2 g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) + \nabla\phi(\mathbf{p}_{t+1}) - \phi(\mathbf{p}_t)\rangle \geq 0 \ .$$

Sector	Description
Consumer Discretionary	Automotive, household durable goods, hotels, restaurants
Consumer Staples	Manufacturing/distributing of food, personal products, etc.
Energy	Construction/provision of oil rigs, drilling, and transportation
Financials	Banking, consumer/mortgage finance, investments, etc.
Health Care	Health care product manufacturing/supply
Industrials	Manufacturing/distributing capital goods, transportation services
Information Tech	Hardware/Software manufacturing/distributing
Materials	Commodity-related manufacturing
Utilities	Electric, gas, or water utility production/distribution

Table B.1: Sector information from sectors represented in the datasets.

Further, by convexity we have

$$f_t(\mathbf{p}_t) - f_t(\mathbf{p}^*) \leq \langle \mathbf{p}_t - \mathbf{p}^*, \nabla f_t(\mathbf{p}_t) \rangle \ , \tag{B.1}$$

$$r(\mathbf{p}_{t+1}) - r(\mathbf{p}^*) \leq \langle \mathbf{p}_{t+1} - \mathbf{p}^*, r'(\mathbf{p}_{t+1}) \rangle \tag{B.2}$$

$$\|\mathbf{p}_{t+1} - \mathbf{p}_t\|_1 - \|\mathbf{p}^* - \mathbf{p}_t\|_1 \leq \langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1}) \rangle \tag{B.3}$$

$$. \tag{B.4}$$

Hence, for any $\mathbf{p}^* \in \mathcal{P}$

$$\begin{aligned}
&\eta[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1 - \lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1] \\
&\quad \leq \eta\langle \mathbf{p}_t - \mathbf{p}^*, \nabla f_t(\mathbf{p}_t)\rangle + \eta\langle \mathbf{p}_{t+1} - \mathbf{p}^*, r'(\mathbf{p}_{t+1})\rangle + \eta\lambda_2\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\quad = \eta\langle \mathbf{p}_{t+1} - \mathbf{p}^*, \nabla f_t(\mathbf{p}_t)\rangle + \eta\langle \mathbf{p}_{t+1} - \mathbf{p}^*, r'(\mathbf{p}_{t+1})\rangle + \eta\lambda_2\langle \mathbf{p}_{t+1} - \mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\qquad\qquad + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle \\
&\quad = \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_t) - \nabla\phi(\mathbf{p}_{t+1}) - \eta\nabla f_t(\mathbf{p}_t) - \eta r'(\mathbf{p}_{t+1}) - \eta\lambda_2 g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\qquad\qquad + \langle \mathbf{p}^* - \mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_{t+1}) - \nabla\phi(\mathbf{p}_t)\rangle + \eta\langle \mathbf{p}_t - \mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle
\end{aligned}$$

First term of last equation is non-positive. Thus we have,

$$\begin{aligned}
&\eta[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1]\\
&\leq \langle \mathbf{p}^*-\mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_{t+1})-\nabla\phi(\mathbf{p}_t)\rangle+\eta\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle\\
&= d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\eta\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle\\
&= d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\eta\langle \sqrt{\frac{\alpha}{\eta}}(\mathbf{p}_t-\mathbf{p}_{t+1}), \sqrt{\frac{\eta}{\alpha}}\nabla f_t(\mathbf{p}_t)\rangle\\
&\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1},\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\alpha}{2}||\mathbf{p}_t-\mathbf{p}_{t+1}||^2+\frac{\eta^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2\\
&\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\eta^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2\ ,
\end{aligned}$$

where we have used $d_\psi(p_{t+1},\mathbf{p}_t)\geq\frac{\alpha}{2}\|\mathbf{p}_{t+1}-\mathbf{p}_t\|^2$. Rearranging the terms, we have (4.22). □

B.1.2 Proof of Lemma 4

Lemma 4. *Let the sequence of $\{\mathbf{p}_t\}$ be defined by the update in (4.21). Let $d_\psi(\cdot,\cdot)$ is α-strongly convex with respect to norm $||\cdot||$, i.e. $d_\psi(\mathbf{p},\hat{\mathbf{p}})\geq\frac{\alpha}{2}\|\mathbf{p}-\hat{\mathbf{p}}\|_2^2$ and $||\mathbf{p}-\hat{\mathbf{p}}||_1\leq L, \forall\mathbf{p},\hat{\mathbf{p}}\in\mathcal{P}$. Assuming f_t are all β-strongly convex, for any $\lambda_2<\frac{\beta}{4}$ and any $\mathbf{p}^*\in\mathcal{P}$, we have*

$$\begin{aligned}
&\eta_t[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1]\\
&\leq d_\psi(\mathbf{p}^*,\mathbf{p}_t)-d_\psi(\mathbf{p}^*,\mathbf{p}_{t+1})+\frac{\eta_t^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2-\left(\frac{\beta}{2}-2\lambda_2\right)\|\mathbf{p}^*-\mathbf{p}_t\|^2\ .
\end{aligned}\tag{4.25}$$

Proof. Let $\mathbf{r}'(\mathbf{p}_{t+1})\in\partial\mathbf{r}(\mathbf{p}_{t+1})$ and $h_{\mathbf{p}_t}(\mathbf{p})=\|\mathbf{p}-\mathbf{p}_t\|_1$ and let $g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\in\partial h_{\mathbf{p}_t}(\mathbf{p}_{t+1})$. Then, for any $\mathbf{p}\in\mathcal{P}$, the optimality condition for (4.21) can be written as

$$\langle \mathbf{p}-\mathbf{p}_{t+1}, \eta_t\nabla f_t(\mathbf{p}_t)+\eta_t r'(\mathbf{p}_{t+1})+\eta_t\lambda_2 g_{\mathbf{p}_t}(\mathbf{p}_{t+1})+\nabla\phi(\mathbf{p}_{t+1})-\phi(\mathbf{p}_t)\rangle\geq 0\ .$$

Further, by convexity we have

$$f_t(\mathbf{p}_t)-f_t(\mathbf{p}^*)+\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2\leq\langle \mathbf{p}_t-\mathbf{p}^*, \nabla f_t(\mathbf{p}_t)\rangle\ ,\tag{B.5}$$

$$r(\mathbf{p}_{t+1})-r(\mathbf{p}^*)\leq\langle \mathbf{p}_{t+1}-\mathbf{p}^*, r'(\mathbf{p}_{t+1})\rangle\tag{B.6}$$

$$\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\|\mathbf{p}^*-\mathbf{p}_t\|_1\leq\langle \mathbf{p}_{t+1}-\mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle\tag{B.7}$$

$$.\tag{B.8}$$

Hence, for any $\mathbf{p}^* \in \mathcal{P}$

$$\begin{aligned}
&\eta_t[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t+\|_1-\lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1+\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2] \\
&\leq \eta_t\langle \mathbf{p}_t-\mathbf{p}^*, \nabla f_t(\mathbf{p}_t)\rangle + \eta_t\langle \mathbf{p}_{t+1}-\mathbf{p}^*, r'(\mathbf{p}_{t+1})\rangle + \eta_t\lambda_2\langle \mathbf{p}_{t+1}-\mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&= \eta_t\langle \mathbf{p}_{t+1}-\mathbf{p}^*, \nabla f_t(\mathbf{p}_t)\rangle + \eta_t\langle \mathbf{p}_{t+1}-\mathbf{p}^*, r'(\mathbf{p}_{t+1})\rangle + \eta_t\lambda_2\langle \mathbf{p}_{t+1}-\mathbf{p}^*, g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\qquad + \eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle \\
&= \langle \mathbf{p}^*-\mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_t)-\nabla\phi(\mathbf{p}_{t+1})-\eta_t\nabla f_t(\mathbf{p}_t)-\eta_t r'(\mathbf{p}_{t+1})-\eta_t\lambda_2 g_{\mathbf{p}_t}(\mathbf{p}_{t+1})\rangle \\
&\qquad + \langle \mathbf{p}^*-\mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_{t+1})-\nabla\phi(\mathbf{p}_t)\rangle + \eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle
\end{aligned}$$

First term of last equation is non-positive. Thus we have,

$$\begin{aligned}
&\eta_t[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1-\lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1+\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2] \\
&\leq \langle \mathbf{p}^*-\mathbf{p}_{t+1}, \nabla\phi(\mathbf{p}_{t+1})-\nabla\phi(\mathbf{p}_t)\rangle + \eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle \\
&= d_\psi(\mathbf{p}^*, \mathbf{p}_t) - d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t) - d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1}) + \eta_t\langle \mathbf{p}_t-\mathbf{p}_{t+1}, \nabla f_t(\mathbf{p}_t)\rangle \\
&= d_\psi(\mathbf{p}^*, \mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t)-d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})+\eta_t\langle \sqrt{\frac{\alpha}{\eta_t}}(\mathbf{p}_t-\mathbf{p}_{t+1}), \sqrt{\frac{\eta_t}{\alpha}}\nabla f_t(\mathbf{p}_t)\rangle \\
&\leq d_\psi(\mathbf{p}^*, \mathbf{p}_t)-d_\psi(\mathbf{p}_{t+1}, \mathbf{p}_t)-d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})+\frac{\alpha}{2}||\mathbf{p}_t-\mathbf{p}_{t+1}||^2+\frac{\eta_t^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2 \\
&\leq d_\psi(\mathbf{p}^*, \mathbf{p}_t)-d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})+\frac{\eta_t^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2 \ ,
\end{aligned}$$

where we have used $d_\psi(p_{t+1}, \mathbf{p}_t) \geq \frac{\alpha}{2}\|\mathbf{p}_{t+1}-\mathbf{p}_t\|^2$. Rearranging terms, we have

$$\begin{aligned}
&\eta_t[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1] \\
&\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t)-d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})+\frac{\eta_t^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2+\lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1-\frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2 \ .
\end{aligned}$$

Now, using the fact that for any $u \in \mathbb{R}$, $|u| \leq 2u^2$, we have

$$\|\mathbf{p}^*-\mathbf{p}_t\|_1 = \sum_{i=1}^{d}|\mathbf{p}^*(i)-\mathbf{p}_t(i)| \leq 2\sum_{i=1}^{d}(\mathbf{p}^*(i)-\mathbf{p}_t(i))^2 = 2\|\mathbf{p}^*-\mathbf{p}_t\|^2.$$

Hence, for any $\lambda_2 < \beta/4$, we have $\lambda_2\|\mathbf{p}^*-\mathbf{p}_t\|_1 < \frac{\beta}{2}\|\mathbf{p}^*-\mathbf{p}_t\|^2$, implying

$$\begin{aligned}
&\eta_t[f_t(\mathbf{p}_t)+r(\mathbf{p}_{t+1})-f_t(\mathbf{p}^*)-r(\mathbf{p}^*)+\lambda_2\|\mathbf{p}_{t+1}-\mathbf{p}_t\|_1] \\
&\quad \leq d_\psi(\mathbf{p}^*, \mathbf{p}_t)-d_\psi(\mathbf{p}^*, \mathbf{p}_{t+1})+\frac{\eta_t^2}{2\alpha}||\nabla f_t(\mathbf{p}_t)||^2-\left(\frac{\beta}{2}-2\lambda_2\right)\|\mathbf{p}^*-\mathbf{p}_t\|^2 \ .
\end{aligned}$$

That completes the proof. □

B.2 Additional experimental results for Chapter 4

In this section, we present the experiments and figures that we have referred to in the main text. Table B.1 shows each of the sectors which we have used with a brief description and examples of 3 companies from each sector.

B.2.1 Effect of λ_1 for Group Sparsity ($\Omega(\mathbf{p})$)

Total Group Lasso: Figure B.1 plots histograms of the group lasso per day for increasing values of λ_1. As discussed in the main text, we see that the number of days with high group lasso decreases and the number of days with low group lasso increases as we increase λ_1.

Active Groups: Figure B.2 plots the number of active groups in which 80 of the portfolio was concentrated for the S&P500 dataset. As discussed in the main text, we can clearly see that number of active groups decrease as we increase λ_1.

Individual Group Weight: Figure B.3 illustrates how the individual weights at the sector level varies for the NYSE dataset as we increase λ_1. With a low value of $\lambda_1 = 0.01$, we observe that the selected 8 sectors are each invested in over the entire range of 22 years and there is high switching between sectors. With a slightly higher value of $\lambda_1 = 0.1$ we see that at any point of time, fewer sectors are still being invested in. Finally, with a high value of $\lambda_1 = 1$ we see significant group sparsity where only two sectors are invested in (Materials and Consumer Discretionary).

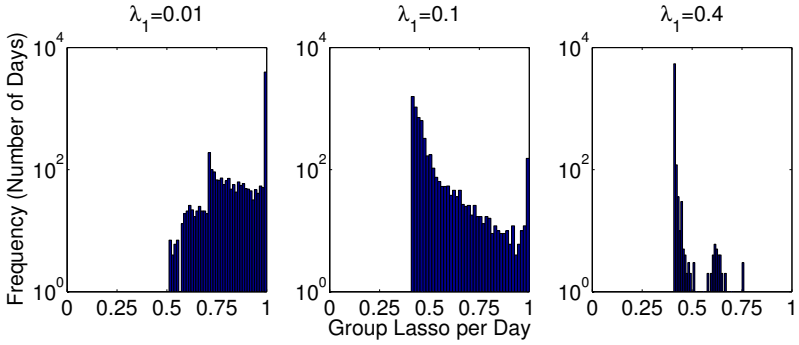


Figure B.1: Histogram of the group lasso value per day showing a move from more days with high group lasso value to more days with lower group lasso value for the NYSE dataset. There exists a gap between 0 and around 0.40 because the group lasso value depends on the size of each group, the larger the group with uniform weight the smaller the group lasso value. For this dataset the group lasso value is around 0.40 when the largest group has a near uniform portfolio.

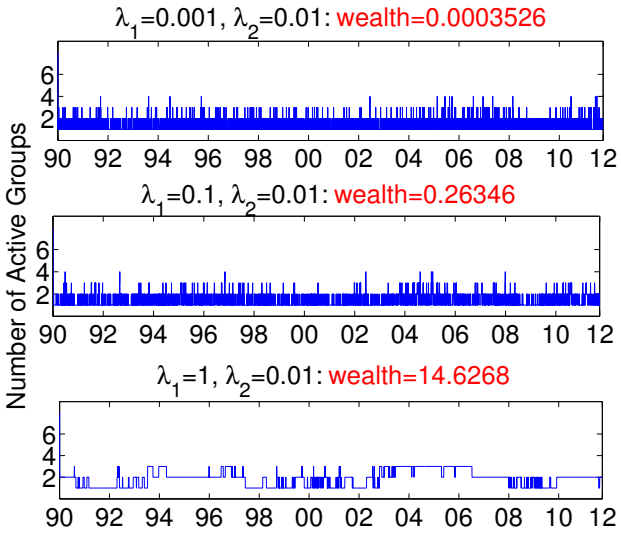


Figure B.2: Number of active groups per day where 80% of the portfolio is concentrated for the S&P500 dataset with increasing λ_1 values. We see that with $\lambda_1 = 1e{-}3$ at most 4 sectors are active. With $\lambda_1 = 0.1$ there are fewer days with 4 active sectors and 3 active sectors. With $\lambda_1 = 1$ most days have only 1 or 2 active sectors.

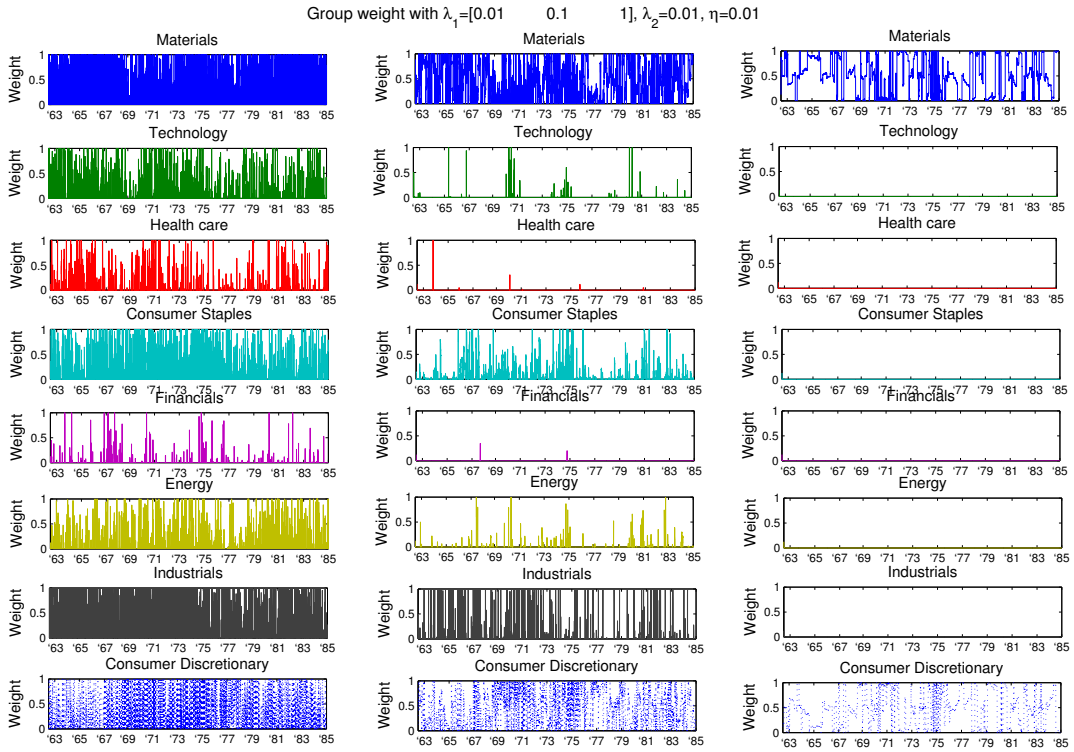


Figure B.3: Individual sector weights with varying λ_1, and fixed λ_2 and η for the NYSE dataset. All sectors except Financials are highly invested in for low $\lambda_1 = 0.01$ with frequent sector switches. As we increase λ_1 the number of sectors invested in and the frequency of sector switching decreases to where we only invest in 2 sectors (Materials and Consumer Discretionary).

B.2.2 Other Experimental Results

We present additional experimental results, which we have omitted from the main text. Figure B.4, plots the wealth with varying λ_2 and γ showing the trade-off between lazy updates (λ_2) and transaction costs (γ) with fixed $\lambda_1 = 0.2$ for the NYSE dataset. If we have a low λ_1 and γ then we allow frequent trading and can accumulate significant wealth because the transaction cost is low. With high γ our wealth goes to zero as we are forced to pay huge transaction cost penalties. With high λ_2 and low γ we accumulate

a moderate wealth due to our portfolio acting as a buy-and-hold and the same can be seen for high γ as we do not trade

Figure B.5, shows the effect of increasing λ_1 on the number of active groups for both the NYSE and S&P500 datasets. For both the datasets, we observe that increasing λ_1, decreases the number of active groups.

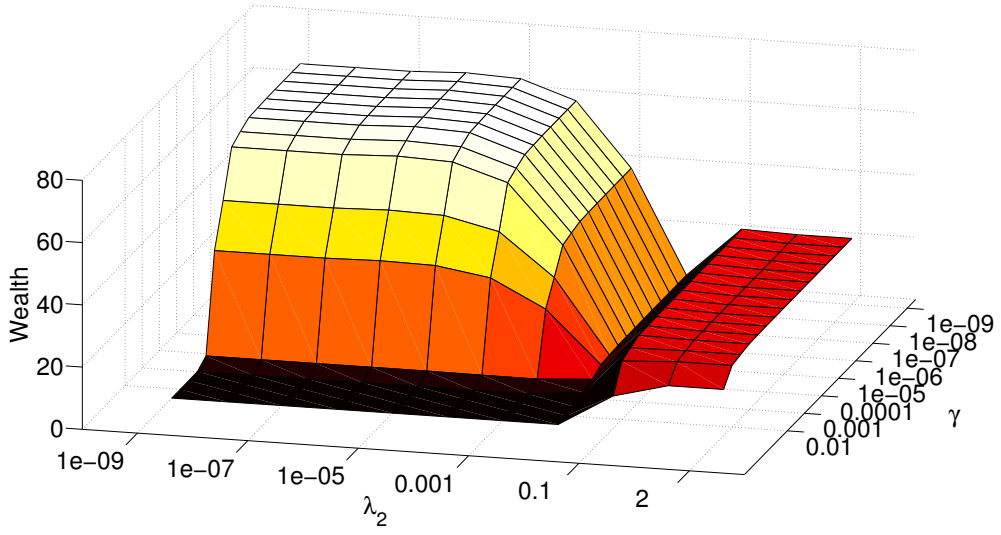
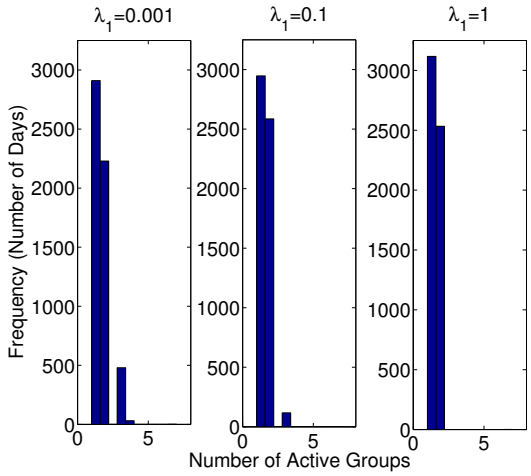
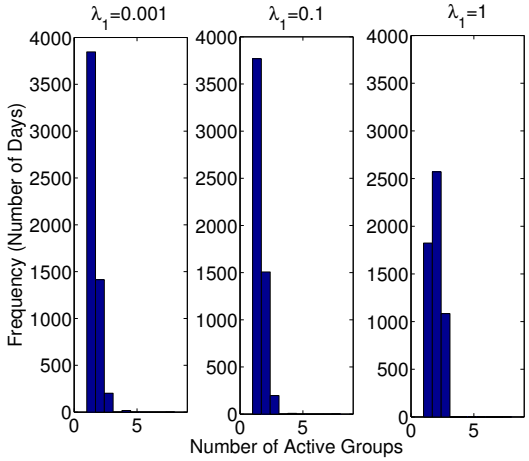


Figure B.4: 3D wealth plot with varying λ_2 and γ showing the trade-off between lazy updates (λ_2) and transaction costs (γ) with fixed $\lambda_1 = 0.2$ for the NYSE dataset. If we have a low λ_1 and γ then we allow frequent trading and can accumulate significant wealth because the transaction cost is low. With high γ our wealth goes to zero as we are forced to pay huge transaction cost penalties. With high λ_2 and low γ we accumulate a moderate wealth due to our portfolio acting as a buy-and-hold and the same can be seen for high γ as we do not trade.



(a) Histogram of the number of active groups with increasing λ_1 and fixed λ_2 and η for the NYSE dataset.



(b) Histogram of the number of active groups with increasing λ_1 and fixed λ_2 and η for the S&P500 dataset.

Figure B.5: Histogram of the active groups showing the number of active groups decrease as λ_1 increases.

Figure B.6, plots the individual sector weights for non-cyclic sector (Consumer Staples) and cyclic sectors (Information Tech and Financials) for the S&P500 dataset. We see the algorithm selects Consumer Staples during a bull market 1997-2000. We also

see that the Information Tech sector has increasing weight on it during this time (the dot-com bubble) and then the weight is rapidly decreased during the dot-com crash (2000-2002). It is interesting to note that the Financials sector also has increasing weight during the dot-com bubble and decreasing weight during the dot-com crash. Financials also has increased weight after the financial crisis of 2007-2009.

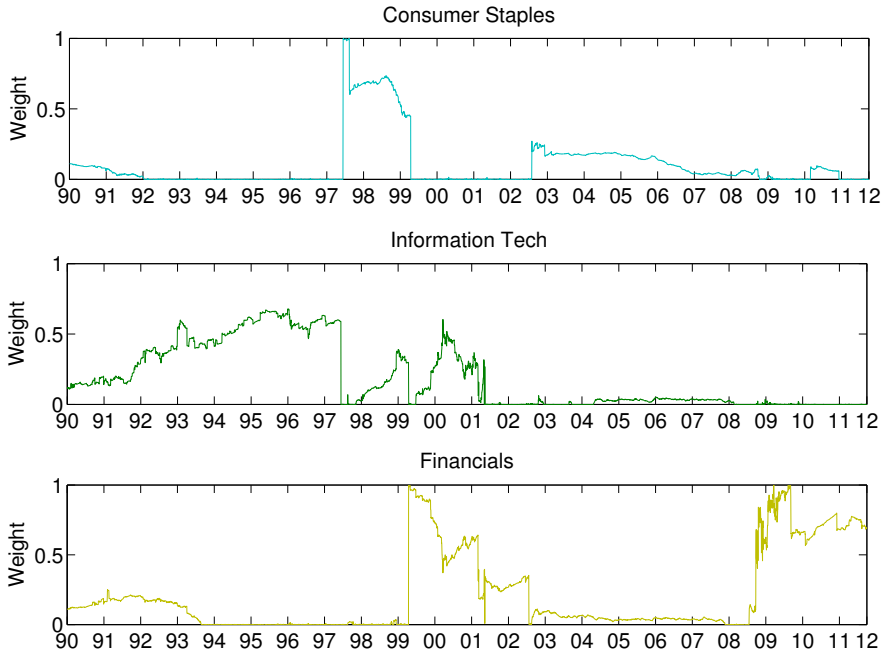


Figure B.6: Individual sector weights for non-cyclic sector (Consumer Staples) and cyclic sectors (Information Tech and Financials) for the S&P500 dataset. We see the algorithm selects Consumer Staples during a bull market 1997-2000. We also see that the Information Tech sector has increasing weight on it during this time (the dot-com bubble) and then the weight is rapidly decreased during the dot-com crash (2000-2002).

Appendix C

Online Convex Optimization with Constraints

C.1 Additional Experiments and Details

We now present a few additional results with RAMP which are particularly interesting. **Base Portfolios and parameters:** EG runs with a single parameter η_{EG} which was set to a value of 0.5 as proposed in [61]. For ONS, parameters β_{ONS}, η_{ONS} and δ_{ONS} which were set to 1, 0 and $\frac{1}{8}$ respectively as per [2]. Anticor was run with a window size of 30 days. Moreover, we present additional results with a variant of Anticor called Anticor$_{BAH}$. We add it to our base pool because of their exceptional empirical performance seen in [18, 40]. Anticor$_{BAH}$ with window sizes ranging from 2 to 30 [18, 40] was used.

Meta Portfolios (MP): We ran RAMP with two non-risk adjusted versions of meta portfolios: MP$_{GD}$ and additionally MP$_{EG}$ for comparison. We added MP$_{EG}$ to our pool of meta portfolios to get additional results. While MP$_{GD}$ uses gradient based update, MP$_{EG}$ [40] uses exponentiated gradient based update. For MP$_{GD}$ we tried out a few different learning rates and found that $\eta_{MP_{GD}} =\sim 0.2$ does well both for NYSE and S&P500. The learning rate $\eta_{MP_{EG}}$ for MP$_{EG}$ was set to 20 to reproduce results observed in [40]. RAMP was then run with the same learning rate and $\beta = 0.5$.

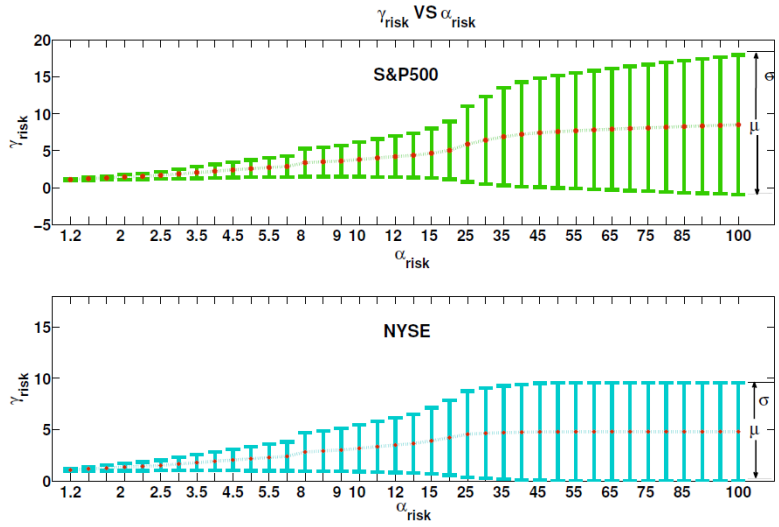


Figure C.1: RAMP: Mean and standard deviation of γ_{risk} for different α_{risk}. The standard deviation (σ) for γ_{risk} increases with the increase in α_{risk}.

Covariance estimation: The covariance matrix Σ_t is estimated as

$$\Sigma_t = \frac{\sum_{s=1}^{t-1} K(\frac{|s-t|}{h}) Z_s Z_s^T}{\sum_{s=1}^{t-1} K(\frac{|s-t|}{h})}, \tag{C.1}$$

$K(\frac{|s-t|}{h})$ is a symmetric non-negative function kernel over time and $Z_s = r_s - \mu_s$. μ_s is computed in the same fashion as a kernel estimator of the mean at time s. We use a RBF kernel for our experiments. The parameter h determines the time frame for the data(price relatives) taken into account for estimating the covariance matrix. For our experiments we chose h to be 50 which is approximately 2 months of trading days.

For our experiments we do a warm start, i.e. we start running RAMP, after we have d days of price relatives for constructing the first covariance matrix. This makes our estimated covariance matrix well-conditioned.

The offset d depends on the number of stocks n in the dataset and as such should be in the least greater than n. We choose $d = 50$ for the NYSE dataset and $d = 260$ for S&P500.

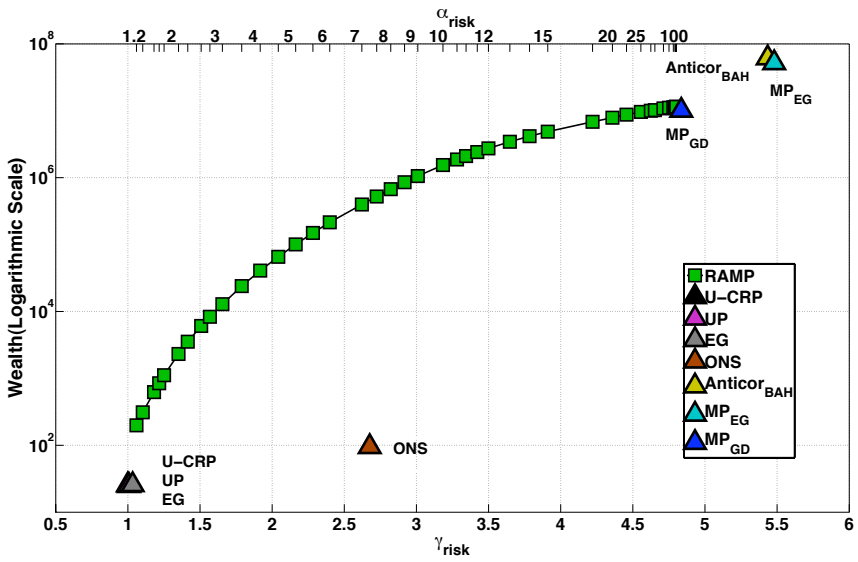
Permissible and True Risk: Figure C.1 shows the mean and standards deviation for γ_{risk} for the corresponding α_{risk} values. We notice two things. The mean for the γ_{risk} grows up to a certain point as α_{risk} increases, and then stabilizes. The greater the

permissible risk, the more the γ_{risk} can fluctuate.

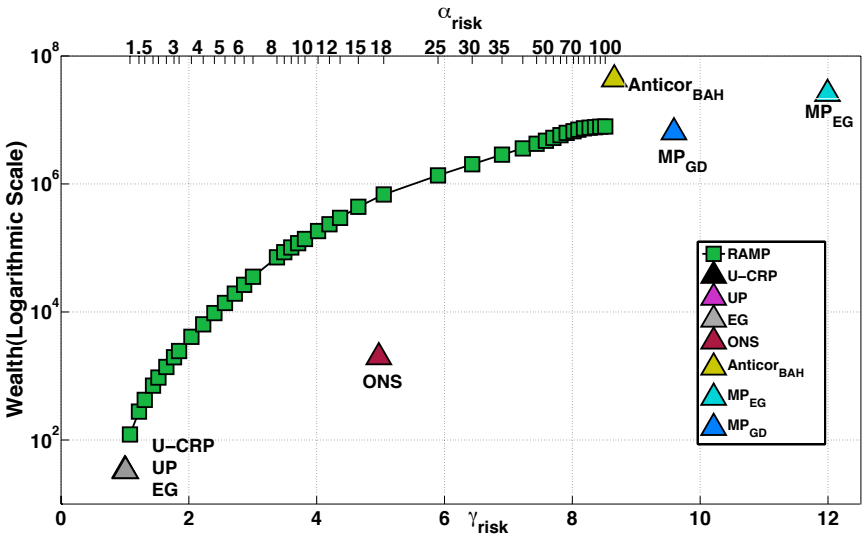
Table C.1: APY and mean γ_{risk} of RAMP, MPs and BPs.

Dataset		U-CRP	EG	RAMP	ONS	RAMP	Anticor_{30}	Anticor_{BAH}	MP_{EG}	MP_{GD}	RAMP
NYSE	APY	15.86	15.89	**27.19**	22.99	**62.03**	83.25	125.95	124.25	108.17	**109.35**
	γ_{risk}	1	1.0331	1.0595	2.6762	2.6218	12.6681	5.4344	5.4798	4.8357	4.8012
S&P500	APY	18.19	18.14	**25.79**	43.58	**82.24**	93.99	131.01	125.35	111.01	**113.09**
	γ_{risk}	1	1.0116	1.0818	4.9690	4.6493	29.1620	8.6481	11.9793	9.5793	8.50680

Comparison with BPs and MPs: Figure C.2 compares the wealth accumulated by the BPs, MPs and RAMP and their corresponding mean true risk (γ_{risk}). Here we present additional results with Anticor_{BAH} and the meta portfolio MP_{EG}. The wealth (in logarithmic scale) is plotted on the Y-axis. The mean γ_{risk} of each algorithm is plotted on the bottom X-axis and the permissible risk (α_{risk} values for RAMP can be found on the top X-axis. Table 2 supplements this information with the APY of the the algorithms for both the datasets. We observe that Anticor_{BAH} with a slightly higher true risk outperforms RAMP run with large α_{risk}. However, it is important to note here that Anticor_{BAH} is just a heuristic like Anticor and unlike RAMP has no performance (regret) guarantees. Thus for any risk aware investor, RAMP would be a natural choice. We also observe the performance of the non-risk adjusted MP_{EG}. We also observe that MP_{EG} which is non-risk adjusted and uses an EG update to determine the meta weights fares slightly better than RAMP for higher risk values. This prompts us to try an EG version of RAMP in our future work where we would use a KL-divergence as a proximal term instead of $\|\|\|^2$ as seen in (6.30) of Section 6.3 of the main section of the journal submission.



(a) Monetary returns and γ_{risk} for NYSE.



(b) Monetary returns and γ_{risk} for S&P500.

Figure C.2: For equivalent γ_{risk}, RAMP is either competitive or does better in terms of monetary returns (for \$1 investment shown here) with the BPs, MP $_{EG}$ and MP$_{GD}$ (best viewed in color).

Appendix D

Dynamic Gaussian Networks

D.1 Proof of Lemma 5

In this section we provide a sketch of the proof of Lemma 5.

Lemma 5. *Let us define* $G := \sup_{g \in \partial f_t(\Theta)} \|g\|_F$ *, where* $\partial f_t(\Theta)$ *denotes the subgradient set of* f_t *at* Θ*, and* $\Theta \succ 0$*. Then, at each step* t *of the algorithm,*

$$\eta_t[f_t(\Theta_t) - f_t(\Theta)] \leq \frac{\eta_t^2}{2} G^2 + \frac{1}{2}\left[\|\Theta - \Theta_t\|_F^2 - (1 + \eta_t \nu)\|\Theta - \Theta_{t+1}\|_F^2\right] , \tag{7.12}$$

where ν *is a strong convexity parameter of* f_t *,*$\forall t$ *.*

Proof. Since Θ_{t+1} is the solution, we are guaranteed to obtain a matrix $g_{t+1} \in \partial f_t(\Theta_{t+1})$ such that

$$0 = \eta_r g_{t+1} + \Theta_{t+1} - \Theta_t \ . \tag{D.1}$$

Since f_t is strongly convex with parameter ν, we have

$$\begin{aligned} f_t(\Theta) &\geq f_t(\Theta_{t+1}) + \langle \Theta - \Theta_{t+1}, g_{t+1} \rangle + \frac{\nu}{2}\|\Theta - \Theta_{t+1}\|^2 \\ \Rightarrow \eta_t[f_t(\Theta_{t+1}) - f_t(\Theta)] &\leq \langle \eta_t g_{t+1}, \Theta_{t+1} - \Theta \rangle - \frac{\eta_t \nu}{2}\|\Theta - \Theta_{t+1}\|^2 \ , \end{aligned} \tag{D.2}$$

where $\|\cdot\|$ represents a suitable matrix norm.

From (D.1) and (D.2) , we have

$$\begin{aligned}
&\eta_t[f_t(\Theta_{t+1}) - f_t(\Theta)] \\
&\leq \langle \Theta_t - \Theta_{t+1}, \Theta_{t+1} - \Theta \rangle - \frac{\eta_t \nu}{2} \|\Theta - \Theta_{t+1}\|^2 \\
&\leq \frac{1}{2}[\|\Theta - \Theta_t\|^2 - \|\Theta - \Theta_{t+1}\| - \|\Theta_t - \Theta_{t+1}\|^2] \\
&\quad - \frac{\eta_t \nu}{2} \|\Theta - \Theta_{t+1}\|^2 \ ,
\end{aligned} \tag{D.3}$$

Also, from the convexity of f_t , we can obtain a matrix $g_t \in \partial f_t(\Theta_t)$, such that

$$\begin{aligned}
f_t(\Theta_t) - f_t(\Theta_{t+1}) &\leq \langle g_t, \Theta_t - \Theta_{t+1} \rangle \\
\Rightarrow \eta_t[f_t(\Theta_t) - f_t(\Theta_{t+1})] &\leq \frac{\eta_t^2}{2} G^2 + \frac{1}{2} \|\Theta_t - \Theta_{t+1}\|^2
\end{aligned} \tag{D.4}$$

Adding (D.3) and (D.4) , we obtain (7.12). □

D.2 Proof of Theorem 10

In this section, we provide the proof of Theorem 10.

Theorem 10. *Let* $\{\Theta_1^*, \ldots, \Theta_T^*\}$ *be the sequence of precision matrices. Then, for* $\eta_t = \frac{1}{\nu t}$ *, the regret given by* (7.11) *is bounded as*

$$R_T \leq \frac{G^2}{2\nu}(1 + \log T) + \nu \nu_1 \sum_{t=1}^{T-1} \|\Theta_t^* - \Theta_{t-1}^*\|_q \ , \tag{7.13}$$

where $\|\cdot\|_p$ *and* $\|\cdot\| q$ *are Schatten matrix norms such that* $\frac{1}{p} + \frac{1}{q} = 1$ *and* $\|\Theta_t\|_p \leq \frac{\nu_1}{2}$ *,*$\forall t$ *.*

The bound is proved as follows.

$$\begin{aligned}
&\|\Theta_t^* - \Theta_t\|^2 - \|\Theta_t^* - \Theta_{t+1}\|^2 - \eta_t \nu \|\Theta_t^* - \Theta_{t+1}\|^2 \\
&= \left\{\|\Theta_t^* - \Theta_t\|^2 - \|\Theta_{t+1}^* - \Theta_{t+1}\|^2\right\} - \eta_t \nu \|\Theta_{t+1}^* - \Theta_{t+1}\|^2 \\
&\quad - (1 + \eta_t \nu) \left\{\|\Theta_t^* - \Theta_{t+1}\|^2 - \|\Theta_{t+1}^* - \Theta_{t+1}\|^2\right\} \\
&\overset{(b)}{\leq} \left\{\|\Theta_t^* - \Theta_t\|^2 - \|\Theta_{t+1}^* - \Theta_{t+1}\|^2\right\} - \eta_t \nu \|\Theta_{t+1}^* - \Theta_{t+1}\|^2 \\
&\quad - (1 + \eta_t \nu) \left\{\|\Theta_t^*\|^2 - \|\Theta_{t+1}^*\|^2 - \nu_1 \|\Theta_t^* - \Theta_{t+1}^*\|_q\right\} \ ,
\end{aligned} \tag{D.5}$$

where (b) follows using the matrix Hölder's Inequality:

$$\begin{aligned}\langle \Theta_{t+1}, \Theta_t^* - \Theta_{t+1}^* \rangle &:= \mathrm{Tr}(\Theta_{t+1}(\Theta_t^* - \Theta_{t+1}^*)) \\ &\leq \|\Theta_{t+1}\|_p \|\Theta_t^* - \Theta_{t+1}^*\|_q \leq \frac{\nu_1}{2} \|\Theta_t^* - \Theta_{t+1}^*\|_q \ ,\end{aligned} \tag{D.6}$$

where we have used Schatten matrix norms and $\frac{1}{p} + \frac{1}{q} = 1$.

By summing both sides of Lemma 5 after substituting (D.5) and noting that $(1 + \eta_t \nu) \geq \eta_t \nu$,we obtain

$$\begin{aligned}R_T &\leq \frac{G^2}{2} \sum_{t=1}^{T} \eta_t + \frac{1}{2} \sum_{t=1}^{T} \|\Theta_t - \Theta_t^*\|^2 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \nu \right) \\ &\quad - \nu \|\Theta_1^*\|^2 + \nu \nu_1 \sum_{t=1}^{T-1} \|\Theta_t^* - \Theta_{t+1}^*\|_q \\ &\overset{(b)}{\leq} \frac{G^2}{2\nu} (1 + \log T) + \nu \nu_1 \sum_{t=1}^{T-1} \|\Theta_t^* - \Theta_{t+1}^*\|_q \ ,\end{aligned} \tag{D.7}$$

where (b) follows by choosing $\eta_t = \frac{1}{\nu t}$. □