Bayesian Online Learning for Online Portfolio Selection

Arnold Salas

Wolfson College
University of Oxford

A thesis submitted for the degree of Doctor of Philosophy

Trinity 2020

Abstract

We propose a novel family of Bayesian learning algorithms for online portfolio selection that overcome many of the shortcomings of traditional techniques, including selection bias (the failure to cover a broad universe of assets), data-snooping bias (the risk that a trading strategy's performance on past data is inflated due to hyperparameter overfitting) and a lack of robustness to transaction costs.

As the basis for this novel family, we develop a Bayesian treatment of the online passive-aggressive and gradient descent algorithms, some of the most popular algorithms in the literature. Our approach starts from a probabilistic interpretation of the underlying objective functions and enables uncertainty modelling, probabilistic predictions as well as automatic, data-dependent hyperparameter tuning.

We conclude by testing our proposals on real-world financial data. We further benchmark our framework on a wide range of canonical test problems, over which it achieves a significant improvement on its competitors. Beyond online portfolio selection, our algorithms contribute to the theory of adaptive gradient methods by equipping these with uncertainty estimates and a self-tuning mechanism for the learning rate parameter, which constitutes a major milestone in the area of Bayesian inference for neural networks.

Bayesian Online Learning for Online Portfolio Selection

Arnold Salas

Wolfson College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2020

To Vicky, Johann, Jessi and Edith.

Acknowledgements

This work was jointly funded by an AFR PhD grant from the Luxembourg National Research Fund and an ESRC (Economic and Social Research Council) Doctoral Training Partnership in collaboration with the Oxford-Man Institute of Quantitative Finance. I shall be eternally grateful to these institutions, without whom this thesis would have never happened, especially to the Luxembourg National Research Fund for providing the largest share of funding.

Personally, I would like to thank:

My daughter Vicky, for inspiring me to apply to the University of Oxford and keeping me motivated to hang in there during all those tough years that we lived 6,300 miles apart;

My wife Jessi, for letting me live in her apartment for free while we were still dating and for believing in my success;

My mom Edith, for loaning me vast amounts of money without complaint and always supporting me in all my endeavours;

My son Johann, for giving me the second bout of enthusiasm that I needed to stay the course as this thesis was close to completion;

My supervisors Steve Roberts and Mike Osborne, for accepting me into the programme despite my non-traditional academic background, shielding me from all administrative formalities, allowing me complete academic freedom and not being too upset by how long this thesis process took;

Stefan Zohren, one of the examiners at the confirmation of my DPhil status, for identifying a close relationship between my research and adaptive gradient methods, which led to a coauthored paper on the training of Bayesian neural networks via such methods, as well as a thesis chapter on the use of Bayesian tools for the automatic determination of the learning rate parameter in the context of dynamic online optimisation.

My thesis is based on a couple of joint-authored publications listed below, followed by a description of my contribution to them.

> A. Salas, S. Roberts and M. Osborne. A Variational Bayesian State-Space Approach to Online Passive-Aggressive Regression. *CoRR*, abs/1509.02438, 2015.
>
> S. Kessler, A. Salas, V. W. C. Tan, S. Zohren and S. Roberts. Practical Bayesian Neural Networks via Adaptive Optimization Methods. In *ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

I contributed the idea, problem setup, all theory and experiments for the first work. Roberts implemented the sequential Kalman filter tested against, while Osborne supplied the wind speed data. Both Roberts and Osborne provided guidance on the general approach and inference schemes.

As for the second paper, Zohren came up with the idea and problem motivation. I proposed the Bayesian approach, and contributed the preliminaries in addition to the probabilistic interpretation of adaptive methods. Zohren carried out the MNIST classification experiment, while Kessler contributed additional experiments, including the application to contextual bandit problems. Additionally, Kessler reinforced the theoretical framework. Roberts provided advice on general theory and practical issues.

Contents

Part I

Overview

1 Introduction

Contents

Today, whether we consider data sets from the internet, consumers or 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. The field has attracted a lot of attention due to the recent emergence of large-scale applications such as online web advertisement placement, online topic detection, online web ranking, finding the 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, by treating the training data as a stream and processing each data object only once. In particular, one can solve a batch problem by processing one or a mini-batch of data points at a time, e.g.

in image classification, text categorisation, bioinformatics (protein classification, cancer classification), and so on.

Online algorithms and their stochastic counterparts share two key properties. Firstly, they are computationally efficient. Each step has no dependence on the data size and there is no 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 achieve theoretical performance guarantees that are competitive compared to their batch counterparts (which have access to the entire data). For all the aforementioned reasons, the study of online learning algorithms is an increasingly important area in machine learning, in addition to its interesting theoretical properties and practical applications.

1.1 Why Online Learning?

With the ever increasing amount of data, there has been a growing amount of interest in scalable machine learning algorithms over recent years. In this context, many common approaches fail, simply because they cannot load the data into memory, or they are not efficient enough. Below we discuss a few areas in which online learning is applied.

1. Social media are pervasive nowadays. Twitter receives over 400 million tweets per day from its users. As microblogging sites gained popularity, a myriad of trend analytics applications have emerged. An example of one such task is the automatic identification of breaking news in a stream of tweets. This requires the detection of novel tweets among a voluminous stream in a scalable manner. The high volume and velocity of such data make them an ideal candidate for online learning.

2. Supervised learning (e.g. logistic regression, support vector machines) on large data sets containing millions of data points can be computationally expensive. For example, the Google brain consists of 20 million images. Iterative methods running over an entire data set usually have a runtime that depends on the size

of the data, and hence the per-iteration complexity becomes a computational bottleneck. The alternative is to process one data point at a time, by treating the huge data set as a stream for online convex optimisation.

3. In financial data analysis, the data also arrive in a streaming fashion. One notable problem in this context is *online portfolio selection* which consists in sequentially distributing wealth among a universe of assets. At the beginning of every rebalancing period (e.g. a business day), an investor has to choose a portfolio without knowing the performance of the underlying assets for that particular period a priori. At the end of the period, the investor receives feedback from the market, in the form of each asset's return over that period, and incurs the gain or loss associated with her portfolio choice. The objective is to maximise her wealth at the end of the trading horizon, i.e. the total number of periods during which the investor intends to trade.

Online portfolio selection has been a success story [Cover, 1991; Helmbold et al., 1998; Cesa-Bianchi and Lugosi, 2006; Agarwal et al., 2006; Borodin et al., 2004; Li et al., 2011; Li and Hoi, 2014] over the last two decades. Online portfolio selection algorithms modelled in the online convex optimisation framework make no statistical assumptions regarding the movement of asset prices [Cover, 1991; Helmbold et al., 1998] and, in a well-defined technical sense, are guaranteed to be competitive relative to 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 the field of online learning.

1.2 Research Scope and Methodology

Our research is primarily focused on Bayesian inference in online supervised learning models. The majority of online algorithms are frequentist in nature, meaning they lack to provide a principled measure of uncertainty in the estimation of their underlying parameters. Furthermore, their practical performance relies heavily on a set of hyperparameters and there is usually no clear systematic guidance

on how to choose appropriate values for them, making it hard to practitioners to deploy these algorithms. The motivation behind this thesis is that a Bayesian treatment of online learning provides an elegant and principled framework to deal with these shortcomings.

Our approach to online Bayesian inference is standard in that it starts from a maximum a posteriori interpretation of the objective function that the underlying frequentist online algorithm optimises over its parameters. This not only allows us to obtain the posterior distribution that the algorithm implicitly imposes over its parameters, but also the implicit hyperparameter posterior, via the underlying marginal likelihood. As such, we are not only able to infer the uncertainty over model parameters, but also a data-driven mechanism to set optimal values for the hyperparameters. When no closed-form algorithm is available or efficient, we have taken the approach of analytical approximation, rather than stochastic approaches. In particular, we have not undertaken any work in the field of estimation of posteriors through sampling, since our main focus is on understanding the mathematical properties relating to exact probabilistic inference in online supervised learning. When exact inference is not possible, we aim to understand which are the mildest analytical approximation schemes that render the inference problem tractable.

Our contributions aim to be methodological with broad applicability; we have however placed our focus on the area of online portfolio selection towards the end of Part II. This has been motivated not only by the non-stationary nature of financial time series and the danger of overfitting to historical data, but also by the main author's background in finance and his desire to pursue a career in quantitative investment management following graduation.

1.3 Contribution

The contributions of this thesis can be ascribed to three main planks, each forming one of three main chapters. First, we discuss the limitations of online passive-aggressive learning, a popular framework in the field of online learning, and demonstrate how to address them. We extend the framework to allow general

loss functions (for both classification and regression), and introduce a unified Bayesian treatment that accommodates probabilistic predictions and automatic hyperparameter tuning. Remarkably, the resulting model bears a close ressemblance to the Kalman filter, bridging the gap between the seemingly disparate literatures of online learning and Bayesian filtering.

Second, we discuss the problem of tuning the learning rate parameter in online gradient descent, a key algorithm for solving online convex optimisitation as well as large-scale learning problems. By appealing to our earlier work on online Bayesian passive-aggressive learning, we derive a data-dependent mechanism to automatically tune the learning rate. This bypasses the requirement for traditional heuristics, and is applicable beyond the area of online convex optimisation, in particular in the training of neural networks, as discussed in a separate paper of which the main author of this thesis is a coauthor.

Third, we take a slightly different tack and present a novel approach to online portfolio selection. We show that our Bayesian-inspired approach in this framework is empirically superior to classical techniques, while remaining simple, flexible and computationally efficient.

1.4 Thesis Structure

The main body of this thesis is split into three parts. Part I includes this introductory chapter and goes on to set out significant background material relevant to the results of the proceeding work. Part II comprises the main novel analytical contribution of the thesis and is split into three chapters, including an empirical evaluation of the algorithms developed in the first three chapters. Taken together, the three planks of work are related under the broader umbrella of novel approaches to online learning and its application to portfolio selection. Finally, Part III ties the thesis together by providing summary conclusions of Part II and discusses some opportunities for further study. Several appendices are also included, setting out some key definitions and derivations.

1.5 Notation

We have tried to use a consistent notation throughout the thesis, although at times this means departing from some of the conventions used in the corresponding research literature. Variables are written as lower-case letters such as x. Vectors are denoted by lower-case bold Roman letters such as $\mathbf{x}$, and all vectors are assumed to be column vectors. A superscript T denotes the transpose of a matrix or vector, so that $\mathbf{x}^{\mathrm{T}}$ will be a row vector[1]. Upper-case bold roman letters, such as $\mathbf{M}$, denote matrices. For time series, the sequential index is given by a subscript such as x_t for the value of variable x at point t. When considering collections of the variable x over a set of indices $a, a+1, \ldots, b-1, b$, we abbreviate the series $x_a, \ldots, x_b \equiv x_{a:b}$. In the case $b < a$, we assume $x_{a:b} = \emptyset$, and also $x_a = \emptyset$ for all $a < 1$.

The notation $[a, b]$ is used to denote the closed interval from a to b, that is the interval including the values a and b themselves, while (a, b) denotes the corresponding open interval, that is the interval excluding a and b. Similarly, $[a, b)$ denotes an interval that includes a but excludes b. For the most part, however, there will be little need to dwell on such refinements as whether the end points of an interval are included or not. Another notational convetion that shall be common in this thesis is the symbol $[n]$ for a positive integer n, which shall signify the set $\{1, 2, \ldots, n\}$.

The $n \times n$ identity matrix (also known as the unit matrix) is denoted $\mathbf{I}_n$, which will be abbreviated to $\mathbf{I}$ where there is no ambiguity about its dimensionality. It has elements I_{ij} that equal 1 if $i = j$ and 0 if $i \neq j$. Similarly, $\mathbf{0}_{n\times m}$ denotes a $n \times m$ matrix all the entries of which are zero, while $\mathbf{1}_{n\times m}$ is a $n \times m$ matrix whose elements are all equal to 1. When it is clear from the context, these will be abbreviated to $\mathbf{0}$ and $\mathbf{1}$, respectively.

Throughout the thesis, the operator $p(\cdot)$ represents a probability density function in an intuitive way. The distribution of a random variable x conditioned on another

[1]This superscript shall mainly be used to denote the dot (inner) product between two vectors, i.e. $\mathbf{x}^{\mathrm{T}}\mathbf{y} \equiv \sum_{i=1}^{n} x_i y_i$ signifies the dot product between two n-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$. For this purpose, we shall also invariably use the symbol "$\cdot$", as in $\mathbf{x} \cdot \mathbf{y} \equiv \mathbf{x}^{\mathrm{T}}\mathbf{y}$.

random variable y is written as $p(x|y)$. Density functions, along with some interesting properties, for the relevant distributions are given in Appendix B.

A major consideration in Bayesian inference is marginalisation, or 'summing out' variables from a distribution. For continuous variables, we write this as

$$\int p(x, y)\,\mathrm{d}x = p(y). \tag{1.1}$$

When the variable of integration is an n-dimensional vector $\mathbf{x}$, we use the compact notation $\mathrm{d}\mathbf{x} = \mathrm{d}\mathbf{x}_1\mathrm{d}\mathbf{x}_2 \ldots \mathrm{d}\mathbf{x}_n$. When the context is clear, we shall use a condensed integral notation: all integrals are definite integrals over the entire domain of interest.

Finally, angled brackets represent expectation:

$$\langle f(x)\rangle \equiv \mathbb{E}[f(x)] = \int f(x)p(x)\,\mathrm{d}x. \tag{1.2}$$

1.6 A Caveat About Our 'Bayesian' Approach

Some readers may feel that the author overemphasised the Bayesian nature of the methods developed in this thesis. As a result of this, we woud like to clarify at this point that, by referring to an approach as 'Bayesian', we shall henceforth mean that such approach is *approximately* or *partially* Bayesian, in the sense that this approach does not involve marginalising out hyperparameter uncertainty, but resorts to point estimates instead, by optimising the relevant (approximate) marginal likelihood function.

2
Online Learning

Contents

2.1 Introduction

Online learning [Cesa-Bianchi and Lugosi, 2006; Shalev-Shwartz, 2011; Hazan, 2016] represents an important family of efficient and scalable algorithms for time-aware and large-scale applications. In general, these algorithms are fast, simple and based only on a handful of statistical assumptions, making them applicable to a wide range of settings. Online methods have therefore rapidly gained traction in the

field of big data analytics, where avoiding processing extremely large batches of data is important.

Online learning has been actively studied in several communities, including machine learning/artificial intelligence, statistics and game theory. Over the past years, a variety of online algorithms have been proposed. To a large extent, the design of these algorithms has been influenced by convex optimisation tools, whence their release under the umbrella of *online convex optimisation*. So far, however, the vast majority of these techniques are frequentist: they make point rather than probabilistic predictions, thus failing to account for model/prediction uncertainty. Furthermore, they rely on a few hyperparameters that must be hand-picked, which renders their predictive performance sensitive to specific hyperparameter configurations. This motivates a Bayesian approach to online learning that explicitly takes model/prediction uncertainty into account, and simultaneously enables automatic hyperparameter tuning by letting the data speak for themselves, which is precisely the raison d'être of this thesis.

Before diving in, however, it is helpful to review the existing literature that will serve as a building block for the methods developed herein. To this end, this chapter contains a brief survey of the relevant, so-called *first-order* and *second-order* online learning algorithms for linear regression — our main focus throughout this thesis. The survey here largely follows the publications of [Hoi, 2013] and [Hoi et al., 2018], which may be referred to for a deeper treatment of online learning.

2.1.1 Online vs offline learning

The traditional learning paradigm in machine learning and related disciplines is *offline learning*, also known as *batch learning*. Perhaps the main building block of offline learning is the assumption that the data are *stationary*, which justifies training a model once, on a batch of data, and repeatedly using it on subsequent observations.

Alas, in many real-world time-series applications, the underlying data generating mechanism keeps changing over time, thereby violating the assumption of stationarity. In this case, offline algorithms need to be retrained every time a

new datum is made available to them. This can be highly inefficient and can introduce costly delays in the decision-making process, particularly in time-aware applications, such as algorithmic trading where the data streams used as inputs by the learning machine may be imported at very high sampling frequencies (e.g. every minute or less) [Montana and Parrella, 2008]. Moreover, in large-scale applications, the dimensionality of the data makes it impossible to store them after they have been processed, due to the practical bounds on memory utilisation [Domingos and Hulten, 2003].

There a few additional reasons why online learning is different from offline/batch learning, namely:

- Offline/batch methods do not learn by minimising (static or dynamic) regret. They are based on alternative optimisation criteria, such as empirical risk minimisation or maximum likelihood estimation. That is why even when performed on a sliding- or rolling-window basis, most offline methods do not fall under the umbrella of 'online learning' technically speaking, albeit in practice, they can indeed be utilised for learning of time series data online, except perhaps in the circumstances outlined in the two bullet points below.

- The environment against which an online learner plays can be *adversarial*, meaning the losses the learner receives from the environment are time-varying and cannot be known in advance (as the environment reveals them only *after* the online learner has played her action).

- The feedback obtained from the environment in online learning can be *partial* rather than complete, meaning that at each round after the online learner has chosen her action $\mathbf{w}_t$, the environment does not reveal the entire loss function $\ell_t(\cdot)$, but only the gradient thereof at $\mathbf{w}_t$ or the loss value incurred $\ell_t(\mathbf{w}_t)$ at the chosen weights such as in the bandit setting (this partial feedback may even be noisy).

Having said that, it is worthwhile noting that the term 'online learning' exists in other communities outside the OCO and no-regret community.

As a result, in real-time forecasting applications, one often prefers online to offline learning, due to its numerous competitive advantages which, for completeness, are listed below:

- Avoid retraining when adding new examples;
- Fast, low memory footprint;
- Strong adaptability to changing environments;
- Simple to implement;
- Easy to be parallelised;
- Theoretical guarantees;
- Can be converted to batch learning.

2.1.2 Applications of online learning

For the aforementioned reasons, online learning is the favourite learning paradigm in streaming environments. Over recent years, it has gained considerable traction in big data mining. Further areas of application are depicted in Figure 2.1.

Figure 2.1: Some areas of application of online learning. Source: [Hoi, 2013].

Starting with social media, online learning is employed as a mining tool for social media streams to gather business intelligence, such as public emotion and product/brand sentiment. In internet security, it is used as an online anomaly detection mechanism, ranging from spam email detection to the identification of fraudulent credit card transactions. In computer vision, it enables real-time object tracking for video surveillance purposes, especially for the detection of anomalous events from realtime video streams. On the other hand, interactive image/video search via online relevance feedback is one of the various applications of online learning to multimedia search. Finally, in finance, online portfolio selection has recently been demonstrated to outperform traditional benchmarks [Li and Hoi, 2014]. As we already hinted, online methods are crucial in this field, as sequential decisions need to be promptly taken regarding the allocation of wealth among different assets.

2.2 Problem Statement and Related Theory

We first give a formal formulation of online linear regression, which will be our main subject of focus throughout this thesis, and then introduce the basics of statistical learning theory and online convex optimisation as the theoretical foundations for online learning techniques.

2.2.1 Online linear regression

Online learning operates on a sequence of data examples with timestamps. At each step t, the learner receives an instance $\mathbf{x}_t \in \mathbb{R}^n$. It first attempts to make a prediction for the output of the incoming instance which, in the case of linear regression, takes the form $\hat{y}_t = \mathbf{w}_t^{\mathrm{T}}\mathbf{x}_t$, where $\mathbf{w}_t \in \mathbb{R}^n$ is the incrementally learned weight vector. After making the prediction, the true output $y_t \in \mathbb{R}$ is revealed, and the learner then computes the incurred loss $\ell(y_t, \hat{y}_t) \in \mathbb{R}_{\geq 0}$, based on some criterion to measure the difference between the learner's prediction and the revealed true output. Using this loss, the learner finally decides whether and how to update the regression model at the end of each learning step.

The following algorithmic framework gives an overview of most first-order online learning algorithms for linear regression, where $\Delta(\mathbf{w}_t; (\mathbf{x}_t, y_t))$ denotes the update of the regression models. Different online learning algorithms operate under different definitions, and designs of the loss function $\ell(\cdot)$ and the updating function $\Delta(\cdot)$.

Algorithm 1 Online learning framework for linear regression

Initialisation: $\mathbf{w}_1 = \mathbf{0}_{n\times 1}$
for $t = 1, 2, \ldots, T$ **do**
  the learner receives an incoming instance $\mathbf{x}_t \in \mathbb{R}^n$
  the learner predicts the output: $\hat{y}_t = \mathbf{w}_t^{\mathrm{T}}\mathbf{x}_t$
  the true output is revealed by the environment: $y_t \in \mathbb{R}$
  the learner calculates the suffered loss: $\ell(\mathbf{w}_t; (\mathbf{x}_t, y_t))$
  if $\ell(\mathbf{w}_t; (\mathbf{x}_t, y_t)) > 0$ **then**
    the learner updates the regression model according to

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \Delta(\mathbf{w}_t; (\mathbf{x}_t, y_t))$$

  end if
end for

By running such an algorithm over a sequence of T rounds, the cumulative loss achieved by the algorithm can be measured as $L_T = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$. The classic goal in an online learning task is to minimise the regret of the underlying online learner's predictions against the best fixed model in hindsight, i.e.

$$R_T = \sum_{t=1}^{T} \ell(y_t, \mathbf{w}_t^{\mathrm{T}} \mathbf{x}_t) - \min_{\mathbf{w}} \sum_{t=1}^{T} \ell(y_t, \mathbf{w}^{\mathrm{T}} \mathbf{x}_t), \tag{2.1}$$

where the second term is the loss suffered by the optimal model $\mathbf{w}^*$, which can only be known in hindsight after seeing all the instances and their corresponding targets. From the theoretical perspective of regret minimisation, the notion of *no regret* means $R_T \leq b(T)$, where $b(T)$ in $o(T)$.

2.2.2 Statistical learning theory

Statistical learning theory, first introduced in the late 1960's, is one of key foundations for the theoretical analysis of machine learning problems, especially for supervised learning. There are many comprehensive survey articles and books on the subject matter, including the seminal work of Vladimir Vapnik [Vapnik, 1998, 1999]. In the following, we review some basic concepts.

Empirical error minimisation

Assume instance $\mathbf{x}_t$ is generated randomly from a fixed but unknown distribution $P(\mathbf{x})$, and that the corresponding output y is also generated from a fixed but unknown distribution $P(y|\mathbf{x})$. The joint distribution of the data is then $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$. The goal of supervised learning is to find a prediction function $f(\mathbf{x})$ that minimises the expected value of the loss function

$$R(f) = \int \ell(y, f(\mathbf{x}))\, \mathrm{d}P(\mathbf{x}, y), \tag{2.2}$$

also termed the *true risk* functional. The solution $f^* = \arg\min R(f)$ is the optimal predictor.

In general, the true risk functional cannot be computed directly because of the unknown distribution $P(\mathbf{x}, y)$. In practice, we approximate it by estimating the risk

over a finite collection of examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, drawn in an i.i.d. fashion. This estimated risk is known as *empirical risk* or *empirical error*, and is given by

$$R_{\text{emp}}(f) = \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, f(\mathbf{x}_i)). \tag{2.3}$$

Learning via empirical risk minimisation (ERM) consists in finding a hypothesis f over a hypothesis space $\mathcal{F}$ by solving

$$\hat{f}_m = \arg\min_{f \in \mathcal{F}} \; R_{\text{emp}}(f). \tag{2.4}$$

ERM forms the theoretical basis for many machine learning algorithms. In the case of regression, assuming $\mathcal{F}$ is the set of linear regressors and the squared loss is used, the ERM principle suggests that the best linear model $\mathbf{w}^*$ can be learned by minimising the following objective with respect to $\mathbf{w}$:

$$R_{\text{emp}}(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} (y_i - \mathbf{w}^{\mathrm{T}} \mathbf{x}_i)^2. \tag{2.5}$$

2.2.3 Convex optimisation theory

Many online learning problems can essentially be (re-)formulated as an online convex optimisation (OCO) task. In the following, we introduce some of the basics of OCO.

An online convex optimisation task typically consists of two major elements: a convex set $\mathcal{S}$ and a convex loss function $\ell_t(\cdot)$. At each time step t, the online algorithm picks a weight vector $\mathbf{w}_t \in \mathcal{S}$, after which it suffers a loss $\ell_t(\mathbf{w}_t)$. The goal of the online algorithm is to play a sequence of decisions $\mathbf{w}_1, \mathbf{w}_2, \ldots$ so as to minimise the regret in hindsight.

More formally, an online algorithm aims to achieve a low regret R_T after T rounds, where the regret R_T is defined as

$$R_T = \sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \inf_{\mathbf{w}^* \in \mathcal{S}} \sum_{t=1}^{T} \ell_t(\mathbf{w}^*). \tag{2.6}$$

In an online linear regression task on a sequence of examples $(\mathbf{x}_t, y_t)$, $t = 1, \ldots, T$, with $\mathbf{x}_t \in \mathbb{R}^n$ and $y_t \in \mathbb{R}$, a commonly used loss function is the squared loss $\ell_t(\mathbf{w}) = (y_t - \mathbf{w}^{\mathrm{T}} \mathbf{x}_t)^2$. In this case, the convex set $\mathcal{S}$ would be $\mathbb{R}^n$. There are a variety of algorithms to solve this problem.

Below, we briefly discuss two major families of OCO methods, namely first-order and second-order algorithms that will play an instrumental role in the development of our research work. For a comprehensive treatment of the subject, the interested reader is referred to the books by Shalev-Shwartz [2011]; Hazan [2016].

2.3 First-Order Online Learning

First-order methods aim to minimise the regret in Eq. (2.6) using first-order gradient information. The majority of first-order algorithms are classification algorithms. Only a few have been developed for regression tasks, such as online gradient descent [Zinkevich, 2003b] and online passive-aggressive regression [Crammer et al., 2006]. We describe these below.

2.3.1 Online passive-aggressive regression

Online passive-aggressive (PA) regression is a popular online regression algorithm that was introduced in [Crammer et al., 2006]. It has been applied across a number of diverse areas, including online portfolio selection [Li et al., 2012], non-negative matrix factorisation and completion [Blondel et al., 2014], and statistical machine translation [Turchi et al., 2014].

On every round, the PA regression algorithm receives an instance $\mathbf{x}_t$ and predicts a target value $\hat{y}_t = \mathbf{w}_t^{\mathrm{T}}\mathbf{x}_t$ using its internal regression function. After making a prediction, the algorithm is given the true target value y_t and suffers an instantaneous loss. PA regression relies on the ϵ-insensitive loss function (ILF) [Vapnik, 1998], defined as

$$\ell_\epsilon(\mathbf{w}; \mathbf{z}) \equiv \begin{cases} 0 & \text{if } |y - \mathbf{w}^{\mathrm{T}}\mathbf{x}| \leq \epsilon \\ |y - \mathbf{w}^{\mathrm{T}}\mathbf{x}| - \epsilon & \text{otherwise} \end{cases}, \tag{2.7}$$

where $\mathbf{z} = (\mathbf{x}, y)$ and ϵ is a positive hyperparameter that controls the algorithm's sensitivity to prediction mistakes. The ILF is zero when the predicted target deviates from the true target by less than ϵ units, and otherwise grows linearly with $|y_t - \hat{y}_t|$. At the end of every round, the algorithm uses $\mathbf{w}_t$ and the example $\mathbf{z}_t$ to generate a new weight vector $\mathbf{w}_{t+1}$, which is then used to extend the prediction on the next round.

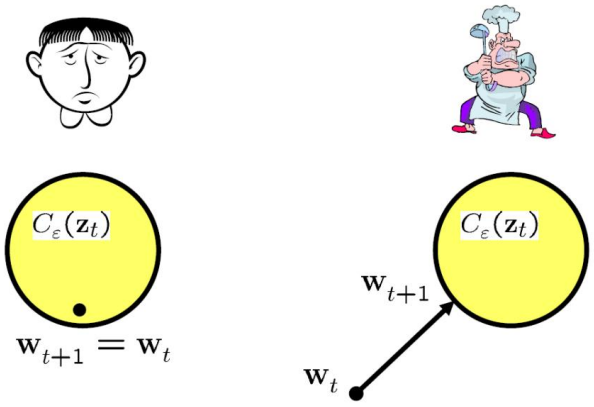


Figure 2.2: The mechanics of linear PA regression. The algorithm *passively* assigns $\mathbf{w}_{t+1} = \mathbf{w}_t$ as long as $\mathbf{w}_t$ incurs no loss on the incoming example $\mathbf{z}_t$. Otherwise, it *aggressively* projects $\mathbf{w}_t$ onto the feasible set $C_\epsilon(\mathbf{z}_t)$ of weight vectors which attain zero loss. Source: `https://home.ttic.edu/~shai/ppt/PassiveAggressive.ppt`.

We now describe how the PA regression algorithm sequentially updates its weight vector. The latter is initialised to the zero vector. Then, on each subsequent round, the algorithm sets the new weight vector to be

$$\mathbf{w}_{t+1} = \underset{\mathbf{w}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \qquad \text{s.t.} \qquad \ell_\epsilon(\mathbf{w}; \mathbf{z}_t) = 0. \tag{2.8}$$

The set $C_\epsilon(\mathbf{z}_t) \equiv \{\mathbf{w} \in \mathbb{R}^n : \ell_\epsilon(\mathbf{w}; \mathbf{z}_t) = 0\}$ is a hyper-slab of width 2ϵ. Geometrically, the PA algorithm for regression projects $\mathbf{w}_t$ onto this hyper-slab at the end of every round, unless $\mathbf{w}_t$ suffers no loss from the new data, i.e. $\mathbf{w}_t \in C_\epsilon(\mathbf{z}_t)$. These mechanics are illustrated in Figure 2.2.

Using the shorthand $\ell_t = \ell_\epsilon(\mathbf{w}; \mathbf{z}_t)$, Crammer et al. [2006] showed that the update in Eq. (2.8) has a closed-form solution, namely

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t \mathbf{v}_t, \tag{2.9}$$

where $\tau_t = \ell_t / \|\mathbf{x}_t\|_2^2$ is the Lagrange multiplier associated with the feasibility constraint $\mathbf{w}_{t+1} \in C_\epsilon(\mathbf{z}_t)$, and $\mathbf{v}_t \equiv \text{sgn}(y_t - \hat{y}_t)\mathbf{x}_t$ represents the direction of the update.

Soft-margin PA regression

As discussed above, the PA algorithm employs an aggressive update strategy, by modifying the weight vector by as much as needed to satisfy the constraint imposed by the current example. In certain real-life situations, this strategy may also result in undesirable consequences. Consider for instance the common phenomenon of outliers. An outlier may cause the PA algorithm to drastically change its weight vector in the wrong direction. A single outlier may thereby lead to several prediction mistakes on subsequent rounds.

To cope with such problems, Crammer et al. [2006] developed two variations on the PA update that employ gentler update strategies. These variations are based on the technique previously used to derive soft-margin classifiers [Vapnik, 1998]: they introduce a non-negative slack variable ξ into the optimisation problem defined in Eq. (2.8). This variable can be introduced in two different ways. First, we consider the update where the objective function scales linearly with ξ:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w},\,\xi} \left\{ \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 + C\xi \right\} \qquad \text{s.t.} \qquad \ell_\epsilon(\mathbf{w}; \mathbf{z}_t) \leq \xi \quad \text{and} \quad \xi \geq 0. \tag{2.10}$$

Here, C is a positive parameter which controls the influence of the slack term on the objective function. Specifically, larger values of C imply a more aggressive update, and C is therefore referred to as the *aggressiveness parameter* of the algorithm. This variant of PA regression is known as *PA-I*.

Alternatively, the objective function scales quadratically with ξ, resulting in the following constrained optimisation problem:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w},\,\xi} \left\{ \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 + C\xi^2 \right\} \qquad \text{s.t.} \qquad \ell_\epsilon(\mathbf{w}; \mathbf{z}_t) \leq \xi. \tag{2.11}$$

Note that the constraint $\xi \geq 0$ which appears in Eq. (2.10) is no longer necessary, since ξ^2 is always non-negative. The algorithm that results from this update is termed *PA-II*. As with PA-I, C is a positive parameter which governs the extent to which the PA-II update is aggressive.

As shown in [Crammer et al., 2006], the updates of PA-I and PA-II share the simple analytical solution $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t \mathbf{v}_t$, as defined in Eq. (2.9), but with the following Lagrange multipliers, respectively:

$$\tau_t = \min\left\{C,\ \frac{\ell_t}{\|\mathbf{x}_t\|_2^2}\right\} \quad \text{(PA-I)} \qquad \text{and} \qquad \tau_t = \frac{\ell_t}{\|\mathbf{x}_t\|_2^2 + \frac{1}{2C}} \quad \text{(PA-II)}. \tag{2.12}$$

All PA variants are outlined in Algorithm 2.

Algorithm 2 Passive-Aggressive Algorithms

1: **Initialisation:** $\mathbf{w}_1 = \mathbf{0}_{n\times 1}$, insensitivity parameter $\epsilon \geq 0$, aggressiveness parameter $C > 0$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: receive $\mathbf{x}_t \in \mathbb{R}^n$, predict $\hat{y}_t$ using $\mathbf{w}_t$
4: suffer loss $\ell_t(\mathbf{w}_t)$
5: set

$$\tau_t = \begin{cases} \ell_t / \|\mathbf{x}_t\|_2^2 & \text{(PA)} \\ \min\{C,\ \ell_t / \|\mathbf{x}_t\|_2^2\} & \text{(PA-I)} \\ \frac{\ell_t}{\|\mathbf{x}_t\|_2^2 + \frac{1}{2C}} & \text{(PA-II)} \end{cases}$$

6: update $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t \mathbf{v}_t$, where $\mathbf{v}_t = \text{sgn}(y_t - \hat{y}_t)\mathbf{x}_t$
7: **end for**

2.3.2 Online gradient descent

As we mentioned earlier, many online learning problems can be formulated as an online convex optimisation (OCO) task, which can be solved by applying the online gradient descent (OGD) algorithm. The latter is perhaps the simplest algorithm that applies to the most general OCO setting. Based on standard gradient descent from offline optimisation, it was introduced in its online form by Martin Zinkevich [Zinkevich, 2003b]. The pseudo-code for the algorithm is given in Algorithm 3.

Algorithm 3 Online Gradient Descent

1: **Initialisation:** convex set $\mathcal{K}$, horizon T, $\mathbf{x}_1 \in \mathcal{K}$, learning-rate schedule $\{\eta_t\}$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: play $\mathbf{x}_t$ and observe cost $f_t(\mathbf{x}_t)$
4: update and project:

$$\mathbf{y}_{t+1} = \mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)$$
$$\mathbf{x}_{t+1} = \Pi_{\mathcal{K}}(\mathbf{y}_{t+1}) \equiv \arg\min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x} - \mathbf{y}_{t+1}\|_2$$

5: **end for**

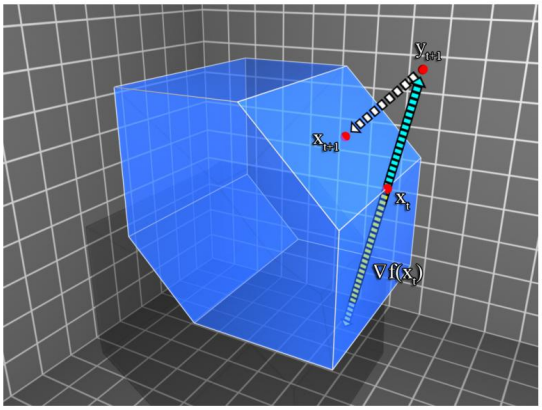


Figure 2.3: Online gradient descent: the iterate $\mathbf{x}_{t+1}$ is derived by moving $\mathbf{x}_t$ into the direction of the current gradient $\nabla f_t(\mathbf{x}_t)$, then projecting it back onto $\mathcal{K}$. Source: [Hazan, 2016].

At each iteration, the algorithm takes a step from the previous point in the direction of the gradient of the previous cost. This step may result in a point outside of the underlying convex set. In such cases, the algorithm projects the point back onto the convex set, i.e. finds its closest point in the convex set. Despite the fact that the next cost function may be completely different from the costs observed thus far, the regret attained by the algorithm is sublinear (see [Hazan, 2016, Theorem 3.1.]). A conceptual illustration of the algorithm is given in Figure 2.3.

In the case of linear regression with a squared loss function, the update rule for OGD takes the form

$$\mathbf{y}_{t+1} = \mathbf{x}_t - 2\eta_t(\hat{y}_t - y_t)\mathbf{x}_t. \tag{2.13}$$

In general, this rule is straightforward to implement, and takes linear time. OGD and PA share similar update rules, but differ in that OGD often employs some predefined learning-rate scheme, whereas PA chooses the optimal learning rate τ_t at each round (albeit subject to a predefined cost parameter C).

2.4 Second-Order Online Learning

To improve the efficacy of first-order learning methods, recent years have witnessed the active development of second-order online learning algorithms. In general,

these algorithms place a Gaussian prior distribution over the weights, with mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$. The model parameters are updated in the online learning process.

Examples of second-order algorithms for linear regression include adaptive regularisation of weight vectors [Crammer et al., 2009], which was inspired by confidence-weighted learning [Dredze et al., 2008], and a more recently introduced Bayesian treatment of online PA learning, namely online Bayesian passive-aggressive learning [Shi and Zhu, 2017].

2.4.1 Confidence-weighted learning

Confidence-weighted (CW) learning was introduced in [Dredze et al., 2008] as a probabilistic extension of online passive-aggressive learning that takes into account model/parameter uncertainty by maintaining a probabilistic measure of confidence in each weight. Less confident weights are updated more aggressively than more confident ones. Weight confidence is formalised with a Gaussian distribution over weight vectors, which is updated for each new training instance so that the probability of correct classification for that instance under the updated distribution meets a specified confidence.

In the CW framework, the linear classifier is modelled with a Gaussian distribution, i.e. $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Given an instance $\mathbf{x}$, the label is then predicted according to the sign of $\mathbf{w}^{\mathrm{T}}\mathbf{x}$. The multivariate Gaussian distribution over weight vectors induces a univariate Gaussian distribution over the margin viewed as a random variable. Specifically:

$$M \equiv y(\mathbf{w}^{\mathrm{T}}\mathbf{x}) \sim \mathcal{N}(y(\boldsymbol{\mu}^{\mathrm{T}}\mathbf{x}), \mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}). \tag{2.14}$$

Following the intuition underlying the PA algorithms, the CW algorithm chooses the distribution closest in the KL divergence sense to the current distribution $\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$. Thus, on round t, the algorithm sets the parameters of the distribution

by solving the following convex optimisation problem:

$$\begin{aligned} \mathcal{N}(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1}) = \min \quad & \mathrm{KL}\Big[\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \,\|\, \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)\Big] \\ \text{s.t.} \quad & \mathbb{P}\Big\{y_t(\mathbf{w}^{\mathrm{T}}\mathbf{x}_t) \geq 0\Big\} \geq \eta. \end{aligned} \tag{2.15}$$

In plain English, this means that the new distribution should remain as close as possible to the previous distribution so that the classifier does not forget the information learnt from previous instances. The constraint reflects the fact that the new classifier should classify the new instance $\mathbf{x}_t$ correctly with probability higher than a predefined threshold parameter $\eta \in (0, 1)$.

Note that this is only the basic form of confidence weighted algorithms and has several drawbacks: i) similar to the hard-margin PA algorithm, the constraint forces the new instance to be correctly classified, which makes this algorithm very sensitive to noise; ii) the constraint is expressed in terms of a probability. It is easy to solve a problem with a constraint of the form $g(\boldsymbol{\mu}, \boldsymbol{\Sigma}) < 0$. However, a problem with a constraint on a probability is only solvable when the distribution is known. Thus, this method is hardly generalisable to other online learning tasks for which the constraint does not follow a Gaussian distribution.

2.4.2 Adaptive regularisation of weights

Confidence-weighted algorithms have been shown to perform well in practice (see, e.g., [Dredze et al., 2008]), but they suffer from several problems. First, the update is quite aggressive, forcing the probability of predicting each example correctly to be at least $\eta > 1/2$ regardless of the cost to the objective. This may cause severe over-fitting when labels are noisy, especially considering the fact that the CW framework assumes that the data are linearly separable. Second, they are designed for classification, and it is not clear how to extend them to alternative settings such as regression. This is in part because the constraint is written in discrete terms where the prediction is either correct or no.

Adaptive regularisation of weights (AROW) is a variant of CW that was proposed by Crammer et al. [2009] to deal with both of the aforementioned issues, coping

more effectively with label noise and generalising the advantages of CW learning in an extensible way. Like CW, AROW maintains a Gaussian distribution over weight vectors, with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. However, it expresses the CW constraint in terms of a set of regularisers, minimising the following unconstrained objective on each round:

$$\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathrm{KL}\Big[\mathcal{N}(\boldsymbol{\mu},\, \boldsymbol{\Sigma}) \,\|\, \mathcal{N}(\boldsymbol{\mu}_t,\, \boldsymbol{\Sigma}_t)\Big] + \lambda_1 \ell_{\mathrm{h}^2}(y_t, \boldsymbol{\mu}^{\mathrm{T}}\mathbf{x}_t) + \lambda_2 \mathbf{x}_t^{\mathrm{T}} \boldsymbol{\Sigma} \mathbf{x}_t, \qquad (2.16)$$

where $\ell_{\mathrm{h}^2}(y_t, \boldsymbol{\mu}^{\mathrm{T}}\mathbf{x}_t) = (\max\{0,\, 1 - y_t(\boldsymbol{\mu}^{\mathrm{T}}\mathbf{x}_t)\})^2$ is the squared-hinge loss suffered using the weight vector $\boldsymbol{\mu}$ to predict the output for input $\mathbf{x}_t$ when the true output is y_t, and $\lambda_1, \lambda_2 \geq 0$ are two trade-off hyperparameters.

The objective balances three desires. First, the parameters should not change radically on each round, since the current parameters contain information about previous examples (first term). Second, the new mean parameters should predict the current example with low loss (second term). Finally, as we see more examples, our confidence in the parameters should generally grow (third term).

Besides the robustness to noisy data, another important advantage of AROW is its ability to be easily generalised to other online learning tasks. Of particular interest to us is online linear regression. Applying the ideas behind AROW to regression problems turns out to yield the well-known recursive least squares (RLS) algorithm [Haykin, 2013], for which AROW offers new bounds.

2.4.3 Online Bayesian passive-aggressive regression

Though enjoying strong discriminative ability suitable for predictive tasks, the online PA regression framework is formulated as a point-estimate problem optimising some deterministic objective function. This may lead to some inconvenience. On the one hand, PA makes point rather than probabilistic predictions, thereby failing to explicitly account for model/prediction uncertainty. On the other hand, the estimated single large-margin model is often less than sufficient to describe complex data, such as those with rich underlying structures. Online Bayesian

passive-aggressive regression (BayesPA) was introduced to effectively address these shortcomings [Shi and Zhu, 2017].

Instead of updating a point estimate of $\mathbf{w}$, BayesPA sequentially infers a new post-data posterior distribution $q_{t+1}(\mathbf{w})$, either parametric or nonparametric, on the arrival of new data $(\mathbf{x}_t, y_t)$ by solving the following optimisation problem[2]:

$$q_{t+1}(\mathbf{w}) = \underset{q(\mathbf{w})\in\mathcal{F}_t}{\arg\min} \Big\{ \mathrm{KL}[q(\mathbf{w}) \,\|\, q_t(\mathbf{w})] + 2C\, \ell_\epsilon(q(\mathbf{w}); \mathbf{x}_{t+1}, y_{t+1}) \Big\}, \tag{2.17}$$

where $\mathcal{P}$ is the probability simplex, $\ell_\epsilon(q(\mathbf{w}); \mathbf{x}_{t+1}, y_{t+1})$ is the expected ϵ-insensitive loss, and C is the parameter for balancing the loss of newly estimated distribution on new data and the similarity between the new distribution and the distribution estimated at time t (the constant 2 is just for convenience in the subsequent inference). In other words, the algorithm finds a post-data posterior distribution $q_{t+1}(\mathbf{w})$ in the feasible zone that is not only close to the current weight distribution $q_t(\mathbf{w})$ in terms of the KL divergence, but also has a high likelihood of explaining the data.

By defining $\ell_\epsilon(q(\mathbf{w}); \mathbf{x}_{t+1}, y_{t+1}) \equiv \mathbb{E}_{q(\mathbf{w})}[\max\{0, |y_{t+1} - \mathbf{w}^{\mathrm{T}}\mathbf{x}_{t+1}| - \epsilon\}]$, then expanding the KL divergence in Eq. (2.17) and reorganising terms, we obtain the following closed-form update rule for BayesPA:

$$q_{t+1}(\mathbf{w}) = \frac{q_t(\mathbf{w}) \exp\{-2C \max\{0, |y_{t+1} - \mathbf{w}^{\mathrm{T}}\mathbf{x}_{t+1}| - \epsilon\}\}}{Q(\mathbf{x}_{t+1}, y_{t+1})}, \tag{2.18}$$

where $Q(\mathbf{x}_{t+1}, y_{t+1})$ is a normalisation constant (see [Shi and Zhu, 2017, Lemma 4] for a proof).

[2]Here we consider the soft-margin version of BayesPA.

3

Online Portfolio Selection

Contents

Online portfolio selection is a fundamental problem in computational finance, which has been extensively studied across several research communities, including finance, statistics, artificial intelligence, machine learning, and data mining. This chapter – a summarised version of [Li and Hoi, 2014] – aims to provide an overview and structural understanding of the most well-known techniques used in the literature.

3.1 Introduction

Portfolio selection aims to optimise the allocation of wealth across a set of assets, and is considered a fundamental research problem in computational finance. There are two major schools of thought: *i*) mean-variance theory [Markowitz, 1952, 1959; Markowitz et al., 2000] which originated in the finance community, and *ii*) capital growth theory (CGT) [Kelly, 1956; Hakansson and Ziemba, 1995] which has its roots in information theory. Mean-variance theory, widely known in the asset management industry, focuses on single-period (batch) portfolio selection and trades off a portfolio's expected return (mean) vs its risk (variance), which typically determines the optimal portfolios subject to the investor's risk-return profile. CGT, on the other hand, focuses on multiple-period or sequential portfolio selection, seeking to maximise the portfolio's expected growth rate, or expected log return. Although both theories solve the task of portfolio selection, the latter is more aligned with the 'online' scenario, which naturally consists of multiple periods and is the focus of this thesis.

The goal of online portfolio selection is to sequentially select a portfolio of assets so as to achieve certain targets. Several algorithms have been proposed to solve this task and can be broadly grouped into three categories, namely *follow the winner*, *follow the loser* and *pattern matching*. In the spirit of CGT, follow-the-winner algorithms try to asymptotically achieve the same growth rate (expected log return) as that of an optimal strategy. In contrast, follow the loser strategies transfer wealth from overperforming to underperforming assets, which may seem counterintuitive, but often leads to significantly better performance in practice. The third category, the pattern-matching approach, tries to predict the next market distribution based on historical data, and explicitly optimises the portfolio based on the sampled distribution. Although these three categories are focused on a single strategy (class), there is another category that combines multiple strategies (classes), known as *meta-learning algorithms*. Table 3.1 outlines the main algorithms and corresponding references.

Table 3.1: Taxonomy of online portfolio selection techniques. Source: [Li and Hoi, 2014].

Category	Algorithms	Representative References
Benchmarks	Buy and Hold	
	Best Stock	
	Constant Rebalanced Portfolios	[Kelly, 1956; Cover, 1991]
Follow the winner	Universal Portfolios	[Cover, 1991; Cover and Ordentlich, 1996]
	Exponential Gradient	[Helmbold et al., 1998]
	Follow the Leader	[Gaivoronski and Stella, 2000]
	Follow the Regularized Leader	[Agarwal et al., 2006]
	Aggregating-Type Algorithms	[Vovk and Watkins, 1998]
Follow the loser	Anti-Correlation	[Borodin et al., 2004]
	Passive-Aggressive Mean Reversion	[Li et al., 2012]
	Confidence-Weighted Mean Reversion	[Li et al., 2011]
	Online Moving Average Reversion	[Li et al., 2015]
	Robust Median Reversion	[Huang et al., 2011]
Pattern matching	Nonparametric Histogram Log-Optimal Strategy	[Györfi et al., 2006]
	Nonparametric Kernel-Based Log-Optimal Strategy	
	Nonparametric Nearest Neighbor Log-Optimal Strategy	[Györfi et al., 2008]
	Correlation-Driven Nonparametric Learning Strategy	[Li et al., 2010]
	Nonparametric Kernel-Based Semi-Log-Optimal Strategy	[Györfi et al., 2007]
	Nonparametric Kernel-Based Markowitz-Type Strategy	[Ottucsák and Vajda, 2007]
	Nonparametric Kernel-Based GV-Type Strategy	[Györfi and Vajda, 2008]
Meta learning	Aggregating Algorithm	[Vovk, 1990; Vovk and Watkins, 1998]
	Fast Universalization Algorithm	[Akcoglu et al., 2002, 2004]
	Online Gradient Updates	
	Online Newton Updates	[Das and Banerjee, 2011]
	Follow the Leading History	[Hazan and Seshadhri, 2009]

This chapter provides a fairly detailed description of the algorithms mentioned in Table 3.1. The remainder of the chapter is structured as follows. Section 3.2 formally posits the problem of online portfolio selection and discusses several practical issues. Section 3.3 introduces the state-of-the-art algorithms, including benchmarks in Section 3.3.1, follow-the-winner approaches in Section 3.3.2, follow-the-loser strategies in Section 3.3.3, pattern-matching approaches in Section 3.3.4, and meta-learning algorithms in Section 3.3.5. Finally, Section 3.4 concludes this chapter.

3.2 Problem Setting

We now formally describe the online portfolio selection (OLPS) problem. Consider an investment task over a financial market with m assets and a T-period horizon. On the t-th period, the asset prices are represented by a *closing-price vector* $\mathbf{p}_t \in \mathbb{R}_+^m$. The price changes are represented by a *price-relative vector* $\mathbf{x}_t \in \mathbb{R}_+^m$ such that $x_{t,i} \equiv \frac{p_{t,i}}{p_{t-1,i}}$ for each asset $i \in [m]$. Thus, an investment in asset i on the t-th period increases by a factor of $x_{t,i}$. Let us denote by $\mathbf{x}_{t_1:t_2} \equiv \{\mathbf{x}_{t_1}, \mathbf{x}_{t_1+1}, \ldots, \mathbf{x}_{t_2}\}$ a sequence of price-relative vectors ranging from period t_1 to t_2. Therefore, $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ represents the sequence of price-relative vectors over the entire T-period horizon.

An investment on the t-th period is specified by a *portfolio vector* $\mathbf{b}_t = (b_{t,1}, \ldots, b_{t,m})$, where $b_{t,i}$ represents the proportion of wealth invested in asset i. Typically, we assume the portfolio is self-financed and that no margin/short sale is allowed, so that each entry of a portfolio is non-negative and all entries add up to one:

$$\mathbf{b}_t \in \Delta_m \equiv \Big\{ \mathbf{b}_t : \mathbf{b}_t \in \mathbb{R}_+^m, \sum_{i=1}^{m} b_{t,i} = 1 \Big\}. \tag{3.1}$$

The investment procedure is represented by a *portfolio strategy* such that $\mathbf{b}$ is initialised at the equally-weighted portfolio by convention, that is $\mathbf{b}_1 = \mathbf{1}/m$, followed by a sequence of mappings $\mathbf{b}_t : \mathbb{R}_+^{m(t-1)} \to \Delta_m$ for $t = 2, 3, \ldots$, where $\mathbf{b}_t = \mathbf{b}_t(\mathbf{x}_{1:t-1})$ is the t-th portfolio given the historical market sequence $\mathbf{x}_{1:t-1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}$. We denote by $\mathbf{b}_{1:T} = \{\mathbf{b}_1, \ldots, \mathbf{b}_T\}$ the strategy for T periods.

On the t-th period, a portfolio $\mathbf{b}_t$ produces a *portfolio period return* s_t, that is, the wealth increases by a factor of $s_t \equiv \mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t = \sum_{i=1}^{m} b_{t,i} x_{t,i}$. We shall assume all profits are reinvested, which implies that wealth grows in a multiplicative fashion. Thus, after T periods, a portfolio strategy $\mathbf{b}_{1:T}$ produces a *portfolio cumulative wealth* of S_T, which amounts to the initial wealth times a factor of $\prod_{t=1}^{T} s_t$, that is

$$S_T(\mathbf{b}_{1:T}) = S_0 \prod_{t=1}^{T} s_t = S_0 \prod_{t=1}^{T} \mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t, \tag{3.2}$$

where S_0 denotes the initial wealth and can be set to \$1 without any loss of generality. Since the model assumes multiperiod reinvestment, we define the *exponential growth rate* for a strategy $\mathbf{b}_{1:T}$ as

$$W_T(\mathbf{b}_{1:T}) = \frac{1}{T} \log S_T(\mathbf{b}_{1:T}) = \frac{1}{T} \sum_{t=1}^{T} \log \mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t. \tag{3.3}$$

Finally, we formally formulate the OLPS procedure, and outline its algorithmic framework in Algorithm 4. In this task, a portfolio manager is a decision maker whose goal is to produce a portfolio strategy $\mathbf{b}_{1:T}$ so as to maximise her cumulative wealth S_T. She computes the portfolios sequentially. On each period t, the manager has access to the sequence of all previous price-relative vectors $\mathbf{x}_{1:t-1}$. Then, she computes a new portfolio $\mathbf{b}_t$ before observing the next price-relative vector $\mathbf{x}_t$, based on a decision criterion that varies among different managers. The portfolio $\mathbf{b}_t$ is

scored based on the portfolio period return s_t. This procedure is repeated until the end of the T-period investment horizon, and the portfolio strategy is finally scored according to its cumulative wealth S_T.

It is important to note that we have made several general and common assumptions in the above description that are non-trivial in practice:

1. Transaction costs: there are no commission fees or taxes;
2. Market liquidity: one can buy and sell any desired amount, even fractional, at the last closing price of any given trading period;
3. Market impact: no portfolio strategy shall influence the market, or the prices of other assets.

Algorithm 4 Online Portfolio Selection Framework

Initialisation: $\mathbf{b}_1 = \left(\frac{\mathbf{1}}{m}, \ldots, \frac{\mathbf{1}}{m}\right)$, $S_0 = 1$
for $t = 1, 2, \ldots, T$ **do**
  the market reveals the price-relative vector $\mathbf{x}_t$
  the portfolio manager realises a profit (or loss) of $s_t = \mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t$, and her wealth changes to $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t)$
  if $t < T$ **then**
    the portfolio manager updates her portfolio from $\mathbf{b}_t$ to $\mathbf{b}_{t+1} \in \Delta_m$
  end if
end for

To better understand these notions and the model presented, let us illustrate with a classical example.

Example 3.2.1 (Synthetic market by [Cover and Gluss, 1986])**.** *Assume a two-asset market with cash and one volatile asset whose sequence of price relatives is given by* $\mathbf{x}_{1:T} = \{(1, 2), (1, \frac{1}{2}), (1, 2), \ldots\}$. *The first price relative vector* $\mathbf{x}_1 = (1, 2)$ *means that if one invests \$1 in the first asset, one gets \$1 at the end of the period; if one invests \$1 in the second asset, one collects \$2.*

Consider the equally-weighted portfolio $\mathbf{b}_{1:T} = \{(\frac{1}{2}, \frac{1}{2}), (\frac{1}{2}, \frac{1}{2}), \ldots\}$*, in which the manager redistributes the capital equally among the two assets at each trading period. In the first period, the portfolio wealth increases by a factor of* $1 \times \frac{1}{2} + 2 \times \frac{1}{2} = \frac{3}{2}$.

Starting with an initial capital of $S_0 = 1$*, the capital at the end of the first period would be* $S_1 = S_0 \times \frac{3}{2} = \frac{3}{2}$*. Similarly,* $S_2 = S_1 \times (1 \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2}) = \frac{3}{2} \times \frac{3}{4} = \frac{9}{8}$*. Thus, at the end of period* T*, the final cumulative wealth is equal to*

$$S_T(\mathbf{b}_{1:T}) = \begin{cases} \left(\frac{9}{8}\right)^{\frac{T}{2}} & (T \text{ even}) \\ \frac{3}{2} \times \left(\frac{9}{8}\right)^{\frac{T-1}{2}} & (T \text{ odd}) \end{cases} \tag{3.4}$$

and the exponential growth rate is

$$W_T(\mathbf{b}_{1:T}) = \begin{cases} \frac{1}{2}\log\frac{9}{8} & (T \text{ even}) \\ \frac{T-1}{2T}\log\frac{9}{8} + \frac{1}{T}\log\frac{3}{2} & (T \text{ odd}) \end{cases}, \tag{3.5}$$

which approaches $\frac{1}{2}\log\frac{9}{8} > 0$ *if* T *is sufficiently large.*

3.2.1 Transaction costs

In reality, the most important and unavoidable issue is that of transaction costs. In this section, we describe how to incorporate them into the OLPS framework for the purpose of evaluating competing OLPS algorithms only. We shall not cover strategies that directly solve the transaction cost issue. For this type of strategy, we refer the interested reader to [Davis and Norman, 1990; Iyengar and Cover, 2000; Akian et al., 2001; Schäfer, 2002; Györfi and Vajda, 2008; Ormos and Urbán, 2013].

The widely adopted transaction-cost model is that of *proportional transaction costs* [Blum and Kalai, 1999; Györfi and Vajda, 2008], which assumes that the incurred transaction cost is proportional to the wealth transferred during rebalancing. At the beginning of the tth period, the portfolio manager intends to rebalance the portfolio from the close-price adjusted portfolio $\widehat{\mathbf{b}}_{t-1}$ to a new portfolio $\mathbf{b}_t$. Here, each element of $\widehat{\mathbf{b}}_{t-1}$ is calculated as $\widehat{b}_{t-1,i} = \frac{b_{t-1,i}x_{t-1,i}}{\mathbf{b}_{t-1}^{\mathrm{T}}\mathbf{x}_{t-1}}$, for $i = 1, \ldots, m$. Assume two transaction cost rates $\gamma_b \in (0, 1)$ and $\gamma_s \in (0, 1)$ for buying and selling, respectively. After rebalancing, S_{t-1} will be decomposed into two parts, namely the net wealth N_{t-1} in the new portfolio $\mathbf{b}_t$ and the transaction costs incurred during buying and selling. If the wealth on asset i before rebalancing is higher than that after rebalancing, i.e. $\widehat{b}_{t-1,i}S_{t-1} \geq b_{t,i}N_{t-1}$, then a sale occurs at rebalancing.

Otherwise, a buy is required. This leads to the decomposition

$$S_{t-1} = N_{t-1} + \gamma_s \sum_{i=1}^{m} \max(0,\, \widehat{b}_{t-1,i} S_{t-1} - b_{t,i} N_{t-1}) + \gamma_b \sum_{i=1}^{m} \max(0,\, b_{t,i} N_{t-1} - \widehat{b}_{t-1,i} S_{t-1}). \tag{3.6}$$

Dividing this by S_{t-1}, we obtain

$$1 = c_{t-1} + \gamma_s \sum_{i=1}^{m} \max(0,\, \widehat{b}_{t-1,i} - b_{t,i} c_{t-1}) + \gamma_b \sum_{i=1}^{m} \max(0,\, b_{t,i} c_{t-1} - \widehat{b}_{t-1,i}), \tag{3.7}$$

where $c_{t-1} = \frac{N_{t-1}}{S_{t-1}} \in (0, 1)$ is the *transaction-cost factor* [Györfi and Vajda, 2008].

Finally, at each period t, the manager's wealth grows according to

$$S_t = S_{t-1} \times c_{t-1} \times (\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t), \tag{3.8}$$

and the final cumulative wealth after T periods equals

$$S_T = S_0 \prod_{t=1}^{T} [c_{t-1} \times (\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t)]. \tag{3.9}$$

3.3 Online Portfolio Selection Approaches

In this section, we provide a brief description of the techniques listed in Table 3.1. These techniques formulate the online portfolio selection problem as in Section 3.2 and derive explicit portfolio update schemes for each period. Basically, the routine is to implicitly assume various price relative predictions and learn optimal portfolios.

We begin by introducing several benchmark algorithms in Section 3.3.1. We then cover algorithms with explicit update schemes, which we classify based on the direction of the weight transfer they employ. The first approach, follow the winner, tries to increase the relative weights of more successful assets, often based on historical performance. In contrast, the follow-the-loser approach seeks to increase the relative weights of assets with lacklustre performance, transferring wealth from winners to losers. The third approach, pattern matching, tries to build a portfolio based on some sampled similar historical patterns without any explicit weight-transfer direction. After that, we survey meta-learning algorithms (MLAs), which can be applied to higher-level experts equipped with any existing algorithm.

3.3.1 Benchmarks

Buy-and-hold strategy

The most common baseline is the buy-and-hold (BAH) strategy, in which one invests wealth among a pool of assets with an initial portfolio $\mathbf{b}_1$ and holds the portfolio through the end of the investment horizon. The manager buys the assets of interest at the beginning of the first period and never rebalances the portfolio, meaning the value of any holding can only vary due to market fluctuations. For example, at the end of the 1st period, the portfolio weights become $\frac{\mathbf{b}_1 \odot \mathbf{x}_1}{\mathbf{b}_1^{\mathrm{T}} \mathbf{x}_1}$, where $\odot$ denotes element-wise multiplication. In summary, the final wealth achieved by a BAH strategy can be expressed as

$$S_T(\mathrm{BAH}(\mathbf{b}_1)) = \mathbf{b}_1^{\mathrm{T}} \left(\odot_{t=1}^{T} \mathbf{x}_t \right). \tag{3.10}$$

The BAH strategy initialised at the equally-weighted portfolio $\mathbf{b}_1 = (\frac{1}{m}, \ldots, \frac{1}{m})$ is referred to as the uniform BAH strategy and is often adopted to produce a market index.

Best-stock strategy

Another widely adopted benchmark is the best-stock (Best) strategy, a particular case of the BAH strategy that assigns all capital to the stock with the best performance in hindsight. Its initial portfolio $\mathbf{b}^\circ$ can be calculated in hindsight as follows:

$$\mathbf{b}^\circ = \underset{\mathbf{b} \in \Delta_m}{\arg\max}\ \mathbf{b}^{\mathrm{T}} \left(\odot_{t=1}^{T} \mathbf{x}_t \right). \tag{3.11}$$

As a result, the final wealth achieved by this strategy is given by

$$S_T(\mathrm{Best}) = \max_{\mathbf{b} \in \Delta_m}\ \mathbf{b}^{\mathrm{T}} \left(\odot_{t=1}^{T} \mathbf{x}_t \right) = S_T(\mathrm{BAH}(\mathbf{b}^\circ)). \tag{3.12}$$

Constant rebalanced portfolios

Another more sophisticated benchmark strategy is the constant rebalanced portfolio (CRP) strategy, which rebalances the portfolio to a fixed portfolio $\mathbf{b}$ every period. In

particular, the portfolio strategy can be represented as $\mathbf{b}_{1:T} = \{\mathbf{b}, \mathbf{b}, \ldots\}$. Thus, the cumulative portfolio wealth achieved by a CRP strategy after T periods is defined as

$$S_T(\mathrm{CRP}(\mathbf{b})) = \prod_{t=1}^{T} \mathbf{b}^{\mathrm{T}} \mathbf{x}_t. \tag{3.13}$$

One special CRP strategy that rebalances to the uniform portfolio $\mathbf{b}_1 = (\frac{1}{m}, \ldots, \frac{1}{m})$ each period is the uniform constant rebalanced portfolio (UCRP). It is possible to calculate an optimal offline portfolio for the CRP strategy as

$$\mathbf{b}^* = \operatorname*{arg\,max}_{\mathbf{b} \in \Delta_m} \log S_T(\mathrm{CRP}(\mathbf{b})) = \operatorname*{arg\,max}_{\mathbf{b} \in \Delta_m} \sum_{t=1}^{T} \log(\mathbf{b}^{\mathrm{T}} \mathbf{x}_t), \tag{3.14}$$

which is convex and can be efficiently solved. The CRP strategy with $\mathbf{b}^*$ is termed best constant rebalanced portfolio (BCRP). The terminal wealth and exponential growth rate achieved by the BCRP strategy are respectively:

$$\begin{aligned} S_T(\mathrm{BCRP}) &= \max_{\mathbf{b} \in \Delta_m} S_T(\mathrm{CRP}(\mathbf{b})) = S_T(\mathrm{CRP}(\mathbf{b}^*)), \\ W_T(\mathrm{BCRP}) &= \max_{\mathbf{b} \in \Delta_m} \frac{1}{T} \log S_T(\mathrm{CRP}(\mathbf{b})) = \frac{1}{T} \log S_T(\mathrm{CRP}(\mathbf{b}^*)). \end{aligned} \tag{3.15}$$

Note that BCRP is a hindsight strategy, and therefore can only be calculated with complete market sequences. Cover [1991] proved the benefits of BCRP as a target: BCRP outperforms the best stock, value line index (geometric mean of component returns) and the arithmetic mean of component returns (in other words, BAH). Moreover, BCRP is invariant under permutations of the price relative sequences, meaning it does not depend on the order of arrival of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$.

Let us now compare BAH against CRP by means of the following example.

Example 3.3.1 (Synthetic Market by [Cover and Gluss, 1986])**.** *Assume a two-asset market with cash and one volatile asset with the price relative sequence* $\mathbf{x}_{1:T} = \{(1, 2), (1, \frac{1}{2}), (1, 2), \ldots\}$. *Consider the uniform BAH portfolio* $\mathbf{b}_1 = (\frac{1}{2}, \frac{1}{2})$ *and the CRP strategy with a uniform portfolio* $\mathbf{b} = (\frac{1}{2}, \frac{1}{2})$ *as well. Clearly, since no asset grows in the long run, the final wealth of BAH equals the uniform weighted sum of two assets, which roughly equals 1 in the long run. On the other hand, according to the analysis in Example 3.2.1, the final cumulative wealth of CRP is roughly* $(\frac{9}{8})^{\frac{T}{2}}$,

which increases exponentially. Note that BAH only rebalances on the 1st period, whereas the CRP rebalances every period. While this synthetic market does not experience any growth, CRP can produce an exponentially increasing return. The underlying idea of CRP is to take advantage of the underlying volatility, or so-called volatility pumping [Luenberger, 1998, Chapter 15].

Since CRP enforces a fixed portfolio each period, its frequent transactions will incur high transaction costs. Helmbold et al. [1998] proposed a semi-constant rebalanced portfolio (Semi-CRP), which rebalances the portfolio on selected periods rather than every period. One desired theoretical result for online portfolio selection is universality [Cover, 1991]. An online portfolio selection algorithm Alg *is universal if its average (external) regret [Stoltz and Lugosi, 2005; Blum and Mansour, 2007] for T periods asymptotically approaches 0:*

$$\frac{1}{T}\text{regret}_T(\text{Alg}) = W_T(\text{BCRP}) - W_T(\text{Alg}) \to 0 \quad \textit{as } T \to \infty. \tag{3.16}$$

In other words, a universal portfolio selection algorithm asymptotically approaches the same exponential growth rate as a BCRP strategy for arbitrary sequences of price relatives.

3.3.2 Follow-the-winner approaches

Follow the winner is characterised by increasing the relative weights of more successful assets. Rather than targeting the market or best stock, algorithms in this category often aim to track the BCRP strategy, which can be shown to be the optimal strategy in an i.i.d. market [Cover and Thomas, 2006, Theorem 15.3.1].

Universal portfolios

The basic idea underlying universal-portfolio algorithms consists in assigning capital to a single class of base experts, letting them trade, and finally pooling their wealth. Strategies in this category are analogous to the BAH strategy. The difference is that the base BAH expert is the strategy investing in a single stock, and thus the number of experts is the same as that of stocks. In other words, the BAH

strategy buys individual stocks, lets them run and finally pools their individual wealth. On the other hand, the base expert in the follow-the-winner category can be any strategy class that invests in any set of stocks. Besides, algorithms in this category are similar to the MLAs further described in subsection 3.3.5, although these generally apply to experts of multiple classes.

Cover [1991] proposed the universal portfolio (UP) strategy, and Cover and Ordentlich [1996] further refined it to the μ-weighted UP, in which μ denotes a given measure over the space of admissible portfolios Δ_m. Intuitively, Cover's UP operates similar to a fund of funds (FOF), and its main idea is to BAH the parameterised CRP strategies over the whole simplex domain. In particular, it initially allocates a proportion of wealth $\mathrm{d}\mu(\mathbf{b})$ to each portfolio manager operating a CRP strategy with $\mathbf{b} \in \Delta_m$, and lets the CRP managers trade. Then, at the end, each manager will grow her wealth to $S_T(\mathbf{b})\mathrm{d}\mu(\mathbf{b})$. Finally, Cover's UP pools the individual experts' wealth over the continuum of portfolio strategies. Note that $S_T(\mathbf{b}) = \exp\{TW_T(\mathbf{b})\}$, which means that the portfolio grows at an exponential rate of $W_T(\mathbf{b})$.

Formally, its update scheme [Cover and Ordentlich, 1996, Definition 1] can be interpreted as a weighted average of the historical performance across all admissible CRPs, i.e.

$$\mathbf{b}_{t+1} = \frac{\int_{\Delta_m} \mathbf{b} S_t(\mathbf{b})\mathrm{d}\mu(\mathbf{b})}{\int_{\Delta_m} S_t(\mathbf{b})\mathrm{d}\mu(\mathbf{b})}. \tag{3.17}$$

Note that at the beginning of period $t+1$, one CRP manager's wealth (historical performance) is equal to $S_t(\mathbf{b})\mathrm{d}\mu(\mathbf{b})$. Incorporating the initial wealth of $S_0 = 1$, the final cumulative wealth is the weighted average of all CRP managers' wealth [Cover and Ordentlich, 1996, Eq. (24)]:

$$S_T(\mathrm{UP}) = \int_{\Delta_m} S_T(\mathbf{b})\mathrm{d}\mu(\mathbf{b}). \tag{3.18}$$

One special case arises when μ is a Lebesgue (uniform) measure. In this case, the portfolio update reduces to Cover's UP [Cover, 1991, Eq. (1.3)]. Another special case is the Dirichlet-weighted UP [Cover and Ordentlich, 1996], which is proved to be a more optimal allocation. Alternatively, if the loss function is the negative

logarithm of the portfolio return, Cover's UP is actually an exponentially weighted average forecaster [Cesa-Bianchi and Lugosi, 2006].

Cover [1991] showed that under suitable smoothness conditions, the average of exponentials grows at the same exponential rate as the maximum, in which case it becomes possible to asymptotically approach BCRP's exponential growth rate. The regret achieved by Cover's UP is $O(m \log T)$, while its time complexity is $O(T^m)$. Cover and Ordentlich [1996] proved that the Dirichlet UP has the same scale of regret bound but a better constant term [Cover and Ordentlich, 1996, Theorem 2].

Exponential gradient

Strategies of the exponential-gradient type generally focus on the following optimisation problem:

$$\mathbf{b}_{t+1} = \underset{\mathbf{b} \in \Delta_m}{\arg\max} \left\{ \eta \log(\mathbf{b}^{\mathrm{T}} \mathbf{x}_t) - R(\mathbf{b}, \mathbf{b}_t) \right\}. \tag{3.19}$$

where $R(\mathbf{b}, \mathbf{b}_t)$ denotes a regularisation term and $\eta > 0$ is the learning rate. One straightforward interpretation of this problem is that it tracks the asset with the best performance in the last period, while keeping the new portfolio close to the old one.

Helmbold et al. [1998] proposed the exponential gradient (EG) strategy, which is based on the algorithm proposed for the mixture estimation problem [Helmbold et al., 1997]. The EG strategy employs the relative entropy as the regularisation term in Eq. (3.19), i.e.

$$R(\mathbf{b}, \mathbf{b}_t) = \sum_{i=1}^{m} b_i \log \frac{b_i}{b_{t,i}}. \tag{3.20}$$

EG's formulation is thus convex in $\mathbf{b}$. Nonetheless, it is hard to solve due to the non-linearity of the log function. To circumvent this issue, the authors approximated the log with its first-order Taylor expansion at $\mathbf{b}_t$:

$$\log(\mathbf{b}^{\mathrm{T}} \mathbf{x}_t) \approx \log(\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t) + \frac{\mathbf{x}_t}{\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t} (\mathbf{b} - \mathbf{b}_t), \tag{3.21}$$

thanks to which the first term in Eq. (3.19) becomes linear, thereby simplifying the solution of the resulting approximate problem. Solving this problem leads to

the following update rule [Helmbold et al., 1998, Eq. (3.3)]:

$$b_{t+1,i} = Z^{-1} b_{t,i} \exp\left\{\eta \frac{x_{t,i}}{\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t}\right\}, \qquad i = 1, \ldots, m, \tag{3.22}$$

where Z denotes the normalisation term ensuring the portfolio weights sum up to 1.

The regret of the EG strategy can be bounded by $O(T \log m)$ with $O(m)$ running time per period. The regret is not as tight as that of Cover's UP; however, its linear running time substantially surpasses that of Cover's UP. Besides, Helmbold et al. [1998] also proposed a variant which has a regret bound of $O(m^{0.5}(\log m)^{0.25} T^{0.75})$. One key parameter of EG is the learning rate $\eta > 0$. In order to achieve the aforementioned regret bound, η has to be small. This introduces a delicate trade-off, however, because as $\eta \to 0$, its weights converge to the uniform portfolio, so EG reduces to UCRP.

Das and Banerjee [2011] built upon the EG principles to develop a MLA named online gradient updates (OGU), which we shall briefly cover in Section 3.3.5. OGU combines its underlying experts in such a way that the overall system can achieve a performance that is no worse than any convex combination of the experts. A variant of OGU, called online lazy updates (OLU), was proposed in [Das et al., 2013] to handle the case of non-zero transaction costs.

Follow the leader

Follow-the-leader (FTL) algorithms seek to track the BCRP strategy up to time t, that is

$$\mathbf{b}_{t+1} = \mathbf{b}_t^* = \operatorname*{arg\,max}_{\mathbf{b} \in \Delta_m} \sum_{\tau=1}^{t} \log(\mathbf{b}^{\mathrm{T}} \mathbf{x}_\tau). \tag{3.23}$$

Clearly, this category follows the BCRP leader, and the ultimate leader is the BCRP over all periods.

Ordentlich [1996, Chapter 4.4] briefly mentioned a strategy to obtain portfolios by combining the BCRP up to time t with the uniform portfolio:

$$\mathbf{b}_{t+1} = \frac{t}{t+1} \mathbf{b}_t^* + \frac{1}{t+1} \frac{\mathbf{1}}{m}. \tag{3.24}$$

He also derived its worst-case bound, which is slightly worse than that of Cover's UP.

Gaivoronski and Stella [2000] proposed the concepts of successive constant rebalanced portfolios (SCRP) and weighted successive constant rebalanced portfolios (WSCRP) for stationary markets. At each period, SCRP directly adopts the BCRP portfolio up to that period, that is $\mathbf{b}_{t+1} = \mathbf{b}_t^*$. The authors further derived the optimal portfolio $\mathbf{b}_t^*$ via stochastic optimisation, resulting in the detailed updates of SCRP [Gaivoronski and Stella, 2000, Algorithm 1]. On the other hand, WSCRP outputs a convex combination of the SCRP portfolio and the last portfolio:

$$\mathbf{b}_{t+1} = (1-\gamma)\mathbf{b}_t^* + \gamma \mathbf{b}_t, \tag{3.25}$$

where $\gamma \in [0, 1]$ represents the trade-off parameter. The regret bounds achieved by SCRP [Gaivoronski and Stella, 2000, Theorem 1] and WSCRP [Gaivoronski and Stella, 2000, Theorem 4] are both $O(K^2 \log T)$, where K is a uniform upper bound on the gradient of $\log(\mathbf{b}^{\mathrm{T}}\mathbf{x})$ with respect to $\mathbf{b}$. It is straightforward to see that given the same assumption of upper/lower bounds on price relatives as Cover's UP [Cover, 1991, Theorem 6.1], the regret bound is on the same scale of Cover's UP, although the constant term is slightly worse.

Rather than assuming a stationary market, some follow-the-leader algorithms are built upon the assumption of non-stationarity. For instance, Gaivoronski and Stella [2000] proposed variable rebalanced portfolios (VRP), which calculates the BCRP portfolio based on a sliding window. To be more specific, VRP updates its portfolio as follows:

$$\mathbf{b}_{t+1} = \operatorname*{arg\,max}_{\mathbf{b} \in \Delta_m} \sum_{\tau=t-W+1}^{t} \log(\mathbf{b}^{\mathrm{T}}\mathbf{x}_\tau), \tag{3.26}$$

where W denotes a specified window size. Following their algorithms for CRP, they further proposed successive variable rebalanced portfolios (SVRP) and Weighted Successive Variable Rebalanced Portfolios (WSVRP). No theoretical results were provided for these two algorithms.

Follow the regularised leader

Another category of approaches follows a similar idea as FTL, but adds a regularisation term, and is therefore known as the follow-the-regularised-leader (FTRL) approach. In general, FTRL methods can be formulated as follows:

$$\mathbf{b}_{t+1} = \arg\max_{\mathbf{b}\in\Delta_m} \sum_{\tau=1}^{t} \log(\mathbf{b}^{\mathrm{T}}\mathbf{x}_\tau) - \frac{\beta}{2} R(\mathbf{b}), \tag{3.27}$$

where β denotes the trade-off parameter and $R(\mathbf{b})$ is a regularisation term on $\mathbf{b}$. Note that here all historical information is captured in the first term, thus the regularisation term only affects the next portfolio, which distinguishes FTRL from EG. One typical regularisation is the L2-norm, i.e. $R(\mathbf{b}) = \|\mathbf{b}\|^2$.

Agarwal et al. [2006] proposed the online newton step (ONS) algorithm by solving the optimisation problem in Eq. (3.27) with L2-norm regularisation, using online convex optimisation techniques [Zinkevich, 2003b; Hazan, 2006; Hazan et al., 2007]. Similar to the Newton method for offline optimisation, the basic idea is to replace the log term by its second-order Taylor expansion at $\mathbf{b}_t$, then solve the approximating problem to obtain a closed-form update scheme. Doing so, the ONS update rule is shown to be [Agarwal et al., 2006, Lemma 2]

$$\mathbf{b}_1 = \left(\frac{1}{m}, \ldots, \frac{1}{m}\right), \quad \mathbf{b}_{t+1} = \Pi_{\Delta_m}^{\mathbf{A}_t}(\delta \mathbf{A}_t^{-1}\mathbf{p}_t), \tag{3.28}$$

with

$$\mathbf{A}_t = \sum_{\tau=1}^{t} \frac{\mathbf{x}_\tau \mathbf{x}_\tau^{\mathrm{T}}}{(\mathbf{b}_\tau^{\mathrm{T}}\mathbf{x}_\tau)^2} + \mathbf{I}_m, \quad \mathbf{p}_t = \left(1 + \frac{1}{\beta}\right) \sum_{\tau=1}^{t} \frac{\mathbf{x}_\tau}{\mathbf{b}_\tau^{\mathrm{T}}\mathbf{x}_\tau}, \tag{3.29}$$

where β is the trade-off parameter, δ is a scale term, and $\Pi_{\Delta_m}^{\mathbf{A}_t}(\cdot)$ is an exact projection onto the simplex domain.

ONS iteratively updates the first- and second-order information as well as the portfolio with a time cost of $O(m^3)$, which is independent of the number of historical instances T. The authors also proved that ONS's regret bound is of the order of $O(m^{1.5}\log(mT))$ [Agarwal et al., 2006, Theorem 1], which is worse than the corresponding bound of Cover's UP or the Dirichlet-weighted UP.

Aggregating-type algorithms

Although BCRP is the optimal strategy for an i.i.d. market, the i.i.d. assumption is controversial in practice, so the optimal portfolio may not be a CRP or fixed-fraction portfolio. Some algorithms have been designed to track a different set of experts. These algorithms share a similar idea to the MLAs in Section 3.3.5. However, their base experts belong to a special class, namely the class of experts that invest all their wealth in a single stock[3].

Vovk and Watkins [1998] applied the aggregating algorithm (AA) [Vovk, 1990] to the online portfolio selection task, of which Cover's UP is a special case. The general setting for AA is to define a countable or finite set of base experts and sequentially allocate resources among them in order to achieve a performance that is no worse than any fixed combination. Its portfolio-weight updates are carried out according to [Vovk and Watkins, 1998, Algorithm 1]

$$\mathbf{b}_{t+1} = \frac{\int_{\Delta_m} \mathbf{b} \prod_{i=1}^{t-1} (\mathbf{b}^{\mathrm{T}} \mathbf{x}_t)^{\eta} P_0(\mathrm{d}\mathbf{b})}{\int_{\Delta_m} \prod_{i=1}^{t-1} (\mathbf{b}^{\mathrm{T}} \mathbf{x}_t)^{\eta} P_0(\mathrm{d}\mathbf{b})}. \tag{3.30}$$

where $P_0(\mathrm{d}\mathbf{b})$ denotes the prior weights of the experts. Cover's UP corresponds to AA with a uniform prior distribution and $\eta = 1$.

Singer [1997] proposed switching portfolios (SP) to track a changing market, in which assets' behaviour may change frequently. Unlike the CRP class, SP decides a set of basic strategies—for example, the pure strategy that invests all wealth in one asset—and chooses a prior distribution over that set. Based on the actual return of each strategy and the prior distribution, SP is able to select a portfolio for each period. In this spirit, the author proposed two algorithms, both of which assume that the duration of using a basic strategy follows a geometric distribution with parameter γ , which can be fixed or variable. With fixed γ , the first version of SP has an explicit update formula [Singer, 1997, Eq. (6)], namely

$$\mathbf{b}_{t+1} = \left(1 - \gamma - \frac{\gamma}{m-1}\right) \mathbf{b}_t + \frac{\gamma}{m-1} \mathbf{1}. \tag{3.31}$$

[3]In general, MLAs often rely on more complex experts from multiple classes.

With variable γ , SP has no explicit update. The authors also adapted the algorithm for transaction costs. Theoretically, the authors further gave the lower bound of SP's logarithmic wealth with respect to any underlying switching regime in hindsight [Singer, 1997, Theorem 2]. Empirical evaluation on Cover's two-stock pairs shows that SP can outperform UP, EG and BCRP in most cases.

It is worthwhile noting that switching portfolios adopt the notion of regime switching [Hamilton, 1994, 2008], which is different from the assumption underlying UP selection methods and seems to be more plausible than an i.i.d. market. Regime switching is also applied to some state-of-the-art trading strategies [Hardy, 2001]. However, the limitation of this approach is its distributional assumption, because nor Geometric neither Gaussian distributions seem to fit the market well (see, e.g., [Cont, 2001]).

3.3.3 Follow-the-loser approaches

The underlying assumption for the optimality of the BCRP strategy is that the market is i.i.d., which usually fails to hold in real-world data and thus often results in inferior empirical performance, as documented in various previous works. Instead of tracking winners, the follow-the-loser approach is characterised by transferring wealth from winners to losers. The underlying assumption of this approach is *mean reversion* [Bondt and Thaler, 1985; Poterba and Summers, 1988; Lo and MacKinlay, 1990], which means that the well(poor)-performing assets during the current period will perform poorly (well) in subsequent periods.

To better understand the mean-reversion principle, let us further analyse the behaviour of CRP in Example 3.3.1 [Li et al., 2012].

Example 3.3.2 (Synthetic market by Cover and Gluss [1986])**.** *As illustrated in Example 3.3.1, the uniform CRP grows exponentially in the synthetic market by Cover and Gluss [1986]. Here, we analyse the behaviour of its portfolio updates, which exhibit mean reversion, as illustrated in Table 3.2.*

Table 3.2: Example illustrating the mean-reversion trading idea.

Period	Price relative (A, B)	CRP	CRP return	Portfolio holdings	Notes
1	(1, 2)	$\left(\frac{1}{2}, \frac{1}{2}\right)$	$\frac{3}{2}$	$\left(\frac{1}{3}, \frac{2}{3}\right)$	$B \rightarrow A$
2	$(1, \frac{1}{2})$	$\left(\frac{1}{2}, \frac{1}{2}\right)$	$\frac{3}{4}$	$\left(\frac{2}{3}, \frac{1}{3}\right)$	$A \rightarrow B$
3	(1, 2)	$\left(\frac{1}{2}, \frac{1}{2}\right)$	$\frac{3}{2}$	$\left(\frac{1}{3}, \frac{2}{3}\right)$	$B \rightarrow A$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Suppose that the initial CRP portfolio is $(\frac{1}{2}, \frac{1}{2})$ *and that at the end of the 1st period, the closing-price adjusted portfolio holding becomes* $(\frac{1}{3}, \frac{2}{3})$*, so the corresponding cumulative wealth increases by a factor of* $\frac{3}{2}$*. At the beginning of the 2nd period, the CRP manager rebalances the portfolio to the initial uniform portfolio by transferring wealth from the well-performing stock (B) to the poor-performing one (A), which actually follows the mean-reversion principle. Then, cumulative wealth changes by a factor of* $\frac{3}{4}$ *and the portfolio holding at the end of the 2nd period becomes* $(\frac{2}{3}, \frac{1}{3})$*. At the beginning of the 3rd period, wealth transfer under the mean-reversion idea continues.*

In summary, CRP implicitly assumes that if one stock performs poorly (well), it tends to perform well (poorly) in subsequent periods, and thus increases the weights of poorly-performing stocks at the expense of well-performing ones.

Anti-correlation

Borodin et al. [2004] proposed a follow-the-loser portfolio strategy named anti-correlation (Anticor). Unlike Cover's UP, Anticor assumes that the market obeys the mean-reversion principle. To exploit this property, Anticor statistically makes a bet on the consistency of positive lagged cross-correlation and negative autocorrelation.

To obtain a portfolio for period $t + 1$, the Anticor algorithm adopts logarithmic price relatives [Hull, 2008] over two specific market windows, namely $\mathbf{y}_1 = \log(\mathbf{x}_{t-2w+1}^{t-w})$ and $\mathbf{y}_2 = \log(\mathbf{x}_{t-w+1}^{t})$. It then calculates the cross-correlation

matrix between $\mathbf{y}_1$ and $\mathbf{y}_2$:

$$M_{\text{cov}}(i,j) = \frac{1}{w-1}(y_{1,i} - \bar{y}_1)(y_{2,j} - \bar{y}_2), \tag{3.32}$$

$$M_{\text{corr}}(i,j) = \begin{cases} \frac{M_{\text{cov}}(i,j)}{\sigma_1(i) \times \sigma_2(j)} & \text{if } \sigma_1(i), \sigma_2(j) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.33}$$

Then, relying on the cross-correlation matrix, Anticor transfers wealth according to the mean-reversion trading idea, or moves the proportions from the stocks that increased more in value to those that increased less, and the corresponding amounts are adjusted based on the cross-correlation matrix. In particular, if asset i rises more than asset j and their sequences in the window are positively correlated, Anticor claims a transfer from asset i to asset j, with an amount equalling the cross-correlation term $M_{\text{corr}}(i,j)$ minus the corresponding autocorrelation values $\min\{0, M_{\text{corr}}(i,i)\}$ and $\min\{0, M_{\text{corr}}(j,j)\}$. These transfer claims are finally normalised so as to keep the portfolio in the simplex domain.

Due to its mean-reversion nature, it is difficult to obtain a useful bound (such as the universal regret bound) on Anticor. Although heuristic and lacking theoretical guarantees, Anticor empirically exhibits a competitive performance. Nonetheless, its heuristic nature cannot fully exploit the mean-reversion property, and therefore using learning algorithms that systematically capitalise on this property is highly desired.

Passive-aggressive mean reversion

Li et al. [2012] developed the passive-aggressive mean reversion (PAMR) strategy, which is so called because it exploits the mean-reversion property via online passive-aggressive (PA) learning [Crammer et al., 2006].

The main idea underlying PAMR is to pick a loss function that appropriately captures the mean-reversion property. That is, if the expected return based on the last price relative exceeds a pre-defined threshold, the loss should grow linearly; otherwise, it should be zero. In particular, to the best of our knowledge, Li et al. [2012] were the first authors to adapt the ϵ-insensitive loss function typically

used in support vector regression [Vapnik, 1998] to the context of online portfolio selection, defining the latter as

$$\ell_\epsilon(\mathbf{b}; \mathbf{x}_t) \equiv \begin{cases} 0 & \text{if } \mathbf{b}^{\mathrm{T}}\mathbf{x}_t \leq \epsilon \\ \mathbf{b}^{\mathrm{T}}\mathbf{x}_t - \epsilon & \text{otherwise} \end{cases}, \tag{3.34}$$

where $\epsilon \in [0, 1]$ is a parameter representing the mean-reversion threshold. Based on this loss function, PAMR passively maintains the last portfolio if its loss is zero; otherwise, it aggressively transitions to a new portfolio that can bring the loss down to zero. Formally, PAMR obtains the new portfolio weights by solving the following optimisation problem:

$$\mathbf{b}_{t+1} = \underset{\mathbf{b} \in \Delta_m}{\arg\min} \ \frac{1}{2}\|\mathbf{b} - \mathbf{b}_t\|^2 \qquad \text{s.t.} \qquad \ell_\epsilon(\mathbf{b}; \mathbf{x}_t) = 0. \tag{3.35}$$

As demonstrated in [Li et al., 2012, Proposition 1], this optimisation problem admits an elegant closed-form solution given by

$$\mathbf{b}_{t+1} = \mathbf{b}_t - \tau_t(\mathbf{x}_t - \bar{x}_t\mathbf{1}), \qquad \tau_t = \max\left\{0, \ \frac{\mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t - \epsilon}{\|\mathbf{x}_t - \bar{x}_t\mathbf{1}\|^2}\right\}. \tag{3.36}$$

Since the authors ignored the non-negativity constraint on the portfolio weights in their derivation, they also added a simplex projection step [Duchi et al., 2008]. The above closed-form update clearly reflects the mean-reversion trading idea by gradually transferring wealth from the well-performing to the poorly-performing assets, where over(under)-performance is measured relative to the cross-sectional average of price relatives. It also coincides with the general form of return-based contrarian strategies [Lo and MacKinlay, 1990, Eq. (1)], except for the adaptive multiplier τ_t. Besides the optimisation problem in Eq. (3.35), the authors also proposed two variants to handle potential noise in price relatives, by introducing some non-negative slack variables into the optimisation, a widely used technique in soft-margin classification.

Similar to the Anticor algorithm, due to PAMR's mean-reversion nature, it is hard to derive a meaningful theoretical regret bound. Nevertheless, PAMR achieves significant performance, beating several competing algorithms, and tends to be robust across different parameter configurations. It also enjoys linear update time

and runs extremely fast in the backtests, which shows its practicability in large-scale real-world applications. The underlying idea is to exploit single-period mean reversion, which is empirically verified by its evaluation on several real market data sets. However, PAMR suffers from drawbacks in risk management, as it experiences significant performance degradation if its single-period mean reversion assumption is not verified. This is clearly indicated by its performance on the Dow Jones Industrial Average (DJIA) data set [Borodin et al., 2004; Li et al., 2012].

Confidence-weighted mean reversion

To further exploit second-order information, Li et al. [2011] proposed the confidence-weighted mean reversion (CWMR) algorithm. In this context, second-order information refers to the covariance of portfolio weights (not prices or price relatives), and mean reversion is exploited via confidence-weighted (CW) online learning [Dredze et al., 2008].

Basically, CWMR models the portfolio weight vector as a multivariate Gaussian distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^m$ and diagonal covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$, with non-zero diagonal elements σ^2 and zero off-diagonal elements. While the mean represents an estimate of the future portfolio, the diagonal covariance matrix measures the confidence we have in that estimate. Then, CWMR sequentially updates the mean and covariance matrix of this Gaussian distribution and samples portfolios at the beginning of each period. In particular, the authors define $\mathbf{b}_t \in \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ and update the distribution parameters according to a mechanism similar to PA learning. More precisely, CWMR keeps the next distribution close to the last distribution in terms of Kullback-Leibler (KL) divergence, under the condition that the probability of the achieved portfolio return being lower than ϵ exceeds a specified threshold. Formally, the CWMR optimisation problem is

$$(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1}) = \underset{\boldsymbol{\mu} \in \Delta_m, \boldsymbol{\Sigma}}{\arg\min} \ \mathrm{KL}\left[\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \,\|\, \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)\right] \quad \text{s.t.} \quad \mathbb{P}\Big\{\boldsymbol{\mu}^{\mathrm{T}} \mathbf{x}_t \leq \epsilon\Big\} \geq \theta. \tag{3.37}$$

To solve this problem, Li et al. [2013] transformed it using two techniques. One transformed optimisation problem [Li et al., 2013, Eq. (3)] is

$$(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1}) = \arg\min \quad \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_t|}{|\boldsymbol{\Sigma}|} + \mathrm{Tr}(\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\Sigma}) + (\boldsymbol{\mu}_t - \boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}_t^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu})\right]$$
$$\text{s.t.} \qquad \epsilon - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{x}_t \geq \phi\mathbf{x}_t^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{x}_t, \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{1} = 1, \quad \boldsymbol{\mu} \succeq \mathbf{0}. \tag{3.38}$$

Solving the transformed problem, one can obtain the following closed-form update scheme [Li et al., 2013, Proposition 4.1]:

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \lambda_{t+1}\boldsymbol{\Sigma}_t(\mathbf{x}_t - \widetilde{x}_t\mathbf{1}), \quad \boldsymbol{\Sigma}_{t+1}^{-1} = \boldsymbol{\Sigma}_t^{-1} + 2\lambda_{t+1}\phi\mathbf{x}_t\mathbf{x}_t^{\mathrm{T}}, \tag{3.39}$$

where λ_{t+1} corresponds to the Lagrangian multiplier calculated by Eq. (11) in [Li et al., 2013] and $\widetilde{x}_t = \frac{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Sigma}_t\mathbf{x}_t}{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Sigma}_t\mathbf{1}}$ denotes the confidence-weighted price relative mean. Clearly, the above update scheme captures the mean-reversion principle, and exploits both the first- and second-order information of a portfolio vector.

As is the case with Anticor and PAMR, CWMR's mean reversion nature makes it hard to obtain a meaningful theoretical regret bound for the algorithm. The experiments conducted in [Li et al., 2011, 2013] demonstrate that the CWMR strategy is able to outperform the state-of-the-art, including PAMR, which only exploits the first-order information of a portfolio vector. However, CWMR also relies on the assumption of single-period mean reversion, and so suffers from the same risk-management issue affecting PAMR.

Online moving average reversion

Observing that PAMR and CWMR implicitly assume *single-period* mean reversion, causing them to fail on the DJIA data set, Li et al. [2015] designed a *multi-period* mean reversion strategy named online moving average reversion, or OLMAR for short.

OLMAR is based on the observation that PAMR and CWMR implicitly predict future prices to be equal to previous prices, i.e. $\widehat{\mathbf{p}}_{t+1} = \mathbf{p}_{t-1}$, where $\mathbf{p}$ is the vector of asset prices. Such an extreme single-period prediction is most likely the cause of the poor performance of PAMR and CWMR on certain real-word data sets,

including the DJIA one. To circumvent this drawback, the authors of [Li et al., 2015] proposed a multi-period mean reversion algorithm, which explicitly predicts the next price vector using the moving average of past price vectors. More precisely, using the simple moving average $MA_t(w) = \frac{1}{w}\sum_{i=t-w+1}^{t}\mathbf{p}_i$, where w is the window length, they predict the next price relative to be

$$\widehat{\mathbf{x}}_{t+1}(w) = \frac{MA_t(w)}{\mathbf{p}_t} = \frac{1}{w}\left(1 + \frac{1}{\mathbf{x}_t} + \ldots + \frac{1}{\odot_{i=0}^{w-2}\mathbf{x}_{t-i}}\right). \tag{3.40}$$

Then, similarly to PAMR, they adopt online PA learning to learn portfolios. Formally,

$$\mathbf{b}_{t+1} = \underset{\mathbf{b}\in\Delta_m}{\arg\min}\ \frac{1}{2}\|\mathbf{b} - \mathbf{b}_t\|^2 \qquad \text{s.t.} \qquad \mathbf{b}^{\mathrm{T}}\widehat{\mathbf{x}}_{t+1} \geq \epsilon. \tag{3.41}$$

Unlike PAMR, this formulation seeks to achieve a 'sufficiently good' performance based on the predicted price relative and the performance threshold ϵ. Its solution being similar to PAMR's, we will omit it here. In practice, OLMAR achieves the best performance across most data sets, especially those on which PAMR and CWMR underperform.

Robust median reversion

Since the previously presented mean-reversion algorithms fail to consider noise and outliers in the data, they often suffer from estimation error, which leads to sub-optimal portfolios when such noise/outliers are indeed present. For this reason, Huang et al. [2011] suggested monetising mean reversion via a robust L_1-median estimator, based on which they devised a novel portfolio selection strategy called robust median reversion (RMR).

The basic principle behind RMR is to explicitly estimate the next price vector via a robust L_1-median at the end of the tth period. In other words, $\widehat{\mathbf{p}}_{t+1} = \boldsymbol{\mu}_{t+1}$, with

$$\boldsymbol{\mu}_{t+1} = \underset{\boldsymbol{\mu}}{\arg\min}\ \sum_{i=0}^{w-1}\|\mathbf{p}_{t-i} - \boldsymbol{\mu}\|. \tag{3.42}$$

In plain English, this is the point with minimal sum of Euclidean distances to all the given price vectors. The solution to this problem is unique provided the data

points are not collinear [Weiszfeld, 1937]. The corresponding price relative is

$$\widehat{\mathbf{x}}_{t+1}(w) = \frac{\boldsymbol{\mu}_{t+1}}{\mathbf{p}_t}. \tag{3.43}$$

Using this predicted price relative, RMR follows a portfolio optimisation method similar to OLMAR's to obtain a recursion on the portfolio weights. Empirically, RMR outperforms the state-of-the-art on most data sets.

3.3.4 Pattern-matching approaches

Besides the two categories of follow-the-winner/follow-the-loser strategies, another type of strategies may utilise both winners and losers. Such strategies fall under the umbrella of pattern matching, which spans non-parametric sequential investment strategies that guarantee universal consistency, meaning they achieve optimality of growth for any stationary or ergodic market process. Note that unlike BCRP's optimality for i.i.d. markets (which, by the way, motivates follow-the-winner approaches), pattern-matching approaches are suitable for non-i.i.d. markets and maximise the conditional expectation of *log*-returns given past observations. In non-i.i.d. markets, there is a big difference between the optimal growth rate and the growth rate of BCRP. For example, for New York Stock Exchange (NYSE) datasets during the 1962–2006 period, the average annual yield (AAY) of BCRP is about 20%, whereas pattern-matching strategies achieve AAYs well beyond 30% (see [Györfi et al., 2012, Chapter 2]).

Now let us describe the main idea of the pattern-matching approaches [Györfi et al., 2006] which consist of two steps: the sample-selection step and the portfolio-optimisation step[4]. In the first step, one selects an index set C of similar historical price relatives which is then used to predict the next price relative. After constructing this similarity set, each sample price relative $\mathbf{x}_i$, $i \in C$, is assigned a probability P_i. Existing methods often employ uniform probabilities, i.e. $P_i = \frac{1}{|C|}$, where $|\cdot|$

[4]Here we only introduce the key idea. All algorithms in this category consist of an additional aggregation step, which is a special case of MLAs in Section 3.3.5.

denotes the cardinality of a set. The second step, portfolio optimisation, learns an optimal portfolio based on the similarity set obtained in the first step, i.e.

$$\mathbf{b}_{t+1} = \arg\max_{\mathbf{b}\in\Delta_m} U(\mathbf{b}; C), \tag{3.44}$$

where $U(\mathbf{b}; C)$ is a specified utility function. A commonly used utility is the log utility which takes the form $U(\mathbf{b}; C) = \sum_{i\in C} \log(\mathbf{b}^{\mathrm{T}}\mathbf{x}_i)$. In case of an empty similarity set, a uniform portfolio is adopted as the optimal portfolio.

Sample-selection techniques

The general idea in this step is to generate samples of similar historical price relatives by comparing the preceding market windows of two price relatives. Suppose that we want to locate the price relatives that are similar to next price relative $\mathbf{x}_{t+1}$. The basic routine is to iterate over all historic price relative vectors $\mathbf{x}_i$, for $i = w+1, \ldots, t$, and count $\mathbf{x}_i$ as similar one if the preceding market window $\mathbf{x}_{i-w:i-1}$ is similar to the latest market window $\mathbf{x}_{t-w+1:t}$. The set C is maintained to contain the indices of similar price relatives. Note that the market window is a $(w \times m)$ matrix, and the similarity between two market windows is often calculated on the concatenated $(w \times m)$-dimensional vector.

Non-parametric histogram-based sample selection [Gyorfi and Schäfer, 2003] predefines a set of discretised partitions, partitions both the latest market window $\mathbf{x}_{t-w+1:t}$ and the historical market window $\mathbf{x}_{i-w:i-1}$, $i = w+1, \ldots, t$, and finally chooses price-relative vectors whose $\mathbf{x}_{i-w:i-1}$ is in the same partition as $\mathbf{x}_{t-w+1:t}$. In particular, given a partition $P_l = A_{j,l}$ of $\mathbb{R}_+^m$ into d_l disjoint sets and a corresponding discretisation function $G_l(\mathbf{x}) = j$ if $\mathbf{x} \in A_{l,j}$, we can define the similarity set as

$$C_H(\mathbf{x}_{1:t}, w) = \Big\{ w < i < t+1 \,:\, G_l(\mathbf{x}_{t-w+1:t}) = G_l(\mathbf{x}_{i-w:i-1}) \Big\}. \tag{3.45}$$

Note that l is adopted to aggregate multiple experts.

Non-parametric kernel-based sample selection [Györfi et al., 2006] identifies the similarity set by comparing two market windows by means of the Euclidean distance:

$$C_K(\mathbf{x}_{1:t}, w) = \Big\{ w < i < t+1 \,:\, \|\mathbf{x}_{t-w+1:t} - \mathbf{x}_{i-w:i-1}\| \leq \frac{c}{l} \Big\}, \tag{3.46}$$

where c and l are thresholds used to control the number of similar samples. Note that the authors adopted two threshold parameters for theoretical analysis.

Nonparametric nearest-neighbour sample selection [Györfi et al., 2008] looks for those price relatives whose preceding market windows are within the nearest neighbours of latest market windows in terms of the Euclidean distance:

$$C_N(\mathbf{x}_{1:t}, w) = \Big\{ w < i < t+1 \,:\, \mathbf{x}_{i-w:i-1} \text{ is among the } l \text{ NNs of } \mathbf{x}_{t-w+1:t} \Big\}, \quad (3.47)$$

where l is a threshold parameter.

Correlation-driven nonparametric sample selection [Li et al., 2010] identifies the linear similarity among two market windows via Pearson's correlation coefficient:

$$C_C(\mathbf{x}_{1:t}, w) = \Big\{ w < i < t+1 \,:\, \frac{\mathrm{cov}(\mathbf{x}_{i-w:i-1}, \mathbf{x}_{t-w+1:t})}{\mathrm{std}(\mathbf{x}_{i-w:i-1})\mathrm{std}(\mathbf{x}_{t-w+1:t})} \geq \rho \Big\}, \quad (3.48)$$

where ρ is a pre-defined correlation coefficient threshold.

Portfolio-optimisation techniques

The second step of pattern-matching approaches is to construct an optimal portfolio based on the similarity set C. The two main approaches are Kelly's CGT and Markowitz's mean-variance theory, as we shall describe in the following paragraphs.

Györfi et al. [2006] proposed to derive a log-optimal (Kelly) portfolio based on the similar price relatives found in the first step. Given a similarity set, the log-optimal utility function is defined as

$$U_L(\mathbf{b}; C(\mathbf{x}_{1:t})) = \mathbb{E}[\log(\mathbf{b}^{\mathrm{T}}\mathbf{x}) \,|\, \mathbf{x}_i, i \in C(\mathbf{x}_{1:t})] = \sum_{i \in C(\mathbf{x}_{1:t})} P_i \log(\mathbf{b}^{\mathrm{T}}\mathbf{x}_i), \quad (3.49)$$

where P_i denotes the probability assigned to a similar price relative $\mathbf{x}_i, i \in C(\mathbf{x}_{1:t})$. Györfi et al. [2006] assume a uniform probability across similar samples, leading to the following utility function:

$$U_L(\mathbf{b}; C(\mathbf{x}_{1:t})) = \sum_{i \in C(\mathbf{x}_{1:t})} \log(\mathbf{b}^{\mathrm{T}}\mathbf{x}_i). \quad (3.50)$$

Györfi et al. [2007] introduced the semi-log optimal strategy, which approximates the log in the log-optimal utility function to make it more tractable. The semi-log optimal utility function is defined as

$$U_S(\mathbf{b}; C(\mathbf{x}_{1:t})) = \mathbb{E}[f(\mathbf{b}^{\mathrm{T}}\mathbf{x}) \,|\, \mathbf{x}_i, i \in C(\mathbf{x}_{1:t})] = \sum_{i \in C(\mathbf{x}_{1:t})} P_i f(\mathbf{b}^{\mathrm{T}}\mathbf{x}_i), \tag{3.51}$$

where f is defined as the second-order Taylor expansion of $\log z$ at $z = 1$, i.e. $f(z) = z - 1 - \frac{1}{2}(z-1)^2$. The authors also assume a uniform probability among similar samples.

Ottucsák and Vajda [2007] proposed the nonparametric Markowitz-type strategy, which is a generalisation of the aforementioned semi-log optimal strategy. The basic idea behind this Markowitz-type strategy is to represent the portfolio returns using Markowitz's principle to trade off portfolio mean versus variance. To be more specific, the Markowitz-type utility function is defined as

$$U_M(\mathbf{b}; C(\mathbf{x}_{1:t})) = \mathbb{E}[\mathbf{b}^{\mathrm{T}}\mathbf{x} \,|\, \mathbf{x}_i, i \in C(\mathbf{x}_{1:t})] - \lambda \mathrm{Var}[\mathbf{b}^{\mathrm{T}}\mathbf{x} \,|\, \mathbf{x}_i, i \in C(\mathbf{x}_{1:t})], \tag{3.52}$$

where λ is a trade-off parameter. In particular, a simple numerical transformation shows that semi-log optimal portfolio is an instance of the log-optimal utility function with a specified λ.

In any of the aforementioned procedures, if the similarity set is non-empty, we can obtain an optimal portfolio based on similar price relatives and their probability distribution. In case of an empty set, we can either choose a uniform or a previous portfolio.

3.3.5 Meta-learning algorithms

Meta-learning algorithms (MLAs) [Das and Banerjee, 2011] represent yet another branch of online portfolio selection that is closely related to expert learning [Cesa-Bianchi and Lugosi, 2006]. This branch is directly applicable to a fund of funds (FOF) allocating resources to 'child' managers. In general, MLA works with several base experts, either from the same strategy class or different classes. Each expert outputs a portfolio vector for the coming period, and MLA combines these portfolios

to form a final portfolio, which is used at the next rebalancing. MLAs are similar to follow-the-winner algorithms; however, they are proposed to handle a broader class of experts, in which CRP can serve as a special case. On one hand, an MLA system can be used to smooth the final performance with respect to all the underlying experts, especially when some of these experts are sensitive to certain environments/parameters. On the other hand, combining a universal strategy and a heuristic algorithm for which it is not straightforward to obtain a theoretical bound (such as Anticor), can provide the universal property to the whole MLA system. Finally, MLA is able to combine all existing algorithms, thus providing a much broader area of application.

Aggregating algorithms

Besides the algorithms discussed in Section 3.3.2, the aggregating algorithm (AA) can also be generalised to include more sophisticated base experts. Given a learning rate $\eta > 0$, a measurable set of experts A with a prior measure P_0, a loss function $\ell(\mathbf{x}, \gamma)$, and an action $\gamma_t(\theta)$ chosen by expert θ at time t, AA updates the experts' weights as follows:

$$P_{t+1}(A) = \int_A \beta^{\ell(\mathbf{x}_t, \gamma_t(\theta))} P_t(\mathrm{d}\theta), \tag{3.53}$$

where $\beta = e^{-\eta}$ and P_t denotes the measure representing the allocation of wealth among experts at time t.

Fast universalisation

Akcoglu et al. [2002, 2004] proposed fast universalisation (FU), which extends Cover's UP [Cover, 1991] from a parameterised CRP class to a wide spectrum of investment strategies, including trading strategies operating on a single stock and portfolio strategies allocating wealth among the whole stock market. FU's basic idea is to evenly split the wealth among a set of base experts, let these experts operate on their own, and finally pool their wealth. FU's update is similar to that of Cover's UP, and it also asymptotically achieves a wealth equal to an

optimal fixed convex combination of base experts. In the case where all experts are CRPs, FU is reduced to Cover's UP.

Besides the universalisation over a continuous parameter space, various discrete buy-and-hold (BAH) combinations have been adopted by various existing algorithms. Rewritten in discrete form, the corresponding portfolio updates can be straightforwardly obtained. For example, Borodin et al. [2004] adopted the BAH strategy to combine Anticor experts with a finite number of window sizes. Similarly, Li et al. [2012] combined PAMR experts with a finite number of mean-reversion thresholds. Moreover, all pattern-matching approaches described in Section 3.3.4 rely on BAH to combine experts characterised by finite number of window sizes.

Online gradient and Newton updates

Das and Banerjee [2011] proposed two meta-optimisation algorithms, namely online gradient updates (OGU) and online Newton updates (ONU), which are natural extensions of EG and ONS, respectively. Since their updates and proofs are similar to their ancestors, we shall omit them here. Theoretically, OGU and ONU can achieve the same growth rate as the optimal convex combination of underlying experts. In particular, if any base expert is universal, the final meta-algorithm benefits from the universal property as well.

Follow the leading history

Hazan and Seshadhri [2009] developed a follow-the-leading-history (FLH) algorithm for changing environments. FLH can incorporate various universal base experts, such as the ONS algorithm. Its basic idea is to maintain a working set of finite experts, which are dynamically activated/deactivated based on their performance, akin to the Herbster-Warmuth algorithm [Herbster and Warmuth, 1998]. Unlike other MLAs where experts operate from the same starting point, FLH adopts experts starting at different periods. Theoretically, the FLH algorithm with universal methods is universal. When equipped with ONS, FLH can significantly outperform ONS.

3.4 Conclusion

This chapter conducted a brief survey of the online portfolio selection (OLPS) literature, an interdisciplinary field at the intersection of machine learning and finance. Focusing on algorithmic aspects, we began by formulating the OLPS task as a sequential decision problem and further grouped the existing algorithms into five major categories: benchmarks, follow the winner, follow the loser, pattern matching and MLAs. We note that although quite a few algorithms have been proposed in the literature, many open research problems remain unsolved and deserve further exploration. This was one of the inspirations for the methods developed in this thesis.

Part II

Contribution

4

Extending Passive-Aggressive Learning

Contents

4.1 Limitations of Passive-Aggressive Learning

In Chapter 2, we saw that online passive-aggressive (PA) algorithms exhibit a range of attractive properties including a unified treatment of three learning tasks (classification, regression and uniclass prediction), closed-form solutions and worst-

case finite-horizon loss bounds. Furthermore, for a suitable choice of Mercer kernel, PA algorithms can capture arbitrary non-linearities in the mapping from input variables to targets.

It might appear, therefore, that PA algorithms constitute a general-purpose framework for solving online learning problems. Unfortunately, they suffer from a number of significant and practical disadvantages, including but not limited to the following:

1. **Restriction to ϵ-insensitive loss** The PA framework limits itself to the ϵ-insensitive loss function (ILF). Despite its appealing properties (e.g. robustness to outliers), it would be more fruitful to consider a generalisation of PA learning that can support any type of loss function, in addition to the ILF.

2. **Failure to account for data distributions** To capture the proximity between consecutive weight vectors, PA algorithms rely on the Euclidean square distance. Whilst this distance is popular, this may not always be appropriate. A fundamental limitation of the Euclidean distance is that it does not take into account how the data are distributed. For example, if the length scales of the components of the weight vector $\mathbf{w}$ vary greatly, the largest length scale will dominate the squared distance, with potentially useful information in other components of $\mathbf{w}$ lost. The *Mahalanobis distance*

$$d^{(\mathrm{M})}(\mathbf{w}, \mathbf{w}_t; \boldsymbol{\Sigma}_t) \equiv \sqrt{(\mathbf{w} - \mathbf{w}_t)^{\mathrm{T}} \boldsymbol{\Sigma}_t^{-1} (\mathbf{w} - \mathbf{w}_t)}, \tag{4.1}$$

   where $\boldsymbol{\Sigma}_t$ is the covariance matrix of the weight vector distribution at round t, can overcome some of these problems since it effectively rescales the weight vector components.

3. **Predictions are not probabilistic** In regression, PA algorithms output a point estimate, and in classification, a 'hard' binary decision. Ideally, we desire to estimate the conditional distribution $p(y_t|\mathcal{D}_{1:t-1})$ of y_t given all the data $\mathcal{D}_{1:t-1}$ up to time $t-1$, so as to capture uncertainty in our prediction. In regression, this may take the form of 'error bars', but it is particularly

crucial in classification, where posterior probabilities of class membership are necessary to adapt to varying class priors and asymmetric misclassification costs.

4. **Sensitivity to hyperparameter settings** The performance of PA algorithms is measured by the cumulative squared loss over a given sequence of examples, i.e. by $\sum_{t=1}^{T} \ell_t^2(\mathbf{w}_t)$, where T is the length of the sequence. The weight vectors $\mathbf{w}_t$ in turn are functions of the insensitivity parameter ϵ, as well as of the aggressiveness parameter C in the case of the PA-I and PA-II variants. As a result, achieving good performance when applying PA algorithms in practice ultimately depends on selecting suitable values for these parameters. In offline learning, this generally entails a cross-validation procedure, which not only is wasteful of data and computation, but is also infeasible in an online setting.

In the following, we shall address the above shortcomings by grouping the last three and treating them separately from the first. The reason for this is because the last three can be tackled using a Bayesian approach. Although most of the discussion in this chapter will be devoted to the regression case, our results can be readily extended to classification and uniclass prediction.

4.2 Generalised Passive-Aggressive Learning

In this section, we discuss how to modify the original PA framework from [Crammer et al., 2006] so that it can support any type of loss function. We start by briefly reviewing Kivinen and Warmuth's gradient descent (GD) algorithm [Kivinen and Warmuth, 1997], from which PA algorithms originated. We then apply the PA logic to GD in order to arrive at our family of *generalised passive-aggressive* (GPA) methods.

4.2.1 Kivinen and Warmuth's gradient descent algorithm

In [Kivinen and Warmuth, 1997], the authors propose two algorithms for online prediction based on a linear model, namely the GD and EG (exponentiated gradient) algorithms.

The GD algorithm maintains a weight vector and updates it after each trial. The t-th weight vector $\mathbf{w}_t$ can be considered as the hypothesis the algorithm has before trial t about the best predictor for the trial sequence. At trial t, the algorithm receives an instance $\mathbf{x}_t$ and extends the prediction $\hat{y}_t = f(\mathbf{w}_t \cdot \mathbf{x}_t)$, where $f(\cdot)$ is a nonlinear activation function in the case of classification — e.g. $f(x) = \text{sgn}(x)$ — and is the identity in the case of regression[5]. After receiving the actual outcome y_t, the algorithm is charged for the possible discrepancy between the predicted outcome $\hat{y}_t$ and y_t. This discrepancy is measured using a loss function $\ell_t(\mathbf{w}) = \ell(\mathbf{w}; (\mathbf{x}_t, y_t))$, for example by the square loss function $(y_t - \mathbf{w} \cdot \mathbf{x}_t)^2$. We shall abbreviate the loss suffered on round t by ℓ_t, that is, $\ell_t = \ell_t(\mathbf{w}_t)$. After incurring the loss ℓ_t, the GD algorithm updates the weight vector according to

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla \ell_t(\mathbf{w}_t), \tag{4.2}$$

where η is a positive learning rate and $\nabla \ell_t(\mathbf{w}_t)$ is a shorthand for the gradient of $\ell_t(\cdot)$ evaluated at $\mathbf{w}_t$.

The GD algorithm can thus be considered as a simple application of the gradient-descent heuristic to the online prediction problem. As pointed out by Kivinen and Warmuth [1997], choosing the learning rate η is non-trivial and can significantly affect the performance of the algorithm.

4.2.2 Problem setting

To overcome the difficulty in tuning the learning rate η, the authors of [Crammer et al., 2006] start from the observation that the update rule in Eq. (4.2) is obtained

[5] As a reminder, $\mathbf{x} \cdot \mathbf{y} \equiv \mathbf{x}^{\mathrm{T}}\mathbf{y} = \sum_{i=1}^{n} x_i y_i$ signifies the dot product between two n-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$.

by minimising a first-order approximation of the loss function $\ell_t(\cdot)$ added to a proximal term $\|\mathbf{w} - \mathbf{w}_t\|_2^2/2$, i.e.

$$\min_{\mathbf{w}\in\mathbb{R}^n} \left\{\frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 + \eta[\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t)]\right\}, \tag{4.3}$$

where we have defined $\mathbf{g}_t \equiv \nabla\ell_t(\mathbf{w}_t)$, and η captures the trade-off between correctiveness (lowering the current loss) and conservativeness (retaining information learned on previous rounds).

Guided by the formalism underlying support vector machines, Crammer et al. [2006] modify Eq. (4.3) by introducing an upper bound $\epsilon \geq 0$ on the loss, leading to the following *constrained* minimisation problem:

$$\min_{\mathbf{w}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \qquad \text{s.t.} \qquad \ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) \leq \epsilon. \tag{4.4}$$

It is clear that this formulation replaces the choice of learning rate η with a choice of insensitivity parameter ϵ, which measures the sensitivity to prediction mistakes. The underlying rationale is that it should be more natural and intuitive for practitioners to specify a value for ϵ such that they do not care about the algorithm's mistakes as long as they are less than ϵ, but will not tolerate any deviation exceeding this threshold. For instance, one would be more likely to know the amount ϵ of money they can stand to lose when dealing with exchange rates rather than what which value would be appropriate for the more abstract learning rate parameter.

While the work of Crammer et al. [2006] is geared towards the absolute loss function $|y - \mathbf{w}\cdot\mathbf{x}|$, we shall solve the problem in Eq. (4.4) for general loss functions, which will form the basis for our GPA algorithms.

4.2.3 Solving the optimisation problem

In this section, we describe how to solve the optimisation problem at the core of GPA, namely Eq. (4.4). In doing so, we assume that $\mathbf{g}_t \neq \mathbf{0}_{n\times 1}$, the n-dimensional zero vector, because otherwise the current weight vector $\mathbf{w}_t$ would trivially solve the problem. The Lagrangian corresponding to the optimisation problem in Eq. (4.4) is

$$\mathcal{L}(\mathbf{w}, \eta) = \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 + \eta[\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \epsilon], \tag{4.5}$$

where $\eta \geq 0$ is a Lagrange multiplier.

The optimisation problem in Eq. (4.4) has a convex objective function and a single feasible affine constraint. These are sufficient conditions for Slater's condition to hold. Therefore, finding the problem's optimum is equivalent to satisfying the Karush-Kuhn-Tucker (KKT) conditions [Boyd and Vandenberghe, 2004]. Setting the partial derivatives of $\mathcal{L}$ with respect to the elements of $\mathbf{w}$ to zero gives

$$\mathbf{0}_{n\times 1} = \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \eta) = \mathbf{w} - \mathbf{w}_t + \eta \mathbf{g}_t \quad \Longrightarrow \quad \mathbf{w} = \mathbf{w}_t - \eta \mathbf{g}_t. \tag{4.6}$$

Substituting the above back into Eq. (4.5), we get

$$\mathcal{L}(\eta) = \frac{1}{2}\eta^2 \|\mathbf{g}_t\|_2^2 + \eta(\ell_t - \eta\|\mathbf{g}_t\|_2^2 - \epsilon) = -\frac{1}{2}\eta^2 \|\mathbf{g}_t\|_2^2 + \eta(\ell_t - \epsilon). \tag{4.7}$$

Taking the derivative of $\mathcal{L}(\eta)$ with respect to η and setting it to zero yields

$$0 = \frac{\partial \mathcal{L}(\eta)}{\partial \eta} = -\eta\|\mathbf{g}_t\|_2^2 + \ell_t - \epsilon \quad \Longrightarrow \quad \eta = \frac{\ell_t - \epsilon}{\|\mathbf{g}_t\|_2^2}. \tag{4.8}$$

Combining the above results with the fact that η must be non-negative, we obtain the following update rule for the GPA algorithm:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t, \qquad \eta_t = \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2}. \tag{4.9}$$

The similarity between this and the GD update in Eq. (4.2) is striking and sheds some light on GPA's behaviour. Namely, GPA can be seen as automatically adjusting the learning rate η of GD in a data-dependent fashion, i.e. based on $\mathbf{w}_t$, $\mathbf{x}_t$, y_t and ϵ.

4.2.4 Regularised updates

The GPA algorithm employs an aggressive update rule by modifying the weight vector by as much as needed to satisfy the constraint imposed by the current example. In certain real-life situations, this strategy might result in undesirable consequences. For instance, a mislabelled example may cause the GPA algorithm to drastically change its weight vector in the wrong direction. A single mislabelled example can lead to several prediction mistakes on subsequent rounds. To cope with such problems, we present two variations of the GPA method that employ gentler update strategies.

Soft formulations of PA learning (the so-called PA-I and PA-II variants) include slack variables in the objective function. We adopt this technique and introduce a non-negative slack variable ξ into the optimisation problem defined in Eq. (4.4). This variable can be introduced in two different ways. First, we consider the update where the objective function scales linearly with ξ, namely

$$\mathbf{w}_{t+1}, \xi^* = \underset{\mathbf{w}\in\mathbb{R}^n,\, \xi\geq 0}{\arg\min} \left\{ \frac{1}{2}\|\mathbf{w}-\mathbf{w}_t\|_2^2 + C\xi \right\} \qquad \text{s.t.} \qquad \ell_t + \mathbf{g}_t\cdot(\mathbf{w}-\mathbf{w}_t) \leq \epsilon + \xi. \quad (4.10)$$

Here $C > 0$ is a parameter which controls the influence of the slack term on the objective function. Specifically, we will show that larger values of C imply a more aggressive update step, and we therefore refer to C as the *aggressiveness parameter* of the algorithm, as in [Crammer et al., 2006]. Following the naming convention of Crammer et al. [2006], we term the algorithm that results from this update *GPA-I*.

Alternatively, we can have a *GPA-II* variant by having the objective function in Eq. (4.4) scale quadratically with ξ, resulting in the following constrained optimisation problem:

$$\mathbf{w}_{t+1}, \xi^* = \underset{\mathbf{w}\in\mathbb{R}^n,\, \xi}{\arg\min} \left\{ \frac{1}{2}\|\mathbf{w}-\mathbf{w}_t\|_2^2 + C\xi^2 \right\} \qquad \text{s.t.} \qquad \ell_t + \mathbf{g}_t\cdot(\mathbf{w}-\mathbf{w}_t) \leq \epsilon + \xi. \quad (4.11)$$

Note that the constraint $\xi \geq 0$ which appears in Eq. (4.10) is no longer necessary since ξ^2 is always non-negative.

Like GPA, the updates of GPA-I and GPA-II share the simple closed-form solution $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t$, where

$$\eta_t = \min\left\{ C,\ \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2} \right\} \qquad \text{(GPA-I)}$$

$$\eta_t = \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2 + \frac{1}{2C}}. \qquad \text{(GPA-II)}$$

A detailed derivation of the GPA-I and GPA-II updates is provided in the next paragraph. It is worth noting that these updates differ in the way they prevent overfitting: the former clips the step size at C, whereas the latter shrinks it towards zero (in the limit, as C tends to infinity, the coefficient η_t in GPA-II becomes infinitesimally small). The three GPA variants are summarised in Algorithm 5.

Algorithm 5 GPA: Generalised Passive-Aggressive Learning

Input: loss function $\ell(\mathbf{w}; (\mathbf{x}, y))$, insensitivity parameter $\epsilon \geq 0$, aggressiveness parameter $C > 0$
Initialise: $\mathbf{w}_1 = \mathbf{0}_{n \times 1}$
for $t = 1, 2, \ldots$ **do**
  receive instance: $\mathbf{x}_t \in \mathbb{R}^n$
  predict:

$$\hat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t \qquad \text{(regression)}$$
$$\hat{y}_t = \operatorname{sgn}(\mathbf{w}_t \cdot \mathbf{x}_t) \qquad \text{(classification)}$$

  receive actual target:

$$y_t \in \mathbb{R} \qquad \text{(regression)}$$
$$y_t \in \{-1, +1\} \qquad \text{(classification)}$$

  suffer loss: $\ell_t = \ell(\mathbf{w}_t; (\mathbf{x}_t, y_t))$
  compute loss gradient at $\mathbf{w} = \mathbf{w}_t$: $\mathbf{g}_t = \nabla \ell_t$
  update:
    1. set:

$$\eta_t = \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2} \qquad \text{(GPA)}$$
$$\eta_t = \min\left\{C,\, \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2}\right\} \qquad \text{(GPA-I)}$$
$$\eta_t = \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2 + \frac{1}{2C}}. \qquad \text{(GPA-II)}$$

    2. update: $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t$

end for

It is straightforward to verify that by setting $\ell_t = \max\{0,\, 1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)\}$ and $\epsilon = 0$ in Algorithm 5, we recover the original PA classification algorithms from [Crammer et al., 2006]. Below we give a sketch of the proof for the vanilla GPA variant:

$$\mathbf{g}_t = -\theta[1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)] y_t \mathbf{x}_t \qquad \Longrightarrow \qquad \|\mathbf{g}_t\|_2^2 = \|\mathbf{x}_t\|_2^2, \tag{4.12}$$

where $\theta(\cdot)$ denotes the Heaviside step function. From this, it follows that

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t = \mathbf{w}_t + \frac{\ell_t}{\|\mathbf{x}_t\|_2^2} \theta[1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)] y_t \mathbf{x}_t = \mathbf{w}_t + \frac{\ell_t}{\|\mathbf{x}_t\|_2^2} y_t \mathbf{x}_t, \tag{4.13}$$

where we have used the fact that $\max\{0,\, x\} \times \theta(x) = \max\{0,\, x\}$.

By a similar reasoning, we can show that when $\ell_t = \max\{0,\, |y_t - \mathbf{w}_t \cdot \mathbf{x}_t| - \epsilon\}$, Algorithm 5 boils down to the original PA regression algorithms. Hence, as one would expect, the GPA framework contains the original PA framework as a special case.

Derivation of the GPA-I and GPA-II updates

As in Section 4.2.3, whenever $\ell_t \leq \epsilon$, no update occurs and the learning rate η_t equals zero. If $\ell_t > \epsilon$, we derive these updates by first writing the Lagrangian of the respective optimisation problems and then solving the KKT conditions. The Lagrangian for the GPA-I optimisation problem is

$$\begin{aligned} \mathcal{L}(\mathbf{w}, \xi, \eta, \theta) &= \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \,+\, C\xi \,+\, \eta[\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \epsilon - \xi] \,-\, \theta\xi \\ &= \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \,+\, \xi(C - \eta - \theta) \,+\, \eta[\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \epsilon], \end{aligned} \tag{4.14}$$

where $\eta \geq 0$ and $\theta \geq 0$ are Lagrange multipliers. We now find the minimum of the Lagrangian with respect to the (unconstrained) primal variables $\mathbf{w}$ and ξ. As in the previously discussed GPA update, differentiating this Lagrangian with respect to the elements of $\mathbf{w}$ and setting these partial derivatives to zero gives

$$\mathbf{w} = \mathbf{w}_t - \eta \mathbf{g}_t. \tag{4.15}$$

Next, note that the minimum of the term $\xi(C - \eta - \theta)$ with respect to ξ is zero whenever $C - \eta - \theta = 0$. If however $C - \eta - \theta \neq 0$, then $\xi(C - \eta - \theta)$ can be made to approach $-\infty$. Since we need to maximise the dual, we can rule out the latter case and impose the following constraint on the dual variables:

$$C - \eta - \theta = 0. \tag{4.16}$$

The KKT conditions confine θ to be non-negative so we conclude that $\eta \leq C$. We now discuss two possible cases: if $(\ell_t - \epsilon)/\|\mathbf{g}_t\|_2^2 \leq C$, then we can plug Eq. (4.16) back into Eq. (4.14) and we recover the Lagrangian of the original GPA algorithm (see Eq. (4.5)). From this point and on, we can repeat the same derivation as

in the original GPA update and get $\eta_t = \max(0,\, \ell_t - \epsilon)/\|\mathbf{g}_t\|_2^2$. The other case is when $(\ell_t - \epsilon)/\|\mathbf{g}_t\|_2^2 > C$. This condition can be rewritten as

$$C\|\mathbf{g}_t\|_2^2 < \ell_t - \epsilon. \tag{4.17}$$

We also know that the constraint in Eq. (4.10) must hold at the optimum, so $\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) \leq \epsilon + \xi$. Using the explicit form of $\mathbf{w}$ given in Eq. (4.15), we can rewrite this constraint as $\ell_t - \epsilon \leq \eta\|\mathbf{g}_t\|_2^2 + \xi$. Combining this inequality with Eq. (4.17) gives

$$(C - \eta)\|\mathbf{g}_t\|_2^2 < \xi. \tag{4.18}$$

We now use our earlier conclusion that $\eta \leq C$ to obtain $\xi > 0$. Turning to the KKT complementary slackness condition, we know that $\theta\xi = 0$ at the optimum. Having concluded that ξ is strictly positive, we get that θ must equal zero. Plugging $\theta = 0$ into Eq. (4.16) gives $\eta = C$. To sum up, we used the KKT conditions to show that in the case where $(\ell_t - \epsilon)/\|\mathbf{g}_t\|_2^2 > C$, it is optimal to select $\eta = C$. Folding all of the possible cases into a single equation, we obtain the following learning rate for GPA-I:

$$\eta_t = \min\left\{C,\ \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2}\right\}. \tag{4.19}$$

Turning to the update of GPA-II, we again recall that $\ell_t \leq \epsilon$ leads to $\eta_t = 0$, and deal with those rounds for which $\ell_t > \epsilon$. The Lagrangian of the optimisation problem in Eq. (4.11) equals

$$\mathcal{L}(\mathbf{w}, \xi, \eta) = \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \,+\, C\xi^2 \,+\, \eta[\ell_t + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \epsilon - \xi], \tag{4.20}$$

where $\eta \geq 0$ is a Lagrange multiplier. Again, differentiating this Lagrangian with respect to the elements of $\mathbf{w}$ and setting these partial derivatives to zero gives Eq. (4.15), so we may write $\mathbf{w} = \mathbf{w}_t - \eta\mathbf{g}_t$. Differentiating the Lagrangian with respect to ξ and setting that partial derivative to zero results in

$$0 = \frac{\partial\mathcal{L}(\mathbf{w}, \xi, \eta)}{\partial\xi} = 2C\xi - \eta \quad \Longrightarrow \quad \xi = \frac{\eta}{2C}. \tag{4.21}$$

Expressing ξ as above and replacing $\mathbf{w}$ in Eq. (4.20) with $\mathbf{w}_t - \eta\mathbf{g}_t$, we can rewrite the Lagrangian as

$$\mathcal{L}(\eta) = -\frac{\eta^2}{2}\left(\|\mathbf{g}_t\|_2^2 + \frac{1}{2C}\right) \,+\, \eta(\ell_t - \epsilon). \tag{4.22}$$

Setting the derivative of the above to zero gives

$$0 = \frac{\partial \mathcal{L}(\eta)}{\partial \eta} = -\eta \left(\|\mathbf{g}_t\|_2^2 + \frac{1}{2C} \right) + \ell_t - \epsilon \quad \Longrightarrow \quad \eta_t = \frac{\ell_t - \epsilon}{\|\mathbf{g}_t\|_2^2 + \frac{1}{2C}}. \tag{4.23}$$

As in GPA and GPA-I, we can give a definition of η_t that complies with all cases:

$$\eta_t = \frac{\max(0,\, \ell_t - \epsilon)}{\|\mathbf{g}_t\|_2^2 + \frac{1}{2C}}. \tag{4.24}$$

4.3 Bayesian Passive-Aggressive Regression

Now that we have explained how to extend the PA regression framework to loss functions other than the ϵ-insensitive loss (ILF), and derived Algorithm 5 for that purpose, we proceed to address the remaining shortcomings mentioned in Section 4.1, namely *i*) the assumption of an isotropic weight covariance matrix, *ii*) the lack of probabilistic predictions, and *iii*) the sensitivity of the algorithm's performance to hyperparameter configurations.

To this end, we propose a Bayesian treatment of PA regression. The Bayesian paradigm supports online learning in a natural fashion: starting from the prior, the first training example produces a posterior distribution incorporating the evidence from the first example. This then becomes the prior distribution awaiting the arrival of the second example, and so on. The resulting sequence of posterior distributions allows for model (i.e. weights) and prediction uncertainty, via the predictive distribution over the targets y. Additionally, the Bayesian framework allows for the online adaptation of the hyperparameters, which is important since cross-validation techniques cannot be deployed in online settings.

A major challenge is the functional form of the ILF which causes the posterior distribution to have a different form from the prior distribution. Some form of approximation is thus required to carry out the sequential update. For this purpose, we develop a novel online variational inference scheme. As a byproduct, we obtain an online data-dependent mechanism to automatically adjust the hyperparameters, which basically consists in the sequential maximisation of the variational lower bound on the marginal likelihood inherent in the PA regression algorithm.

4.3.1 Related work

Perhaps the first probabilistic approach to PA learning was confidence-weighted (CW) learning [Dredze et al., 2008], which maintains a Gaussian distribution over weight vectors and updates it for each new training instance so that the Kullback-Leibler divergence between the new and the old distributions is minimised, and the probability of correct classification for that instance under the updated distribution meets a specified confidence level.

Strictly speaking, CW learning and its subsequently developed variants, namely adaptive regularisation of weights [Crammer et al., 2009] and soft-confidence weighted learning [Wang et al., 2012, 2016], are not Bayesian treatments of the PA framework since they all rely on different update criteria than the standard updates used in Bayesian inference. Recently, Shi and Zhu [2014, 2017] introduced online Bayesian passive-aggressive learning (BayesPA), which is an application of regularised Bayesian inference to the PA setting that subsumes Bayes' rule updates as well as the original PA algorithms. The key principle behind BayesPA is to combine the flexibility of Bayesian models with the discriminative power of large-margin methods such as PA. Whilst the authors of [Shi and Zhu, 2014, 2017] restricted themselves to classification tasks, Deng et al. [2016] adapted their work to regression problems and also deployed a variational procedure for approximate inference. Our work generalises the results in [Deng et al., 2016] in the following non-trivial aspects:

- [Deng et al., 2016] relies on a pseudo-likelihood, i.e. a likelihood function that is unnormalised with respect to the observations, while this paper considers a proper likelihood, which matters a lot when it comes to hyperparameter tuning;

- we derive a predictive distribution over the targets y_t, whereas Deng et al. [2016] restrict themselves to inferring the weights;

- while Deng et al. [2016] assume the hyperparameters to be fixed and user-defined, we introduce an online data-driven adaptation mechanism to tune them.

Of particular relevance to our setup is Kalman filtering [Kalman, 1960]. In the original Kalman filter, there are strong assumptions about the dynamical model, such as a linear state-space and a Gaussian noise model. However, we depart from the classical setting by assuming no particular dynamics for the states (i.e. weights). We defer the discussion of the relationship between the Kalman filter and our proposed Bayesian PA regression framework to Section 4.3.5.

4.3.2 Probabilistic interpretation of passive-aggressive regression

Passive-aggressive regression

In this paragraph, we briefly review the notion of online passive-aggressive (PA) regression, which was introduced in Crammer et al. [2006]. There are three types of PA regression algorithms, namely PA, PA-I and PA-II. For the remainder of this chapter, we shall confine ourselves to the PA-I variant.

The PA-I algorithm operates as follows. On every round $t = 1, 2, \ldots$, the algorithm receives an instance $\mathbf{x}_t \in \mathbb{R}^n$ and predicts a target value $\hat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t \in \mathbb{R}$, where $\mathbf{w}_t$ is the incrementally learned weight vector. After making a prediction, the algorithm is given the true target value $y_t \in \mathbb{R}$ and suffers an instantaneous loss measured by the ϵ-insensitive loss function (ILF), which is defined by

$$\ell^{\epsilon}(\mathbf{w}; (\mathbf{x}, y)) \equiv |y - \mathbf{w} \cdot \mathbf{x}|_{\epsilon} \equiv \max\Big\{0,\ |y - \mathbf{w} \cdot \mathbf{x}| - \epsilon\Big\}, \tag{4.25}$$

where the insensitivity parameter $\epsilon \geq 0$ controls the algorithm's sensitivity to prediction mistakes. This loss is zero when the predicted target deviates from the true target by less than ϵ units, and otherwise grows linearly with the discrepancy $|y - \hat{y}|$.

At the end of every round, the PA-I algorithm uses its current weight vector $\mathbf{w}_t$ along with the current example $(\mathbf{x}_t, y_t)$ to generate a new weight vector $\mathbf{w}_{t+1}$ which will be used to extend the prediction on the next round. Specifically, the PA-I weights are initialised to zero, i.e. $\mathbf{w}_1 = \mathbf{0}_{n \times 1}$, then sequentially updated according to the rule

$$\mathbf{w}_{t+1},\ \xi^* = \underset{\mathbf{w} \in \mathbb{R}^n,\ \xi}{\arg\min} \left\{ \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|_2^2 + C\xi \right\} \qquad \text{s.t.} \qquad \ell_t^{\epsilon}(\mathbf{w}) \leq \xi \ \text{ and } \ \xi \geq 0, \tag{4.26}$$

which holds for every integer $t \geq 1$, and where we have used $\ell_t^\epsilon(\mathbf{w})$ as a shorthand notation for $\ell^\epsilon(\mathbf{w}; (\mathbf{x}_t, y_t))$.

Probabilistic interpretation

For the purpose of providing a probabilistic interpretation to the PA-I regression algorithm, it is convenient to reformulate its update rule, i.e. Eq. (4.26), as a proximal operator, like so[6]:

$$\begin{aligned}
\mathbf{w}_{t+1},\, \xi^* &= \underset{\mathbf{w}\in\mathbb{R}^n,\, \xi}{\arg\min} \; \left\{ \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \,+\, C\xi \,:\, \ell_t^\epsilon(\mathbf{w}) \leq \xi,\, \xi \geq 0 \right\} \\
&= \underset{\mathbf{w}\in\mathbb{R}^n}{\arg\min} \; \left\{ \underset{\xi}{\arg\min} \; \left\{ \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \,+\, C\xi \,:\, \xi \geq \ell_t^\epsilon(\mathbf{w}),\, \xi \geq 0 \right\} \right\} \\
&= \underset{\mathbf{w}\in\mathbb{R}^n}{\arg\min} \; \left\{ C\ell_t^\epsilon(\mathbf{w}) \,+\, \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\}. \qquad (4.27)
\end{aligned}$$

The last step involved in the derivation of the desired probabilistic interpretation is to rewrite the PA-I objective above as a maximisation problem of the form

$$\begin{aligned}
&\min_{\mathbf{w}\in\mathbb{R}^n} \; \left\{ C\ell_t^\epsilon(\mathbf{w}) \,+\, \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\} = \max_{\mathbf{w}\in\mathbb{R}^n} \; \left\{ -C\ell_t^\epsilon(\mathbf{w}) \,-\, \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\} \\
&= \max_{\mathbf{w}\in\mathbb{R}^n} \; \log\left[\exp\left\{ -C\ell_t^\epsilon(\mathbf{w}) \,-\, \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\} \right]. \qquad (4.28)
\end{aligned}$$

From Eq. (4.28), we see that the individual likelihood associated with the t-th iteration of the PA-I regressor satisfies the relation

$$p(y_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon) \propto \exp\left\{ -C\ell_t^\epsilon(\mathbf{w}) \right\}. \qquad (4.29)$$

By integrating the right-hand side with respect to y_t and taking the inverse of the resulting expression, we identify the corresponding normalisation factor to be

[6]The reader is referred to [Parikh and Boyd, 2013] for a detailed discussion of proximal operators and algorithms.

$C/[2(1 + C\epsilon)]$, as demonstrated below:

$$\begin{aligned}
&\int_{\mathbb{R}} \exp\Big\{ -C\ell_t^{\epsilon}(\mathbf{w})\Big\}\,\mathrm{d}y_t \\
&= \int_{\{y_t\in\mathbb{R}\,:\,|y_t-\mathbf{w}\cdot\mathbf{x}_t|\leq\epsilon\}} \mathrm{d}y_t + \int_{\{y_t\in\mathbb{R}\,:\,|y_t-\mathbf{w}\cdot\mathbf{x}_t|>\epsilon\}} \exp\Big\{ -C(|y_t-\mathbf{w}\cdot\mathbf{x}_t|-\epsilon)\Big\}\,\mathrm{d}y_t \\
&= \int_{\mathbf{w}\cdot\mathbf{x}_t-\epsilon}^{\mathbf{w}\cdot\mathbf{x}_t+\epsilon} \mathrm{d}y_t + \exp\Big\{C(\mathbf{w}\cdot\mathbf{x}_t+\epsilon)\Big\}\int_{\mathbf{w}\cdot\mathbf{x}_t+\epsilon}^{\infty} \exp(-Cy_t)\,\mathrm{d}y_t \\
&\quad + \exp\Big\{ -C(\mathbf{w}\cdot\mathbf{x}_t-\epsilon)\Big\}\int_{-\infty}^{\mathbf{w}\cdot\mathbf{x}_t-\epsilon} \exp(Cy_t)\,\mathrm{d}y_t \\
&= 2\epsilon + \frac{1}{C}\exp\Big\{C(\mathbf{w}\cdot\mathbf{x}_t+\epsilon)\Big\}\exp\Big\{ -C(\mathbf{w}\cdot\mathbf{x}_t+\epsilon)\Big\} \\
&+ \quad \frac{1}{C}\exp\Big\{ -C(\mathbf{w}\cdot\mathbf{x}_t-\epsilon)\Big\}\exp\Big\{C(\mathbf{w}\cdot\mathbf{x}_t-\epsilon)\Big\} = 2C^{-1}(1+C\epsilon). \qquad (4.30)
\end{aligned}$$

Thus we can write

$$p(y_t|\mathbf{x}_t,\mathbf{w},C,\epsilon) = \frac{C}{2(1+C\epsilon)}\exp\Big\{ -C\ell_t^{\epsilon}(\mathbf{w})\Big\}. \qquad (4.31)$$

To complete the specification of the probabilistic model underlying PA-I regression, we need to identify the prior $p(\mathbf{w})$ over the weight vector that is implicitly used by the algorithm. To this end, we consider the PA-I objective as expressed in Eq. (4.28) at time $t = 1$, and recall that PA-I sets the initial weight vector to the zero vector, by convention. This yields $p(\mathbf{w}) \propto \exp\{-\|\mathbf{w}\|_2^2/2\}$, which we recognise as an isotropic Gaussian distribution of the form $\mathcal{N}(\mathbf{w}|\mathbf{0}_{n\times 1}, \mathbf{I}_n)$. Since one of our desiderata here is to rectify the inability of PA-I to handle data with non-spherical distributions — which in turn is due to its use of the squared Euclidean norm (see Section 4.1 for details) — we shall consider a slightly more general Gaussian prior, with mean $\boldsymbol{\mu}_0^{\mathbf{w}}$ and covariance $\boldsymbol{\Sigma}_0^{\mathbf{w}}$, i.e.

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_0^{\mathbf{w}}, \boldsymbol{\Sigma}_0^{\mathbf{w}}), \qquad (4.32)$$

where we have set the initial time index to 0 instead of 1 to adhere to the usual notational convention in Bayesian analysis.

Because the individual likelihood in Eq. (4.31) does not take the form of the exponential of a quadratic form in $\mathbf{w}$, the Gaussian PA-I weight prior is not a conjugate distribution for this likelihood function. A remedy for this situation is to explore *data augmentation* techniques [Tanner and Wong, 1987; Polson

and Scott, 2011]. The basic idea behind DA methods is to achieve conjugacy between prior and data during inference by introducing augmented variables. The purpose of the next subsection is to show that, in our case, this leads to a Gaussian mixture representation of Eq. (4.31), which in turn will allow us to perform Bayesian inference for PA-I regression using familiar tools developed for Gaussian linear models.

4.3.3 Mixture representation

Our main theoretical result in this section expresses the likelihood contribution $p(y_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon)$ as a dual location-scale mixture of Gaussian distributions based on the aforementioned principle of data augmentation. The result allows us to pair observation y_t with latent variables λ_t and κ_t in such a way that $p(y_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon)$ is the marginal of a joint distribution $p(y_t, \lambda_t, \kappa_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon)$ in which $\mathbf{w}$ appears as part of a quadratic form. This implies that the *augmented likelihood* contribution $p(y_t, \lambda_t, \kappa_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon)$ is conjugate to a multivariate Gaussian prior distribution, in particular the PA-I weight prior in Eq. (4.32). In effect, the augmented data space allows the awkward PA-I optimality criterion in Eq. (4.26) to be expressed as a conditionally Gaussian linear model that is familiar to most Bayesian statisticians.

Theorem 4.3.1. *The likelihood contribution from observation* $(\mathbf{x}_t, y_t)$ *can be expressed as*

$$\begin{aligned} p(y_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon) &= \mathcal{Z} \exp\Big\{ -C \max\Big\{0,\ |y_t - \mathbf{w}\cdot\mathbf{x}_t| - \epsilon\Big\}\Big\} \\ &= \int_0^\infty \int_0^\infty p(y_t, \lambda_t, \kappa_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon)\, \mathrm{d}\lambda_t\, \mathrm{d}\kappa_t, \end{aligned} \tag{4.33}$$

where

$$\begin{aligned} p(y_t, \lambda_t, \kappa_t|\mathbf{x}_t, \mathbf{w}, C, \epsilon) = \mathcal{Z} &\frac{\sqrt{\gamma}}{\sqrt{2\pi\lambda_t}} \exp\Big\{ -\frac{1}{2\gamma^{-1}\lambda_t}(\lambda_t - \mathbf{w}\cdot\mathbf{x}_t + y_t - \epsilon)^2 \Big\} \\ &\times \frac{\sqrt{\gamma}}{\sqrt{2\pi\kappa_t}} \exp\Big\{ -\frac{1}{2\gamma^{-1}\kappa_t}(\kappa_t + \mathbf{w}\cdot\mathbf{x}_t - y_t - \epsilon)^2 \Big\}, \end{aligned} \tag{4.34}$$

where we have defined $\gamma \equiv C/2$ *and* $\mathcal{Z} \equiv \gamma/(1 + 2\gamma\epsilon)$.

Proof. We proceed in two steps:

1. first, we establish the integral identity

$$\exp\Big\{-2\gamma\max\big\{0,\,u\big\}\Big\} = \int_0^\infty \frac{\sqrt{\gamma}}{\sqrt{2\pi\lambda}}\exp\Big\{-\frac{1}{2\gamma^{-1}\lambda}(u+\lambda)^2\Big\}\,\mathrm{d}\lambda; \quad (4.35)$$

2. then we apply the above identity to the ILF, after proving that it can be decomposed as follows:

$$\begin{aligned}&\exp\Big\{-C\max\big\{0,\,|y_t-\mathbf{w}\cdot\mathbf{x}_t|-\epsilon\big\}\Big\}\\ &=\exp\Big\{-2\gamma\max\big\{0,\,y_t-\mathbf{w}\cdot\mathbf{x}_t-\epsilon\big\}\Big\}\exp\Big\{-2\gamma\max\big\{0,\,\mathbf{w}\cdot\mathbf{x}_t-y_t-\epsilon\big\}\Big\}.\end{aligned} \quad (4.36)$$

Proof of Step 1 Expanding out the square in the integrand of Eq. (4.35) and rearranging terms, we obtain

$$\begin{aligned}&\int_0^\infty \frac{\sqrt{\gamma}}{\sqrt{2\pi\lambda}}\exp\Big\{-\frac{1}{2\gamma^{-1}\lambda}(u+\lambda)^2\Big\}\,\mathrm{d}\lambda\\ &=\exp(-\gamma u)\int_0^\infty \sqrt{\frac{\gamma}{2\pi}}\lambda^{-1/2}\exp\Big\{-\frac{1}{2}(\gamma\lambda+\gamma u^2\lambda^{-1})\Big\}\,\mathrm{d}\lambda\\ &=\exp\Big\{-\gamma(|u|+u)\Big\}\underbrace{\int_0^\infty \mathcal{GIG}(\lambda|1/2,\,\gamma u^2,\,\gamma)\,\mathrm{d}\lambda}_{=1},\end{aligned} \quad (4.37)$$

where

$$\mathcal{GIG}(x|\delta,\chi,\psi) = \frac{(\psi/\chi)^{\delta/2}}{2K_\delta(\sqrt{\chi\psi})}x^{\delta-1}\exp\Big\{-\frac{1}{2}(\psi x+\chi x^{-1})\Big\} \quad (4.38)$$

is the density of the generalised inverse Gaussian distribution, in which $K_\delta(\cdot)$ denotes the modified Bessel function of the second kind (see Appendix B.5 for details). Finally, using the identity $\max\{0,u\} = (|u|+u)/2$ gives the desired result.

Proof of Step 2 Using the identity

$$\max\big\{0,\,|x|-\epsilon\big\} = \max\big\{0,\,x-\epsilon\big\} + \max\big\{0,\,-x-\epsilon\big\} \quad (4.39)$$

and the definition of γ, we may write

$$\begin{aligned}&-C\max\big\{0,\,|y_t-\mathbf{w}\cdot\mathbf{x}_t|-\epsilon\big\}\\ &=-2\gamma\max\big\{0,\,y_t-\mathbf{w}\cdot\mathbf{x}_t-\epsilon\big\}-2\gamma\max\big\{0,\,\mathbf{w}\cdot\mathbf{x}_t-y_t-\epsilon\big\},\end{aligned} \quad (4.40)$$

whence

$$\begin{aligned}&\exp\Big\{-C\max\Big\{0,\,|y_t-\mathbf{w}\cdot\mathbf{x}_t|-\epsilon\Big\}\Big\}\\&=\exp\Big\{-2\gamma\max\Big\{0,\,y_t-\mathbf{w}\cdot\mathbf{x}_t-\epsilon\Big\}\Big\}\exp\Big\{-2\gamma\max\Big\{0,\,\mathbf{w}\cdot\mathbf{x}_t-y_t-\epsilon\Big\}\Big\}.\end{aligned}\tag{4.41}$$

We shall now make use of the integral identity derived in Step 1, i.e. Eq. (4.35). The latter allows us to express the factors in Eq. (4.41) as

$$\begin{aligned}&\exp\Big\{-2\gamma\max\Big\{0,\,y_t-\mathbf{w}\cdot\mathbf{x}_t-\epsilon\Big\}\Big\}\\&=\int_0^\infty\frac{\sqrt{\gamma}}{\sqrt{2\pi\lambda_t}}\exp\Big\{-\frac{1}{2\gamma^{-1}\lambda_t}(\lambda_t-\mathbf{w}\cdot\mathbf{x}_t+y_t-\epsilon)^2\Big\}\,\mathrm{d}\lambda_t,\end{aligned}\tag{4.42}$$

and

$$\begin{aligned}&\exp\Big\{-2\gamma\max\Big\{0,\,\mathbf{w}\cdot\mathbf{x}_t-y_t-\epsilon\Big\}\Big\}\\&=\int_0^\infty\frac{\sqrt{\gamma}}{\sqrt{2\pi\kappa_t}}\exp\Big\{-\frac{1}{2\gamma^{-1}\kappa_t}(\kappa_t+\mathbf{w}\cdot\mathbf{x}_t-y_t-\epsilon)^2\Big\}\,\mathrm{d}\kappa_t,\end{aligned}\tag{4.43}$$

respectively. Multiplying the above integrals together completes the proof. □

Before discussing how to perform inference in this augmented framework, it is worth emphasising that the related paper by Deng et al. [2016] has the parameter γ as part of the prior distribution over $\mathbf{w}$, while here in Eq. (4.33) it is included instead in the likelihood. This is done because *i*) it puts the latent variables λ_t, κ_t and the regularisation parameter γ together, and *ii*) it allows more freedom in the choice of the prior for $\mathbf{w}$. Additionally, Eq. (4.33) has an interesting interpretation, in that the ϵ-insensitive loss function behaves like a global-local shrinkage distribution. Specifically, by virtue of the properties of the Gaussian distribution (see Appendix B.4.1), we have

$$\begin{aligned}&p(y_t,\lambda_t,\kappa_t|\mathbf{x}_t,\mathbf{w},C,\epsilon)\\&=\mathcal{Z}\,\mathcal{N}(y_t|\mathbf{w}\cdot\mathbf{x}_t+\epsilon-\lambda_t,\,\gamma^{-1}\lambda_t)\times\mathcal{N}(y_t|\mathbf{w}\cdot\mathbf{x}_t-\epsilon+\kappa_t,\,\gamma^{-1}\kappa_t)\\&\propto\mathcal{N}\Big[y_t\,\Big|\,\mathbf{w}\cdot\mathbf{x}_t+(\lambda_t^{-1}+\kappa_t^{-1})^{-1}(\lambda_t^{-1}-\kappa_t^{-1})\epsilon,\;\gamma^{-1}(\lambda_t^{-1}+\kappa_t^{-1})^{-1}\Big],\end{aligned}\tag{4.44}$$

which tells us that γ^{-1} corresponds to a 'global' scaling of the variance, whereas $(\lambda_t^{-1}+\kappa_t^{-1})^{-1}$ represents the 'local' scaling, or *relative* variance, for observation t.

4.3.4 Inference

Having laid out the probabilistic model associated with PA-I regression, Bayesian inference proceeds by computing, from Bayes' rule, the posterior over all unknowns $\mathbf{w}$, $\lambda_{1:t}$[7] and $\kappa_{1:t}$, given the data $D_{1:t} \equiv \{(\mathbf{x}_\tau, y_\tau)\}_{\tau=1}^t$ and the hyperparameters $\boldsymbol{\psi} \equiv (C, \epsilon)$:

$$p(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t} | \mathcal{D}_{1:t}, \boldsymbol{\psi}) \propto p(y_{1:t}, \lambda_{1:t}, \kappa_{1:t} | \mathbf{x}_{1:t}, \mathbf{w}, \boldsymbol{\psi}) p(\mathbf{w}). \tag{4.45}$$

In a fully Bayesian treatment, we would define additional prior distributions over the hyperparameters $\boldsymbol{\psi}$. Here we adopt instead an online version of the type-II maximum likelihood ('evidence') framework [Berger, 1985; MacKay, 1992], in which the optimal set of hyperparameters is found by maximising the marginal likelihood, or evidence[8], $p(y_{1:t}|\mathbf{x}_{1:t}, \boldsymbol{\psi}) = \int p(y_{1:t}, \lambda_{1:t}, \kappa_{1:t}|\mathbf{x}_{1:t}, \mathbf{w}, \boldsymbol{\psi}) p(\mathbf{w}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t} \, \mathrm{d}\kappa_{1:t}$ with respect to $\boldsymbol{\psi}$ in a sequential manner (see Section 4.3.6).

In order to construct an online PA-I regression algorithm within the Bayesian framework, we have to find out how the posterior distribution in Eq. (4.45) changes when a new data point $\mathcal{D}_{t+1} = \{(\mathbf{x}_{t+1}, y_{t+1})\}$ is observed. It can be easily shown that the new posterior corresponding to the new data set $\mathcal{D}_{1:t+1} = \mathcal{D}_{1:t} \cup \mathcal{D}_{t+1}$ is given in terms of the old posterior and the likelihood of the new example by

$$\begin{aligned} &p(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1} | \mathcal{D}_{1:t+1}, \boldsymbol{\psi}) \\ &= \frac{p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) p(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t} | \mathcal{D}_{1:t}, \boldsymbol{\psi})}{\int p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) p(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t} | \mathcal{D}_{1:t}, \boldsymbol{\psi}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1}}. \end{aligned} \tag{4.46}$$

Technically speaking, the above expression does not have the form of an online algorithm, because it requires the knowledge of the *entire* old data set $\mathcal{D}_{1:t}$. The basic idea to turn this into an online algorithm is to replace the true posterior $p(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1} | \mathcal{D}_{1:t+1}, \boldsymbol{\psi})$ by a simpler distribution $q_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$, from which we omit the dependence on the hyperparameters $\boldsymbol{\psi}$ to keep the notation uncluttered. Hence, the resulting Bayesian PA-I algorithm for regression will be based on the repetition of two basic steps:

[7] $x_{1:t}$ denotes $x_1, x_2, \ldots, x_t$.

[8] Henceforth, when the context is clear, we shall use a condensed integral notation: all integrals are definite integrals over the entire domain of interest.

- **Update step** In this step, the old approximate posterior $q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})$ is used to perform an update of the form (4.46):

$$\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \equiv \frac{p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})}{\widetilde{q}_t(y_{t+1} | \mathbf{x}_{t+1})}, \quad (4.47)$$

  where

$$\begin{aligned} &\widetilde{q}_t(y_{t+1} | \mathbf{x}_{t+1}) \\ &\equiv \int p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1}. \end{aligned} \quad (4.48)$$

- **Approximation step** The new posterior $\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ in Eq. (4.47) cannot be computed directly since we cannot perform the normalising integral on the right-hand side. We must therefore seek an effective approximation. To mitigate the loss of information, the new approximate posterior $q_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ should be chosen so that it is sufficiently close to $\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$. It is not clear a priori which measure of dissimilarity between distributions should be used, and different choices are likely to lead to different algorithms. We shall opt for the Kullback-Leibler (KL) divergence between q_{t+1} and $\widetilde{q}_{t+1}$, i.e.

$$\begin{aligned} &\mathrm{KL}(q_{t+1} \,||\, \widetilde{q}_{t+1}) \\ &= \int q_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \log \frac{q_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})}{\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})} \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1}, \end{aligned} \quad (4.49)$$

  and discuss how to derive the optimal q_{t+1} by minimising this quantity in the following subsection. We refer to the resulting algorithm for sequential distributional updates as *online variational inference*.

The treatment here is inspired by the work of Opper [1998] which is also based on the minimisation of a KL divergence but of the reverse form, i.e. $\mathrm{KL}(\widetilde{q}_{t+1} \,||\, q_{t+1})$, which gives the approximation rather different properties (since the KL divergence is not symmetric in its arguments). This alternative form of approximate inference is known as expectation propagation (EP) [Minka, 2001a,b], and results in moment matching in the case of approximate posteriors belonging to the exponential family. We did not explore this avenue because the moments of $\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ turn out to be analytically intractable.

Online variational inference

In online variational inference (OVI), we aim to find the approximate posterior in an online setting where the data arrive sequentially. Similar to streaming Bayesian updating (see Eq. (4.46)), we use the previous approximated posterior as the new prior distribution. For example, on round $t+1$, the optimal variational distribution is the solution to the following optimisation problem:

$$q_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) = \underset{q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})}{\arg\min} \mathrm{KL}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \,||\, \widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big]. \tag{4.50}$$

Using the result (4.47) for the posterior distribution $\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$, we can write

$$\begin{aligned} &\mathrm{KL}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \,||\, \widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big] \\ &= \int q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \Big[\log \frac{q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})}{p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})} \\ &\qquad + \log \widetilde{q}_t(y_{t+1} | \mathbf{x}_{t+1}) \Big] \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1}, \end{aligned} \tag{4.51}$$

whence

$$\begin{aligned} &\log \widetilde{q}_t(y_{t+1} | \mathbf{x}_{t+1}) \\ &= \mathcal{L}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big] \,+\, \mathrm{KL}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \,||\, \widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big], \end{aligned} \tag{4.52}$$

where we have defined

$$\begin{aligned} &\mathcal{L}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big] \\ &\equiv \int q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \log \frac{p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})}{q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})} \\ &\qquad \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1}. \end{aligned} \tag{4.53}$$

Thus, minimising the KL divergence of interest is tantamount to maximising the lower bound $\mathcal{L}[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})]$ with respect to $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$. If we allow any possible choice for $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$, then the maximum of the lower bound occurs when the KL divergence vanishes, which occurs when $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ equals the target posterior distribution $\widetilde{q}_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$. However, as we already pointed out, the Bayesian PA-I model is such that working with this target posterior is intractable.

We therefore consider instead a restricted family of approximating distributions $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ and then seek the member of this family for which the KL divergence is minimised. Our goal is to restrict the family sufficiently that they comprise only tractable distributions, while at the same time allowing the family to be sufficiently rich and flexible that it can provide a good approximation to the posterior of interest. It is important to emphasise that the restriction is imposed purely to achieve tractability, and that subject to this requirement, we should use as rich a family of approximating distributions as possible. In particular, there is no 'overfitting' associated with highly flexible distributions. Using more flexible approximations simply allows us to approach the target posterior more closely.

Factorised distributions

One way to restrict the family of approximating distributions is to use a parametric distribution $q_{\boldsymbol{\omega}}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ governed by a set of parameters $\boldsymbol{\omega}$. The lower bound $\mathcal{L}(\cdot)$ then becomes a function of $\boldsymbol{\omega}$, and we can exploit standard non-linear optimisation techniques to determine the optimal values of the parameters.

Here, we consider an alternative way in which to restrict the family of distributions $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$. We assume that this distribution factorises with respect to its arguments, so that

$$q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) = q(\mathbf{w})q(\lambda_{1:t+1})q(\kappa_{1:t+1}) = q(\mathbf{w}) \times \prod_{\tau=1}^{t+1} q(\lambda_\tau) \times \prod_{\tau=1}^{t+1} q(\kappa_\tau). \quad (4.54)$$

It should be emphasised that we are making no further assumptions about the distribution. In particular, we place no restriction on the functional forms of the individual factors $q(\mathbf{w})$, $q(\lambda_\tau)$ and $q(\kappa_\tau)$, for $\tau \in [t+1] \equiv \{1, 2, \ldots, t+1\}$. This factorised form of variational inference corresponds to an approximation framework developed in physics and known as *mean field theory* [Parisi, 1988].

Among all distributions $q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ having the form (4.54), we now seek that distribution for which the lower bound $\mathcal{L}[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})]$ is largest. We therefore wish to make a variational optimisation of $\mathcal{L}[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})]$ with respect to all of the distributions $q(\mathbf{w})$, $q(\lambda_\tau)$ and $q(\kappa_\tau)$, $\tau \in [t+1]$, which we do

by optimising with respect to each of the factors in turn. To achieve this, we first substitute Eq. (4.54) into Eq. (4.53) and then dissect out the dependence on one of the factors, say $q(\mathbf{w})$, without loss of generality. We then obtain

$$\begin{aligned}
&\mathcal{L}\Big[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})\Big] \\
&= \int q(\mathbf{w}) q(\lambda_{1:t+1}) q(\kappa_{1:t+1}) \Big\{ \log \Big[p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t}) \Big] \\
&\qquad - \log q(\mathbf{w}) - \log q(\lambda_{1:t+1}) - \log q(\kappa_{1:t+1}) \Big\} \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:t+1} \, \mathrm{d}\kappa_{1:t+1} \\
&= \int q(\mathbf{w}) \, \langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q(\lambda_{1:t+1}) q(\kappa_{1:t+1})} \, \mathrm{d}\mathbf{w} - \int q(\mathbf{w}) \log q(\mathbf{w}) \, \mathrm{d}\mathbf{w} + \text{const} \\
&= \int q(\mathbf{w}) \log \frac{\exp \Big\{ \langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q(\lambda_{1:t+1}) q(\kappa_{1:t+1})} \Big\}}{q(\mathbf{w})} \, \mathrm{d}\mathbf{w} + \text{const}, \qquad (4.55)
\end{aligned}$$

where 'const' denotes terms independent of $\mathbf{w}$, $\langle \cdot \rangle_{p(x)}$ signifies an expectation with respect to the distribution $p(x)$, and

$$E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \equiv \log \Big[p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t}) \Big]. \quad (4.56)$$

Now suppose we keep the factors $q(\lambda_{1:t+1})$ and $q(\kappa_{1:t+1})$ fixed and maximise $\mathcal{L}[q(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})]$ with respect to all possible forms for the distribution $q(\mathbf{w})$. This is easily done by recognising that Eq. (4.55) is a negative KL divergence between $q(\mathbf{w})$ and a distribution proportional to $\exp\{\langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q(\lambda_{1:t+1}) q(\kappa_{1:t+1})}\}$. As a result, maximising Eq. (4.55) is equivalent to minimising the KL divergence, and the minimum occurs at

$$q_{t+1}(\mathbf{w}) \propto \exp \Big\{ \langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q_{t+1}(\lambda_{1:t+1}) q_{t+1}(\kappa_{1:t+1})} \Big\}. \qquad (4.57)$$

Repeating the same process for all the other factors, we get

$$q_{t+1}(\lambda_{1:t+1}) \propto \exp \Big\{ \langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q_{t+1}(\mathbf{w}) q_{t+1}(\kappa_{1:t+1})} \Big\}, \qquad (4.58)$$

$$q_{t+1}(\kappa_{1:t+1}) \propto \exp \Big\{ \langle \, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \, \rangle_{q_{t+1}(\mathbf{w}) q_{t+1}(\lambda_{1:t+1})} \Big\}. \qquad (4.59)$$

The set of equations (4.57)-(4.59) represent a set of consistency conditions for the maximum of the lower bound subject to the factorisation constraint. However, they do not represent an explicit solution because the expression on their respective right-hand sides depends on expectations computed with respect to the other

factors. We will therefore seek a consistent solution by first initialising all of the factors appropriately and then optimising each one iteratively in turn, keeping all of the other factors fixed at their current estimates. Convergence is guaranteed because the KL divergence is convex with respect to each of the factors [Boyd and Vandenberghe, 2004].

Determining $q_{t+1}(\mathbf{w})$

Optimally, $q_{t+1}(\mathbf{w})$ is Gaussian since the augmented likelihood contribution (4.44) is quadratic in $\mathbf{w}$ and the PA-I weight prior is Gaussian. We prove this assertion below.

Using the definition of the augmented likelihood contribution (4.44), the factorisation assumption (4.54), and picking out just those terms that involve $\mathbf{w}$, we have

$$\begin{aligned}
&E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) = \log p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) + \log q_t(\mathbf{w}) + \text{const} \\
&= -\frac{1}{2\gamma^{-1}(\lambda_{t+1}^{-1} + \kappa_{t+1}^{-1})^{-1}} \Big[y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1} - (\lambda_{t+1}^{-1} + \kappa_{t+1}^{-1})^{-1}(\lambda_{t+1}^{-1} - \kappa_{t+1}^{-1})\epsilon \Big]^2 \\
&\quad + \log q_t(\mathbf{w}) + \text{const} \\
&= -\frac{1}{2\gamma^{-1}(\lambda_{t+1}^{-1} + \kappa_{t+1}^{-1})^{-1}} (y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1})^2 + \gamma\epsilon(\lambda_{t+1}^{-1} - \kappa_{t+1}^{-1})(y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1}) \\
&\quad + \log q_t(\mathbf{w}) + \text{const}. \qquad (4.60)
\end{aligned}$$

From this we obtain

$$\begin{aligned}
&\log q_{t+1}(\mathbf{w}) = \langle\, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \,\rangle_{q_{t+1}(\lambda_{1:t+1}) q_{t+1}(\kappa_{1:t+1})} \\
&= \text{const} + \left\langle -\frac{1}{2\gamma^{-1}(\lambda_{t+1}^{-1} + \kappa_{t+1}^{-1})^{-1}} (y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1})^2 \right\rangle_{q_{t+1}(\lambda_{t+1}) q_{t+1}(\kappa_{t+1})} \\
&\quad + \Big\langle \gamma\epsilon(\lambda_{t+1}^{-1} - \kappa_{t+1}^{-1})(y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1}) \Big\rangle_{q_{t+1}(\lambda_{t+1}) q_{t+1}(\kappa_{t+1})} + \log q_t(\mathbf{w}) \\
&= \text{const} - \frac{1}{2\gamma^{-1}(\langle\lambda_{t+1}^{-1}\rangle_{t+1} + \langle\kappa_{t+1}^{-1}\rangle_{t+1})^{-1}} (y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1})^2 \\
&\quad + \gamma\epsilon(\langle\lambda_{t+1}^{-1}\rangle_{t+1} - \langle\kappa_{t+1}^{-1}\rangle_{t+1})(y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1}) + \log q_t(\mathbf{w}) \\
&= \text{const} - \frac{1}{2\gamma^{-1}(\langle\lambda_{t+1}^{-1}\rangle_{t+1} + \langle\kappa_{t+1}^{-1}\rangle_{t+1})^{-1}} \Big[y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1} \\
&\quad - (\langle\lambda_{t+1}^{-1}\rangle_{t+1} + \langle\kappa_{t+1}^{-1}\rangle_{t+1})^{-1}(\langle\lambda_{t+1}^{-1}\rangle_{t+1} - \langle\kappa_{t+1}^{-1}\rangle_{t+1})\epsilon \Big]^2 + \log q_t(\mathbf{w}), \qquad (4.61)
\end{aligned}$$

where we have once again applied the factorisation assumption in the second equality, $\langle x \rangle_t$ is a shorthand for $\langle x \rangle_{q_t(x)}$, and the last equality is simply the result of completing the square with respect to $\mathbf{w}$ (see Appendix B.4.1).

We recognise the second term on the right-hand side of Eq. (4.61) as the exponent of a Gaussian distribution over y_{t+1} conditioned on $\mathbf{w}$. Taking the exponential of both sides of Eq. (4.61) thus gives

$$q_{t+1}(\mathbf{w}) \propto \mathcal{N}(y_{t+1}|\mathbf{w} \cdot \mathbf{x}_{t+1} + m_{t+1},\, s_{t+1}^2) \times q_t(\mathbf{w}), \tag{4.62}$$

where

$$m_{t+1} \equiv (\langle \lambda_{t+1}^{-1} \rangle_{t+1} + \langle \kappa_{t+1}^{-1} \rangle_{t+1})^{-1} (\langle \lambda_{t+1}^{-1} \rangle_{t+1} - \langle \kappa_{t+1}^{-1} \rangle_{t+1}) \epsilon, \tag{4.63}$$

$$s_{t+1}^2 \equiv \gamma^{-1} (\langle \lambda_{t+1}^{-1} \rangle_{t+1} + \langle \kappa_{t+1}^{-1} \rangle_{t+1})^{-1}. \tag{4.64}$$

The recursion (4.62) is initialised using the prior (4.32), i.e. we set $q_0(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_0^{\mathbf{w}},\, \boldsymbol{\Sigma}_0^{\mathbf{w}})$. This causes $q_{t+1}(\mathbf{w})$ to be a Gaussian of the form

$$q_{t+1}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{t+1}^{\mathbf{w}},\, \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}}), \tag{4.65}$$

where the posterior covariance and mean are respectively:

$$\boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} = \Big[(\boldsymbol{\Sigma}_t^{\mathbf{w}})^{-1} + s_{t+1}^{-2} \mathbf{x}_{t+1} \mathbf{x}_{t+1}^{\mathrm{T}} \Big]^{-1}, \tag{4.66}$$

$$\boldsymbol{\mu}_{t+1}^{\mathbf{w}} = \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} \Big[s_{t+1}^{-2} (y_{t+1} - m_{t+1}) \mathbf{x}_{t+1} + (\boldsymbol{\Sigma}_t^{\mathbf{w}})^{-1} \boldsymbol{\mu}_t^{\mathbf{w}} \Big]. \tag{4.67}$$

It is straightforward to prove this statement by induction. Suppose $q_t(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t^{\mathbf{w}},\, \boldsymbol{\Sigma}_t^{\mathbf{w}})$. Substituting into Eq. (4.62), we obtain a joint Gaussian over $\mathbf{w}$ and y_{t+1}, expressed as the product of the underlying marginal over $\mathbf{w}$ and conditional over y_{t+1} given $\mathbf{w}$. Using the Gaussian properties from Appendix B.4.1, we can rearrange this joint Gaussian in the form of the product of a Gaussian marginal over y_{t+1} times a Gaussian conditional over $\mathbf{w}$ given y_{t+1}, which yields the desired result.

Determining $q_{t+1}(\lambda_{1:t+1})$

From the factorisation assumption (4.54), we know that

$$q_{t+1}(\lambda_{1:t+1}) = \prod_{\tau=1}^{t+1} q_\tau(\lambda_\tau), \tag{4.68}$$

so we can limit our derivations to one of these factors, say $q_{t+1}(\lambda_{t+1})$. By inspection of the form of the augmented likelihood in Eq. (4.34), we see that $q_{t+1}(\lambda_{t+1})$ must

be a GIG distribution. To see this, we start by picking out the terms in the function E_{t+1} that depend on λ_{t+1}, which gives

$$\begin{aligned} E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) &= \text{const} + \log p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1} | \mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi}) \\ &= \text{const} - \frac{1}{2}\log \lambda_{t+1} - \frac{1}{2\gamma^{-1}\lambda_{t+1}}(\lambda_{t+1} - \mathbf{w}\cdot\mathbf{x}_{t+1} + y_{t+1} - \epsilon)^2. \end{aligned} \tag{4.69}$$

Expanding out the square, this simplifies to

$$E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) = \text{const} - \frac{1}{2}\log \lambda_{t+1} - \frac{\lambda_{t+1}}{2\gamma^{-1}} - \frac{1}{2\gamma^{-1}\lambda_{t+1}}(y_{t+1} - \mathbf{w}\cdot\mathbf{x}_{t+1} - \epsilon)^2, \tag{4.70}$$

and so

$$\begin{aligned} &\langle\, E_{t+1}(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1}) \,\rangle_{q_{t+1}(\mathbf{w}) q_{t+1}(\kappa_{1:t+1})} \\ &= \text{const} - \frac{1}{2}\log \lambda_{t+1} - \frac{1}{2}\Big[\gamma\lambda_{t+1} + \gamma \left\langle (y_{t+1} - \mathbf{w}\cdot\mathbf{x}_{t+1} - \epsilon)^2 \right\rangle_{q_{t+1}(\mathbf{w})} \lambda_{t+1}^{-1}\Big]. \end{aligned} \tag{4.71}$$

Up to constant of proportionality, this expression is equal to the log of the density of a GIG distribution with parameters $\delta = 1/2$, $\psi = \gamma$ and

$$\begin{aligned} \chi_{t+1}^{\lambda} &\equiv \gamma \left\langle (y_{t+1} - \mathbf{w}\cdot\mathbf{x}_{t+1} - \epsilon)^2 \right\rangle_{q_{t+1}(\mathbf{w})} \\ &= \gamma\Big[(y_{t+1} - \epsilon)^2 - 2(y_{t+1} - \epsilon)\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\mu}_{t+1}^{\mathbf{w}} + \mathbf{x}_{t+1}^{\mathrm{T}}\langle \mathbf{w}\mathbf{w}^{\mathrm{T}}\rangle_{q_{t+1}(\mathbf{w})}\mathbf{x}_{t+1}\Big]. \end{aligned} \tag{4.72}$$

Using the standard property that the covariance matrix is equal to the autocorrelation matrix minus the outer product of the mean, we can obtain the required moment as follows

$$\langle \mathbf{w}\mathbf{w}^{\mathrm{T}}\rangle_{q_{t+1}(\mathbf{w})} = \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} + \boldsymbol{\mu}_{t+1}^{\mathbf{w}}(\boldsymbol{\mu}_{t+1}^{\mathbf{w}})^{\mathrm{T}}. \tag{4.73}$$

Plugging this back into the definition of χ_{t+1}^{λ} and rearranging terms gives

$$\chi_{t+1}^{\lambda} = \gamma\Big[(y_{t+1} - \boldsymbol{\mu}_{t+1}^{\mathbf{w}}\cdot\mathbf{x}_{t+1} - \epsilon)^2 + \mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_{t+1}^{\mathbf{w}}\mathbf{x}_{t+1}\Big]. \tag{4.74}$$

We conclude that

$$q_{t+1}(\lambda_{1:t+1}) = \prod_{\tau=1}^{t+1} \mathcal{GIG}(\lambda_\tau | 1/2,\, \chi_\tau^{\lambda},\, \gamma). \tag{4.75}$$

We still need to evaluate $\langle \lambda_{t+1}^{-1}\rangle_{q_{t+1}(\lambda_{t+1})}$, as the moments of $q_{t+1}(\mathbf{w})$ depend on this quantity. To this end, we make use of Corollary B.5.1, from which it follows that

$$\langle \lambda_{t+1}^{-1}\rangle_{q_{t+1}(\lambda_{t+1})} = \sqrt{\frac{\gamma}{\chi_{t+1}^{\lambda}}}. \tag{4.76}$$

Determining $q_{t+1}(\kappa_{1:t+1})$

We will not discuss how to derive the variational posterior over the latent variables $\kappa_{1:t+1}$. Suffice it to say that repeating the above calculations, we obtain

$$q_{t+1}(\kappa_{1:t+1}) = \prod_{\tau=1}^{t+1} q_\tau(\kappa_\tau), \qquad q_\tau(\kappa_\tau) = \mathcal{GIG}(\kappa_\tau|1/2,\, \chi_\tau^\kappa,\, \gamma), \tag{4.77}$$

where

$$\chi_\tau^\kappa \equiv \gamma\Big[(y_\tau - \boldsymbol{\mu}_\tau^{\mathbf{w}} \cdot \mathbf{x}_\tau + \epsilon)^2 + \mathbf{x}_\tau^{\mathrm{T}} \boldsymbol{\Sigma}_\tau^{\mathbf{w}} \mathbf{x}_\tau\Big]. \tag{4.78}$$

Similarly, we have

$$\langle \kappa_{t+1}^{-1} \rangle_{q_{t+1}(\kappa_{t+1})} = \sqrt{\frac{\gamma}{\chi_{t+1}^\kappa}}. \tag{4.79}$$

Summary

The results of our proposed OVI scheme are summarised below. To keep the notation less cluttered, we shall omit the subscript $t+1$ from the latent variables λ_{t+1} and κ_{t+1} (this also makes sense given that they are local variables, and thereby independent across time steps).

Weight updates

$$q_{t+1}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{t+1}^{\mathbf{w}},\, \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}}) \tag{4.80}$$

$$\boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} = \Big[(\boldsymbol{\Sigma}_t^{\mathbf{w}})^{-1} + s_{t+1}^{-2} \mathbf{x}_{t+1} \mathbf{x}_{t+1}^{\mathrm{T}}\Big]^{-1} \tag{4.81}$$

$$\boldsymbol{\mu}_{t+1}^{\mathbf{w}} = \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} \Big[s_{t+1}^{-2}(y_{t+1} - m_{t+1})\mathbf{x}_{t+1} + (\boldsymbol{\Sigma}_t^{\mathbf{w}})^{-1} \boldsymbol{\mu}_t^{\mathbf{w}}\Big] \tag{4.82}$$

$$m_{t+1} = \Big[\langle \lambda^{-1} \rangle_{q_{t+1}(\lambda)} + \langle \kappa^{-1} \rangle_{q_{t+1}(\kappa)}\Big]^{-1} \Big[\langle \lambda^{-1} \rangle_{q_{t+1}(\lambda)} - \langle \kappa^{-1} \rangle_{q_{t+1}(\kappa)}\Big] \epsilon \tag{4.83}$$

$$s_{t+1}^2 = \gamma^{-1} \Big[\langle \lambda^{-1} \rangle_{q_{t+1}(\lambda)} + \langle \kappa^{-1} \rangle_{q_{t+1}(\kappa)}\Big]^{-1}. \tag{4.84}$$

Latent-variable updates

$$q_{t+1}(\lambda) = \mathcal{GIG}(\lambda|1/2,\, \chi^{\lambda}_{t+1},\, \gamma) \tag{4.85}$$

$$\chi^{\lambda}_{t+1} = \gamma\Big[(y_{t+1} - \boldsymbol{\mu}^{\mathbf{w}}_{t+1} \cdot \mathbf{x}_{t+1} - \epsilon)^2 + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t+1}\mathbf{x}_{t+1}\Big] \tag{4.86}$$

$$\langle \lambda^{-1} \rangle_{q_{t+1}(\lambda)} = \sqrt{\frac{\gamma}{\chi^{\lambda}_{t+1}}} \tag{4.87}$$

$$q_{t+1}(\kappa) = \mathcal{GIG}(\kappa|1/2,\, \chi^{\kappa}_{t+1},\, \gamma) \tag{4.88}$$

$$\chi^{\kappa}_{t+1} = \gamma\Big[(y_{t+1} - \boldsymbol{\mu}^{\mathbf{w}}_{t+1} \cdot \mathbf{x}_{t+1} + \epsilon)^2 + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t+1}\mathbf{x}_{t+1}\Big] \tag{4.89}$$

$$\langle \kappa^{-1} \rangle_{q_{t+1}(\kappa)} = \sqrt{\frac{\gamma}{\chi^{\kappa}_{t+1}}}. \tag{4.90}$$

4.3.5 Relation to Kalman filtering

At this point, it is worth spending a moment to understand the relationship between OVI and a closely related algorithm, namely the Kalman filter (KF) [Kalman, 1960]. Specifically, the moment update equations (4.81)-(4.82) for the variational weight posterior are somewhat reminiscent of the KF equations (see e.g. [Bishop, 2006, p. 639]). Here we will discuss the similarities, and differences, between these.

Let us begin with the variational weight covariance matrix in Eq. (4.81). Using the Woodbury inversion identity (see Appendix A.3), we can write

$$\begin{aligned}
\boldsymbol{\Sigma}^{\mathbf{w}}_{t+1} &= \Big[(\boldsymbol{\Sigma}^{\mathbf{w}}_{t})^{-1} + s^{-2}_{t+1}\mathbf{x}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1}\Big]^{-1} \\
&= \boldsymbol{\Sigma}^{\mathbf{w}}_{t} - \boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}(s^2_{t+1} + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1})^{-1}\mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t} \\
&= (\mathbf{I}_n - \mathbf{k}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1})\boldsymbol{\Sigma}^{\mathbf{w}}_{t},
\end{aligned} \tag{4.91}$$

where we have defined $\mathbf{k}_{t+1} \equiv \boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}(s^2_{t+1} + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1})^{-1}$, which is reminiscent of the *Kalman gain*. Plugging Eq. (4.91) into Eq. (4.82), we obtain

$$\begin{aligned}
&\boldsymbol{\mu}^{\mathbf{w}}_{t+1} = (\mathbf{I}_n - \mathbf{k}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1})\boldsymbol{\mu}^{\mathbf{w}}_{t} + s^{-2}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t+1}\mathbf{x}_{t+1}(y_{t+1} - m_{t+1}) \\
&= (\mathbf{I}_n - \mathbf{k}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1})\boldsymbol{\mu}^{\mathbf{w}}_{t} + \frac{s^{-2}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}(s^2_{t+1} + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}) - s^{-2}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}}{s^2_{t+1} + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}} \\
&\quad \times (y_{t+1} - m_{t+1}) \\
&= (\mathbf{I}_n - \mathbf{k}_{t+1}\mathbf{x}^{\mathrm{T}}_{t+1})\boldsymbol{\mu}^{\mathbf{w}}_{t} + \underbrace{\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1}(s^2_{t+1} + \mathbf{x}^{\mathrm{T}}_{t+1}\boldsymbol{\Sigma}^{\mathbf{w}}_{t}\mathbf{x}_{t+1})^{-1}}_{=\mathbf{k}_{t+1}}(y_{t+1} - m_{t+1}) \\
&= \boldsymbol{\mu}^{\mathbf{w}}_{t} + \mathbf{k}_{t+1}(y_{t+1} - \boldsymbol{\mu}^{\mathbf{w}}_{t} \cdot \mathbf{x}_{t+1} - m_{t+1}).
\end{aligned} \tag{4.92}$$

The above alternative expressions for the posterior weight mean and covariance highlight the difference between the Kalman filter and our proposed Bayesian PA regression framework: in our framework, we have kept the prior $p(\mathbf{w})$ static, meaning that $\mathbf{w}$ is assumed to be random but not evolving over time, whereas the KF imposes stochastic dynamics on the weights by means of a (Gaussian) *transition distribution* $p(\mathbf{w}_{t+1}|\mathbf{w}_t)$. Tackling such a scenario is trivial from our perspective, as one only needs to modify the PA probabilistic model slightly in order to arrive at a dynamic algorithm. In particular, in addition to the update step, one needs to employ a prediction step, according to the assumed dynamics of the weight vector. Here, this means generalising our model to a *linear Gaussian state-space model* (LGSSM) whose transition and emission distributions are respectively:

$$p(\mathbf{w}_{t+1}|\mathbf{w}_t) = \mathcal{N}(\mathbf{w}_{t+1}|\mathbf{A}_t\mathbf{w}_t,\, \boldsymbol{\Gamma}_t^{\mathbf{w}}) \tag{4.93}$$

$$p(y_{t+1}|\mathbf{x}_{t+1}, \mathbf{w}_{t+1}, \boldsymbol{\psi}) = \mathcal{N}(y_{t+1}|\mathbf{w}_{t+1}\cdot\mathbf{x}_{t+1} + m_{t+1},\, s_{t+1}^2). \tag{4.94}$$

As shown in e.g. [Särkkä, 2013, Theorem 4.2], the Bayesian filtering equations for the LGSSM above can be evaluated in closed form and the resulting distributions are Gaussian:

$$p(\mathbf{w}_{t+1}|\mathcal{D}_{1:t}, \boldsymbol{\psi}) = \mathcal{N}(\mathbf{w}_{t+1}|\boldsymbol{\mu}_{t+1/2}^{\mathbf{w}},\, \boldsymbol{\Sigma}_{t+1/2}^{\mathbf{w}}) \tag{4.95}$$

$$p(\mathbf{w}_{t+1}|\mathcal{D}_{1:t+1}, \boldsymbol{\psi}) = \mathcal{N}(\mathbf{w}_{t+1}|\boldsymbol{\mu}_{t+1}^{\mathbf{w}},\, \boldsymbol{\Sigma}_{t+1}^{\mathbf{w}}). \tag{4.96}$$

The parameters of these distributions can be computed via the following KF prediction and update steps.

- Prediction step:

$$\boldsymbol{\mu}_{t+1/2}^{\mathbf{w}} = \mathbf{A}_t\boldsymbol{\mu}_t^{\mathbf{w}}, \qquad \boldsymbol{\Sigma}_{t+1/2}^{\mathbf{w}} = \mathbf{A}_t\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{A}_t^{\mathrm{T}} + \boldsymbol{\Gamma}_t^{\mathbf{w}}; \tag{4.97}$$

- Update step:

$$\boldsymbol{\mu}_{t+1}^{\mathbf{w}} = \boldsymbol{\mu}_{t+1/2}^{\mathbf{w}} + \mathbf{k}_{t+1}(y_{t+1} - \mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\mu}_{t+1/2}^{\mathbf{w}} - m_{t+1}) \tag{4.98}$$

$$\boldsymbol{\Sigma}_{t+1}^{\mathbf{w}} = (\mathbf{I}_n - \mathbf{k}_{t+1}\mathbf{x}_{t+1}^{\mathrm{T}})\boldsymbol{\Sigma}_{t+1/2}^{\mathbf{w}} \tag{4.99}$$

$$\mathbf{k}_{t+1} = \boldsymbol{\Sigma}_{t+1/2}^{\mathbf{w}}\mathbf{x}_{t+1}(\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_{t+1/2}^{\mathbf{w}}\mathbf{x}_{t+1} + s_{t+1}^2)^{-1}. \tag{4.100}$$

Note that Eqs. (4.91) and (4.92) are recovered as a special case if, for all t, we use $\mathbf{A}_t = \mathbf{I}_n$ and the degenerate covariance matrix $\mathbf{\Gamma}_t^{\mathbf{w}} = \mathbf{0}_{n\times n}$.

4.3.6 Hyperparameter tuning

So far, we have considered the inference problem for Bayesian PA regression assuming that the hyperparameters $\boldsymbol{\psi} = (C, \epsilon)$ are known. Next, we consider the determination of these hyperparameters using approximate maximum posterior (MAP) estimation, which is based on approximating the hyperparameter posterior $p(\boldsymbol{\psi}|\mathcal{D}_{1:t+1})$ and maximising the resulting approximate distribution with respect to $\boldsymbol{\psi}$. In order to remain as conceptually close as possible to the original PA framework, we avoid specifying a prior $p(\boldsymbol{\psi})$ over the hyperparameters, which is equivalent to assuming a flat prior, that is $p(\boldsymbol{\psi}) = \text{const}$. In this case, MAP estimation boils down to *type-II maximum likelihood* [Berger, 1985; MacKay, 1992], as we shall see shortly. Despite this 'simplification', the presented framework can be readily extended to more general hyperpriors.

Applying Bayes' rule recursively, the hyperparameter posterior can be expressed as

$$p(\boldsymbol{\psi}|\mathcal{D}_{1:t+1}) \propto p(y_{t+1}|\mathbf{x}_{t+1}, \mathcal{D}_{1:t}, \boldsymbol{\psi})p(\boldsymbol{\psi}|\mathcal{D}_{1:t}) = p(\boldsymbol{\psi}) \prod_{\tau=1}^{t+1} p(y_\tau|\mathbf{x}_\tau, \mathcal{D}_{1:\tau-1}, \boldsymbol{\psi}), \tag{4.101}$$

with the convention $p(y_1|\mathbf{x}_1, \mathcal{D}_{1:0}, \boldsymbol{\psi}) = p(y_1|\mathbf{x}_1, \boldsymbol{\psi})$. In practice, it is more convenient to maximise the log of this quantity. Under our assumption of a flat prior, the MAP solution is equivalent to setting $\boldsymbol{\psi}$ to that value which maximises the log likelihood of observing the data, a procedure known as type-II maximum likelihood in the literature:

$$\boldsymbol{\psi}_{t+1}^{\text{ML}} = \arg\max_{\boldsymbol{\psi}\in\Psi} \sum_{\tau=1}^{t+1} \log p(y_\tau|\mathbf{x}_\tau, \mathcal{D}_{1:\tau-1}, \boldsymbol{\psi}), \tag{4.102}$$

where $\Psi \equiv (0, \infty) \times [0, \infty)$ is the hyperparameter space (i.e. the set of admissible hyperparameter values).

By application of the sum and product rules of probability, we can write

$$\begin{aligned}&\log p(y_\tau|\mathbf{x}_\tau, \mathcal{D}_{1:\tau-1}, \boldsymbol{\psi}) \\ &= \log \int p(y_\tau, \lambda_\tau, \kappa_\tau|\mathbf{x}_\tau, \mathbf{w}, \boldsymbol{\psi}) p(\mathbf{w}, \lambda_{1:\tau-1}, \kappa_{1:\tau-1}|\mathcal{D}_{1:\tau-1}, \boldsymbol{\psi}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:\tau} \, \mathrm{d}\kappa_{1:\tau}\end{aligned} \tag{4.103}$$

which is analytically intractable. While we can make use of the variational posteriors $q_{\tau-1}(\mathbf{w}, \lambda_{1:\tau-1}, \kappa_{1:\tau-1})$ and the definition (4.48) to approximate this quantity as

$$\begin{aligned}&\log p(y_\tau|\mathbf{x}_\tau, \mathcal{D}_{1:\tau-1}, \boldsymbol{\psi}) \\ &\simeq \log \int p(y_\tau, \lambda_\tau, \kappa_\tau|\mathbf{x}_\tau, \mathbf{w}, \boldsymbol{\psi}) q_{\tau-1}(\mathbf{w}, \lambda_{1:\tau-1}, \kappa_{1:\tau-1}) \, \mathrm{d}\mathbf{w} \, \mathrm{d}\lambda_{1:\tau} \, \mathrm{d}\kappa_{1:\tau} \\ &= \log \widetilde{q}_{\tau-1}(y_\tau|\mathbf{x}_\tau),\end{aligned} \tag{4.104}$$

evaluating the above integral leads to an objective function whose gradient with respect to the hyperparameters is computationally intractable (see Section 4.3.7). Nevertheless, from Eq. (4.52), we know we can lower bound the log of the distribution $\widetilde{q}_{\tau-1}(y_\tau|\mathbf{x}_\tau)$ as follows:

$$\log \widetilde{q}_{\tau-1}(y_\tau|\mathbf{x}_\tau) \geq \mathcal{L}[q_\tau(\mathbf{w}, \lambda_{1:\tau}, \kappa_{1:\tau})]. \tag{4.105}$$

Thus, instead of aiming to solve the intractable problem in Eq. (4.102), we may adopt a variational type-II maximum likelihood approach whereby we optimise the cumulative fitted variational lower bound, where by 'fitted' we mean the lower bound expressed as a function of the hyperparameters, after convergence of the OVI procedure:

$$\max_{\boldsymbol{\psi} \in \Psi} \sum_{\tau=1}^{t+1} \mathcal{L}_\tau(\boldsymbol{\psi}), \qquad \mathcal{L}_\tau(\boldsymbol{\psi}) \equiv \mathcal{L}[q_\tau(\mathbf{w}, \lambda_{1:\tau}, \kappa_{1:\tau}|\boldsymbol{\psi})]. \tag{4.106}$$

We solve this surrogate problem by alternating between optimisation steps with respect to the hyperparameters and the variational parameters. That is, we first determine the approximately optimal hyperparameter vector $\widehat{\boldsymbol{\psi}}_{t+1}^{\mathrm{ML}}$ for round $t+1$, based on data observed up to and including round t. Next, we carry out the iterations (4.81), (4.82), (4.87) and (4.90) until convergence, keeping $\widehat{\boldsymbol{\psi}}_{t+1}^{\mathrm{ML}}$ **fixed**. Once these iterations converge, we update the hyperparameters once again, by way of maximising the running sum of fitted variational lower bounds with respect to $\boldsymbol{\psi}$, and so on.

To ensure an online adaptation mechanism for tuning the hyperparameters, we solve the problem in Eq. (4.106) by means of the online gradient descent (OGD) algorithm [Zinkevich, 2003a], in which the approximate type-II ML hyperparameter values, $\widehat{\boldsymbol{\psi}}^{\mathrm{ML}}$, are sequentially updated via the recursion

$$\widehat{\boldsymbol{\psi}}^{\mathrm{ML}}_{\tau+1} = \Pi_{\Psi}\big[\widehat{\boldsymbol{\psi}}^{\mathrm{ML}}_{\tau} + \eta_{\tau}\nabla\mathcal{L}_{\tau}\big(\widehat{\boldsymbol{\psi}}^{\mathrm{ML}}_{\tau}\big)\big], \tag{4.107}$$

with the optimal learning rate given by $\eta_{\tau} = 1/\sqrt{\tau}$ [Zinkevich, 2003a, Theorem 1], and where $\Pi_{\Psi}(\cdot)$ denotes the projection onto the nearest point in the set Ψ, i.e.

$$\Pi_{\Psi}(\boldsymbol{\psi}') = \underset{\boldsymbol{\psi}\in\Psi}{\arg\min}\ \|\boldsymbol{\psi} - \boldsymbol{\psi}'\|_2^2. \tag{4.108}$$

Variational lower bound

In addition to opening up the possibility of determining the optimal hyperparameter values, evaluating the lower bound during the re-estimation procedure allows us to test for convergence. It can also provide a valuable check on both the mathematical expressions for the solutions and their software implementation, because at each step of the iterative re-estimation procedure, the value of this bound should not decrease. We can take this a stage further to provide a deeper test of the correctness of both the mathematical derivation of the update equations and of their software implementations by using finite differences to check that each update does indeed give a (constrained) maximum of the bound.

From Eq. (4.53) and the factorisation assumption (4.54), the lower bound at step τ is given by

$$\begin{aligned}
\mathcal{L}_{\tau}(\psi) &= \int q_{\tau}(\mathbf{w}, \lambda_{1:\tau}, \kappa_{1:\tau}) \log \frac{p(y_{\tau}, \lambda_{\tau}, \kappa_{\tau}|\mathbf{x}_{\tau}, \mathbf{w}, \boldsymbol{\psi}) q_{\tau-1}(\mathbf{w}, \lambda_{1:\tau-1}, \kappa_{1:\tau-1})}{q_{\tau}(\mathbf{w}, \lambda_{1:\tau}, \kappa_{1:\tau})} \\
&\qquad \mathrm{d}\mathbf{w}\,\mathrm{d}\lambda_{1:\tau}\,\mathrm{d}\kappa_{1:\tau} \\
&= \big\langle \log p(y_{\tau}, \lambda_{\tau}, \kappa_{\tau}|\mathbf{x}_{\tau}, \mathbf{w}, \boldsymbol{\psi}) \big\rangle_{q_{\tau}(\mathbf{w})q_{\tau}(\lambda_{\tau})q_{\tau}(\kappa_{\tau})} + \big\langle \log q_{\tau-1}(\mathbf{w}) \big\rangle_{q_{\tau}(\mathbf{w})} \\
&\quad - \mathrm{H}[q_{\tau-1}(\lambda_{1:\tau-1})] - \mathrm{H}[q_{\tau-1}(\kappa_{1:\tau-1})] + \mathrm{H}[q_{\tau}(\mathbf{w})] + \mathrm{H}[q_{\tau}(\lambda_{1:\tau})] + \mathrm{H}[q_{\tau}(\kappa_{1:\tau})] \\
&= \big\langle \log p(y_{\tau}, \lambda_{\tau}, \kappa_{\tau}|\mathbf{x}_{\tau}, \mathbf{w}, \boldsymbol{\psi}) \big\rangle_{q_{\tau}(\mathbf{w})q_{\tau}(\lambda_{\tau})q_{\tau}(\kappa_{\tau})} + \big\langle \log q_{\tau-1}(\mathbf{w}) \big\rangle_{q_{\tau}(\mathbf{w})} \\
&\quad + \mathrm{H}[q_{\tau}(\mathbf{w})] + \mathrm{H}[q_{\tau}(\lambda_{\tau})] + \mathrm{H}[q_{\tau}(\kappa_{\tau})],
\end{aligned} \tag{4.109}$$

where $\mathrm{H}[p(x)] \equiv -\langle \log p(x) \rangle_{p(x)}$ signifies the entropy of the distribution $p(x)$, and the last equality follows from the additive property of the entropy for independent random variables (see Appendix A.1.3). Evaluation of the various terms is straightforward, making use of the results from previous sections, along with Corollary B.5.1, Eq. (B.37), Eq. (B.20) and Eq. (B.42) in that order, which gives

$$\begin{aligned}
&\langle \log p(y_\tau, \lambda_\tau, \kappa_\tau | \mathbf{x}_\tau, \mathbf{w}, \boldsymbol{\psi}) \rangle_{q_\tau(\mathbf{w}) q_\tau(\lambda_\tau) q_\tau(\kappa_\tau)} = \log \frac{\gamma}{2\pi} + \log \mathcal{Z} - \frac{1}{2}\Big[\langle \log \lambda_\tau \rangle_{q_\tau(\lambda_\tau)} \\
&+ \langle \log \kappa_\tau \rangle_{q_\tau(\kappa_\tau)}\Big] - \frac{\gamma}{2}\Big[\langle \lambda_\tau \rangle_{q_\tau(\lambda_\tau)} + \langle \kappa_\tau \rangle_{q_\tau(\kappa_\tau)}\Big] - \frac{1}{2}\Big[\chi_\tau^\lambda \langle \lambda_\tau^{-1} \rangle_{q_\tau(\lambda_\tau)} - \chi_\tau^\kappa \langle \kappa_\tau^{-1} \rangle_{q_\tau(\kappa_\tau)}\Big] + 2\gamma\epsilon \\
&= \log \frac{\gamma}{2\pi} + \log \mathcal{Z} - \frac{1}{2}\Big[\langle \log \lambda_\tau \rangle_{q_\tau(\lambda_\tau)} + \langle \log \kappa_\tau \rangle_{q_\tau(\kappa_\tau)}\Big] + \Big(-1 - \frac{1}{2}\sqrt{\gamma \chi_\tau^\lambda} - \frac{1}{2}\sqrt{\gamma \chi_\tau^\kappa}\Big) \\
&\quad - \frac{1}{2}\Big(\sqrt{\gamma \chi_\tau^\lambda} - \sqrt{\gamma \chi_\tau^\kappa}\Big) + 2\gamma\epsilon \\
&= \log \frac{\gamma}{2\pi} + \log \mathcal{Z} - \frac{1}{2}\Big[\langle \log \lambda_\tau \rangle_{q_\tau(\lambda_\tau)} + \langle \log \kappa_\tau \rangle_{q_\tau(\kappa_\tau)}\Big] - \sqrt{\gamma \chi_\tau^\lambda} + 2\gamma\epsilon - 1 \qquad (4.110) \\
&\langle \log q_{\tau-1}(\mathbf{w}) \rangle_{q_\tau(\mathbf{w})} \\
&= -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}}| - \frac{1}{2}\Big\langle (\mathbf{w} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}})^{\mathrm{T}} (\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}})^{-1} (\mathbf{w} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}}) \Big\rangle_{q_\tau(\mathbf{w})} \\
&= -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}}| - \frac{1}{2}(\boldsymbol{\mu}_\tau^{\mathbf{w}} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}})^{\mathrm{T}} (\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}})^{-1} (\boldsymbol{\mu}_\tau^{\mathbf{w}} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}}) \\
&\quad - \frac{1}{2}\mathrm{Tr}\Big[(\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}})^{-1} \boldsymbol{\Sigma}_\tau^{\mathbf{w}}\Big] \qquad (4.111) \\
&\mathrm{H}[q_\tau(\mathbf{w})] = \frac{n}{2}[1 + \log(2\pi)] + \frac{1}{2}\log|\boldsymbol{\Sigma}_\tau^{\mathbf{w}}| \qquad (4.112) \\
&\mathrm{H}[q_\tau(\lambda_\tau)] = \frac{1}{2}\left(1 - \log\frac{\gamma}{2\pi}\right) + \frac{1}{2}\langle \log \lambda_\tau \rangle_{q_\tau(\lambda_\tau)} \qquad (4.113) \\
&\mathrm{H}[q_\tau(\kappa_\tau)] = \frac{1}{2}\left(1 - \log\frac{\gamma}{2\pi}\right) + \frac{1}{2}\langle \log \kappa_\tau \rangle_{q_\tau(\kappa_\tau)}. \qquad (4.114)
\end{aligned}$$

Some simplifications and combination of terms can be performed when the above expressions are plugged back into Eq. (4.109). Doing so yields

$$\begin{aligned}
\mathcal{L}_\tau(\boldsymbol{\psi}) = \log \mathcal{Z} + 2\gamma\epsilon - \sqrt{\gamma \chi_\tau^\lambda} + \frac{1}{2}\Big(\log|\boldsymbol{\Sigma}_\tau^{\mathbf{w}}| - \log|\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}}|\Big) \\
- \frac{1}{2}(\boldsymbol{\mu}_\tau^{\mathbf{w}} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}})^{\mathrm{T}} (\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}})^{-1} (\boldsymbol{\mu}_\tau^{\mathbf{w}} - \boldsymbol{\mu}_{\tau-1}^{\mathbf{w}}) + \frac{1}{2}\mathrm{Tr}\Big[\mathbf{I}_n - (\boldsymbol{\Sigma}_{\tau-1}^{\mathbf{w}})^{-1} \boldsymbol{\Sigma}_\tau^{\mathbf{w}}\Big]. \qquad (4.115)
\end{aligned}$$

4.3.7 Prediction

Once the hyperparameters have been estimated and the corresponding OVI iterations have converged, our ultimate goal is to make sequential one-step (next datum)

lookahead predictions of the output variable y_{t+1} at each round t, averaging over all sources of posterior uncertainty

$$\begin{aligned} &p(y_{t+1}|\mathbf{x}_{t+1}, \mathcal{D}_{1:t}, \boldsymbol{\psi}) \\ &= \int p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1}|\mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi})\, p(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t}|\mathcal{D}_{1:t}, \boldsymbol{\psi})\, \mathrm{d}\mathbf{w}\, \mathrm{d}\lambda_{1:t+1}\, \mathrm{d}\kappa_{1:t+1} \end{aligned} \tag{4.116}$$

and conditioning on the optimal hyperparameter values $\boldsymbol{\psi} = \widehat{\boldsymbol{\psi}}_{t+1}^{\mathrm{ML}}$.

As we pointed out earlier, the $n(t+1)^2$-dimensional integral over $(\mathbf{w}, \lambda_{1:t+1}, \kappa_{1:t+1})$ cannot be computed analytically. To overcome this issue, we replace the exact posterior $p(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t}|\mathcal{D}_{1:t}, \boldsymbol{\psi})$ with our variational approximation $q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})$, which gives

$$\begin{aligned} &p(y_{t+1}|\mathbf{x}_{t+1}, \mathcal{D}_{1:t}, \boldsymbol{\psi}) \approx \widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1}, \boldsymbol{\psi}) \\ &= \int p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1}|\mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi})\, q_t(\mathbf{w}, \lambda_{1:t}, \kappa_{1:t})\, \mathrm{d}\mathbf{w}\, \mathrm{d}\lambda_{1:t+1}\, \mathrm{d}\kappa_{1:t+1} \\ &= \int \Big[\int p(y_{t+1}, \lambda_{t+1}, \kappa_{t+1}|\mathbf{x}_{t+1}, \mathbf{w}, \boldsymbol{\psi})\, \mathrm{d}\lambda_{t+1}\, \mathrm{d}\kappa_{t+1} \Big]\, q_t(\mathbf{w})\, \mathrm{d}\mathbf{w} \\ &= \int \mathcal{Z} \exp\Big\{ -C \max\big\{0,\, |y_{t+1} - \mathbf{w} \cdot \mathbf{x}_{t+1}| - \epsilon \big\} \Big\} \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t^{\mathbf{w}},\, \boldsymbol{\Sigma}_t^{\mathbf{w}})\, \mathrm{d}\mathbf{w}, \end{aligned} \tag{4.117}$$

where, in the second equality, we have made use of the factorisation assumption (4.54), along with the fact that $q_t(\lambda_{1:t})$ and $q_t(\kappa_{1:t})$ integrate to 1 by definition.

To compute the predictions, it would appear that we need to carry out an integral in n dimensions. However, since the ILF depends on $\mathbf{w}$ via the scalar product $\mathbf{w} \cdot \mathbf{x}_{t+1}$, we only require the integral over the one-dimensional projection $h_{t+1} \equiv \mathbf{w} \cdot \mathbf{x}_{t+1}$ [Barber and Bishop, 1998, Appendix A.1]. That is

$$\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1}, \boldsymbol{\psi}) = \int \mathcal{Z} \exp\Big\{ -C \max\big\{0,\, |y_{t+1} - h_{t+1}| - \epsilon \big\} \Big\}\, q_t(h_{t+1})\, \mathrm{d}h_{t+1}. \tag{4.118}$$

Since under our variational approximation $\mathbf{w}$ is Gaussian distributed, then so is the linear projection h_{t+1} (see Appendix B.4.1):

$$q_t(h_{t+1}) = \mathcal{N}(h_{t+1}|\boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1},\, \mathbf{x}_{t+1}^{\mathrm{T}} \boldsymbol{\Sigma}_t^{\mathbf{w}} \mathbf{x}_{t+1}). \tag{4.119}$$

Using this result, we can further simplify Eq. (4.118) as follows:

$$\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1}, \boldsymbol{\psi}) = \int \mathcal{Z} \exp\Big\{ -C \max\big\{0,\, |y_{t+1} - h_{t+1}(u)| - \epsilon \big\} \Big\} \mathcal{N}(u|0, 1)\, \mathrm{d}u, \tag{4.120}$$

where $h_{t+1}(u) \equiv u\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}} + \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1}$. Predictions may then be made by numerically evaluating this one-dimensional integral. An alternative, not immediately apparent, approach is to derive a closed-form expression for this integral, which we discuss in detail below.

Analytical derivation of the predictive distribution

At first sight, it might seem that the integral in Eq. (4.120) is analytically intractable. However, it is actually possible to evaluate it in closed form, by noting that the non-Gaussian factor in the underlying integrand can be expressed a mixture of a uniform and two shifted exponential distributions, as shown below:

$$
\begin{aligned}
&\mathcal{Z}\exp\Big\{-C\max\Big\{0,\,|y_{t+1}-h_{t+1}(u)|-\epsilon\Big\}\Big\}\\
&=\mathcal{Z}\Big\{\mathbb{1}_{[h_{t+1}(u)-\epsilon,\,h_{t+1}(u)+\epsilon]}(y_{t+1})\\
&\qquad+\mathbb{1}_{[h_{t+1}(u)-\epsilon,\,h_{t+1}(u)+\epsilon]^{\mathsf{c}}}(y_{t+1})\exp\Big\{-C[\,|y_{t+1}-h_{t+1}(u)|-\epsilon]\Big\}\Big\}\\
&=\pi\frac{\mathbb{1}_{[h_{t+1}(u)-\epsilon,\,h_{t+1}(u)+\epsilon]}(y_{t+1})}{2\epsilon}\\
&\quad+\frac{1-\pi}{2}\mathbb{1}_{(-\infty,\,h_{t+1}(u)-\epsilon)}(y_{t+1})\,C\exp\Big\{-C[-y_{t+1}+h_{t+1}(u)-\epsilon]\Big\}\\
&\quad+\frac{1-\pi}{2}\mathbb{1}_{(h_{t+1}(u)+\epsilon,\,\infty)}(y_{t+1})\,C\exp\Big\{-C[y_{t+1}-h_{t+1}(u)-\epsilon]\Big\} \qquad (4.121)\\
&=\pi\,\mathcal{U}(y_{t+1}|h_{t+1}(u)-\epsilon,\;h_{t+1}(u)+\epsilon)\\
&\quad+\frac{1-\pi}{2}\Big[\mathcal{E}(-y_{t+1}|C,\,\epsilon-h_{t+1}(u))+\mathcal{E}(y_{t+1}|C,\,\epsilon+h_{t+1}(u))\Big], \qquad (4.122)
\end{aligned}
$$

where S^{c} denotes the complement of a set S, $\mathbb{1}_S(\cdot)$ the indicator function of S, $\pi \equiv C\epsilon/(1+C\epsilon)$ is the mixing coefficient and $\mathcal{E}(x|\lambda,\theta)$ is the density of the shifted exponential distribution (see Appendix B.3).

Using Eq. (4.121), the integral in Eq. (4.120) may be decomposed into a sum of three integrals, namely

$$\mathfrak{I}_{t+1}^{0} \equiv \int \pi\frac{\mathbb{1}_{[h_{t+1}(u)-\epsilon,\,h_{t+1}(u)+\epsilon]}(y_{t+1})}{2\epsilon}\,\mathcal{N}(u|0,1)\,\mathrm{d}u, \qquad (4.123)$$

$$\mathfrak{I}_{t+1}^{-} \equiv \int \frac{1-\pi}{2}\mathbb{1}_{(-\infty,\,h_{t+1}(u)-\epsilon)}(y_{t+1})\,C\exp\Big\{-C[-y_{t+1}+h_{t+1}(u)-\epsilon]\Big\}\mathcal{N}(u|0,1)\,\mathrm{d}u \qquad (4.124)$$

and

$$\mathfrak{I}_{t+1}^{+} \equiv \int \frac{1-\pi}{2} \mathbb{1}_{(h_{t+1}(u)+\epsilon,\,\infty)}(y_{t+1})\, C \exp\Big\{ -C[y_{t+1} - h_{t+1}(u) - \epsilon]\Big\} \mathcal{N}(u|0,1)\,\mathrm{d}u. \tag{4.125}$$

To evaluate these integrals, it is necessary to express the indicator functions in terms of u, as follows:

$$\mathbb{1}_{[h_{t+1}(u)-\epsilon,\, h_{t+1}(u)+\epsilon]}(y_{t+1}) = \mathbb{1}_{\left[\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\epsilon}{\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}},\, \frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\epsilon}{\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right]}(u) \tag{4.126}$$

$$\mathbb{1}_{(-\infty,\, h_{t+1}(u)-\epsilon)}(y_{t+1}) = \mathbb{1}_{\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\epsilon}{\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}},\, \infty\right)}(u) \tag{4.127}$$

$$\mathbb{1}_{(h_{t+1}(u)+\epsilon,\,\infty)}(y_{t+1}) = \mathbb{1}_{\left(-\infty,\, \frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\epsilon}{\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)}(u). \tag{4.128}$$

It immediately follows that

$$\begin{aligned}\mathfrak{I}_{t+1}^{0} &= \frac{\pi}{2\epsilon} \int_{(y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\epsilon)/\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}^{(y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\epsilon)/\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}} \mathcal{N}(u|0,1)\,\mathrm{d}u \\ &= \frac{\pi}{4\epsilon}\left[\operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right) - \operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right],\end{aligned} \tag{4.129}$$

where we have made use of the following link between the error function, $\operatorname{erf}(x) \equiv (2/\sqrt{\pi})\int_0^x \exp\{-z^2\}\,\mathrm{d}z$, and the standard Gaussian cumulative distribution function (CDF), denoted by $\Phi(\cdot)$:

$$\Phi(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right]. \tag{4.130}$$

For each of the remaining integrals $\mathfrak{I}_{t+1}^{-}$ and $\mathfrak{I}_{t+1}^{+}$, we still need to gather together the terms which depend on u in the integrand and use the technique of 'completing the square' (Appendix B.4.1), so as to arrive at a Gaussian-like integral. Doing so, we find that

$$\begin{aligned}&\exp\Big\{ -C[-y_{t+1} + h_{t+1}(u) - \epsilon]\Big\}\mathcal{N}(u|0,1) \\ &= \exp\Big\{C(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1} + \epsilon)\Big\} \times \frac{1}{\sqrt{2\pi}} \exp\Big\{ -\frac{1}{2}u^2 - uC\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}\Big\} \\ &= \exp\left\{C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1} + \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} + \epsilon\right)\right\}\mathcal{N}\Big(u\,\Big|\, -C\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}},\, 1\Big).\end{aligned} \tag{4.131}$$

Similarly, we have

$$\begin{aligned}
&\exp\Big\{ -C[y_{t+1} - h_{t+1}(u) - \epsilon]\Big\}\mathcal{N}(u|0,1) \\
&= \exp\Big\{ -C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} - \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} - \epsilon\right)\Big\}\mathcal{N}\Big(u\,\Big|\, C\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}},\, 1\Big).
\end{aligned} \tag{4.132}$$

Consequently, $\mathfrak{I}_{t+1}^{-}$ and $\mathfrak{I}_{t+1}^{+}$ boil down to

$$\begin{aligned}
\mathfrak{I}_{t+1}^{-} &= \frac{1-\pi}{2}C\exp\Big\{C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} + \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} + \epsilon\right)\Big\} \\
&\quad \times \int_{(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1} + \epsilon)/\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}^{\infty} \mathcal{N}\Big(u\,\Big|\, -C\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}},\, 1\Big)\mathrm{d}u \\
&= \frac{1-\pi}{4}C\exp\Big\{C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} + \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} + \epsilon\right)\Big\} \\
&\quad \times \left[1 - \mathrm{erf}\left(\frac{y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} + C\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} + \epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right]
\end{aligned} \tag{4.133}$$

and

$$\begin{aligned}
\mathfrak{I}_{t+1}^{+} &= \frac{1-\pi}{2}C\exp\Big\{ -C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} - \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} - \epsilon\right)\Big\} \\
&\quad \times \int_{-\infty}^{(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1} - \epsilon)/\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}} \mathcal{N}\Big(u\,\Big|\, C\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}},\, 1\Big)\mathrm{d}u \\
&= \frac{1-\pi}{4}C\exp\Big\{ -C\left(y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} - \frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} - \epsilon\right)\Big\} \\
&\quad \times \left[1 + \mathrm{erf}\left(\frac{y_{t+1} - \boldsymbol{\mu}_t^{\mathbf{w}} \cdot \mathbf{x}_{t+1} - C\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1} - \epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right],
\end{aligned} \tag{4.134}$$

respectively.

Finally, adding up the final expressions for $\mathfrak{I}_{t+1}^{0}$, $\mathfrak{I}_{t+1}^{-}$ and $\mathfrak{I}_{t+1}^{+}$ that we derived above, we obtain the following closed-form expression for the approximate one-step

lookahead predictive distribution:

$$\begin{aligned}\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi}) = \frac{\pi}{4\epsilon}\left[\operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right) - \operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right] \\ + \frac{1-\pi}{4}C\exp\left\{C\left(y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+\frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}+\epsilon\right)\right\} \\ \times\left[1-\operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}+C\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}+\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right] \\ + \frac{1-\pi}{4}C\exp\left\{-C\left(y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-\frac{C}{2}\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}-\epsilon\right)\right\} \\ \times\left[1+\operatorname{erf}\left(\frac{y_{t+1}-\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}-C\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}-\epsilon}{\sqrt{2\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}}\right)\right].\end{aligned} \tag{4.135}$$

Predictive mean and variance

For prediction purposes, the whole predictive distribution is usually not deployed. Often instead, only the mean and variance of that distribution are retained. The former represents the predicted output value, while the latter captures the uncertainty characterising that prediction. We now derive the mean and the variance of $\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})$.

Using Eq. (4.117) and the integral dimensionality reduction technique from [Barber and Bishop, 1998], the approximate predictive mean is given by

$$\begin{aligned}\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})} &= \int\langle y_{t+1}\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},\mathbf{w},\boldsymbol{\psi})}\,\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t^{\mathbf{w}},\,\boldsymbol{\Sigma}_t^{\mathbf{w}})\,\mathrm{d}\mathbf{w} \\ &= \int\langle y_{t+1}\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},u,\boldsymbol{\psi})}\,\mathcal{N}(u|0,1)\,\mathrm{d}u,\end{aligned} \tag{4.136}$$

where $p(y_{t+1}|\mathbf{x}_{t+1},u,\boldsymbol{\psi}) = \mathcal{Z}\exp\{-C|y_{t+1}-h_{t+1}(u)|_\epsilon\}$, with $|x|_\epsilon = \max\{0,\,|x|-\epsilon\}$. Similarly, the approximate predictive variance is

$$\begin{aligned}&\langle y_{t+1}^2\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})} - \left[\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})}\right]^2 \\ &= \int\langle y_{t+1}^2\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},\mathbf{w},\boldsymbol{\psi})}\,\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t^{\mathbf{w}},\,\boldsymbol{\Sigma}_t^{\mathbf{w}})\,\mathrm{d}\mathbf{w} - \left[\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})}\right]^2 \\ &= \int\langle y_{t+1}^2\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},u,\boldsymbol{\psi})}\,\mathcal{N}(u|0,1)\,\mathrm{d}u - \left[\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\boldsymbol{\psi})}\right]^2.\end{aligned} \tag{4.137}$$

Thus, to obtain the predictive mean and variance, we first need to evaluate the first and second (non-central) moments of the likelihood contribution $p(y_{t+1}|\mathbf{x}_{t+1},u,\boldsymbol{\psi})$, which we do in the paragraph below.

First moments of the likelihood contribution

In order to derive the first and second moments of the likelihood contribution from the $(t+1)$-th example, it is convenient to make use of its uniform-exponential mixture representation given in Eq. (4.122), because we can then apply the results (B.3) and (B.10). This gives

$$\begin{aligned}\langle y_{t+1}\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},u,\psi)} &= \pi\frac{h_{t+1}(u)-\epsilon+h_{t+1}(u)+\epsilon}{2} \\ &\quad+\frac{1-\pi}{2}[-(C^{-1}+\epsilon-h_{t+1}(u))+C^{-1}+\epsilon+h_{t+1}(u)] \\ &= h_{t+1}(u). \end{aligned} \tag{4.138}$$

By a similar reasoning, we obtain the second moment as

$$\begin{aligned}&\langle y_{t+1}^2\rangle_{p(y_{t+1}|\mathbf{x}_{t+1},u,\psi)} \\ &= \pi\frac{[h_{t+1}(u)-\epsilon]^2+[h_{t+1}(u)+\epsilon]^2+[h_{t+1}(u)-\epsilon][h_{t+1}(u)+\epsilon]}{3} \\ &\quad+\frac{1-\pi}{2}\Big[C^{-2}+(C^{-1}+\epsilon-h_{t+1}(u))^2+C^{-2}+(C^{-1}+\epsilon+h_{t+1}(u))^2\Big] \\ &= \pi\frac{2h_{t+1}(u)^2+2\epsilon^2+h_{t+1}(u)^2-\epsilon^2}{3} \\ &\quad+\frac{1-\pi}{2}\Big[2C^{-2}+2(C^{-1}+\epsilon)^2+2h_{t+1}(u)^2\Big] \\ &= h_{t+1}(u)^2+\pi\frac{\epsilon^2}{3}+(1-\pi)\Big[C^{-2}+(C^{-1}+\epsilon)^2\Big]. \end{aligned} \tag{4.139}$$

Plugging the above moments back into Eqs. (4.136) and (4.137), and using the definition of $h_{t+1}(u)$, we find that the approximate predictive mean and variance are

$$\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\psi)} = \int\Big[u\sqrt{\mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}}+\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1}\Big]\mathcal{N}(u|0,1)\,\mathrm{d}u = \boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1} \tag{4.140}$$

and

$$\begin{aligned}&\langle y_{t+1}^2\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\psi)}-\Big[\langle y_{t+1}\rangle_{\widetilde{q}_t(y_{t+1}|\mathbf{x}_{t+1},\psi)}\Big]^2 \\ &= \int h_{t+1}(u)^2\,\mathcal{N}(u|0,1)\,\mathrm{d}u+\pi\frac{\epsilon^2}{3}+(1-\pi)\Big[C^{-2}+(C^{-1}+\epsilon)^2\Big]-(\boldsymbol{\mu}_t^{\mathbf{w}}\cdot\mathbf{x}_{t+1})^2 \\ &= \mathbf{x}_{t+1}^{\mathrm{T}}\boldsymbol{\Sigma}_t^{\mathbf{w}}\mathbf{x}_{t+1}+\pi\frac{\epsilon^2}{3}+(1-\pi)\Big[C^{-2}+(C^{-1}+\epsilon)^2\Big], \end{aligned} \tag{4.141}$$

respectively.

4.3.8 Applications

The contributions in this chapter aim to be methodological with broad applicability. Nonetheless, we decided to include some example applications. These experiments are not intended as exhaustive data analysis but rather to indicate how our online Bayesian passive-aggressive (PA) regression model performs on real problems, specifically against a more standard linear Gaussian state-space model (LGSSM).

Model specification and benchmark

In the following experiments, unless otherwise stated, we used an autoregressive measurement equation of order 1 (AR(1)): $y_t = w_{t,0} + w_{t,1}y_{t1} + u_t$, where $w_{t,0}$ is a bias parameter and u_t is an additive noise term. While this is perhaps not the best specification, feature selection goes beyond the scope of the present study. It is worthwhile noting, however, that there is no theoretical or practical obstacle that would prevent us from considering more complex predictors. This would be expected to further improve the model's performance.

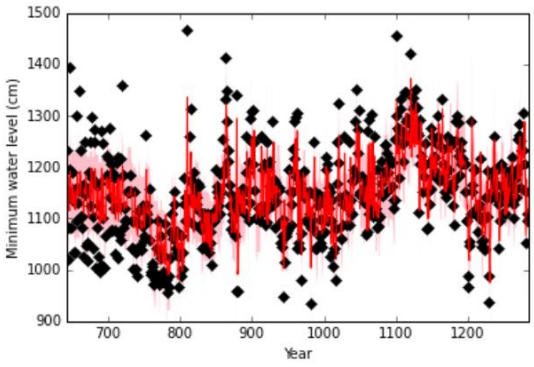
We make comparisons with a standard LGSSM in which a MAP recursion is used to govern the adaptation of the model parameters, by sequentially using the maximum-likelihood formulation first proposed by Jazwinski [1970]. To ensure full comparability of results, we also endow this model with an AR(1) hypothesis, and refer to it as *sequential Kalman filter* (SKF) in the applications below. Also, given its similarity to state-space models, we shall henceforth refer to our Bayesian PA regressor as *adaptive Bayesian passive-aggressive state-space model*, or ADA-BYPASS for short.

In both models, one-step ahead forecasts are successively iterated to provide multi-step forecasts of arbitrary length, as needed. Missing values, if they occur, are accommodated for using the scheme advocated by Shumway and Stoffer [2011], in which they are replaced by their expectations under the corresponding model.

Nile data

We first consider a canonical changepoint data set, the minimum water levels of the Nile river during the period AD 622-1284 [Whitcher et al., 2002]. Several authors have found evidence supporting a changepoint for these data around AD 720-722 [Whitcher et al., 2002; Garnett et al., 2010; Ray and Tsay, 2002]. The conjectured reason for this changepoint is the construction in AD 715 of a new device (a 'nilometer') on the island of Roda, which affected the nature and accuracy of the measurements.

We performed one-year lookahead prediction on this data set. The results can be seen in Figure 4.1. We note the superior performance of ADA-BYPASS compared with the SKF.



Metric	ADA-BYPASS	SKF
RMSE (cm)	**0.72**	0.85
MAD (cm)	**0.42**	0.45
MAE (cm)	**0.54**	0.62
LL	**−754.2**	−971.78

Figure 4.1: Online one-year ahead predictions for the Nile's minimum water levels. Left panel: observed levels (black diamonds), predicted levels (red line) and ±1 standard deviation error bars (pink area). Right panel: predictive performances; error metrics shown are root mean squared error (RMSE), mean absolute deviation (MAD), mean absolute error (MAE) and predictive log likelihood (LL).

Wind speed data

To demonstrate the superior performance of ADA-BYPASS on a large data set, we next present the series of anemometer wind speed measurements (in m/s) from a Danish wind turbine. The data were sampled at ten-minute intervals for just over nine months, resulting in a total of 40,174 measurements. The ten-minute lookahead predictive performance achieved by each method is reported in Table 4.1.

Table 4.1: Predictive performance of ADA-BYPASS vs SKF on the wind speed data set.

Metric	ADA-BYPASS	SKF
RMSE (m/s)	**0.6**	0.64
MAD (m/s)	**0.3**	0.31
MAE (m/s)	**0.42**	0.44
LL	**-24,971.75**	-30,140.04

Statistical arbitrage

LGSSMs, and variants thereof, have seen a widespread use in statistical arbitrage strategies, notably in pairs trading [Chan, 2013; Triantafyllopoulos and Montana, 2011; Montana et al., 2009]. In this area, they serve as a dynamic model for the price spread between two assets. In our application, we seek to find the *hedge ratio*[9] and the predictive standard deviation of the spread. The observable variable is thus one of the price series y, and the hidden variable is the hedge ratio w. We assume that both variables can be modelled via the following measurement and transition equations, respectively:

$$y_t = w_t x_t + u_t, \qquad u_t \sim \mathcal{N}(u_t|\mu, \beta^{-1}), \tag{4.142}$$

$$w_t = w_{t-1} + \xi_t, \qquad \xi_t \sim \mathcal{N}(\xi_t|0, \alpha^{-1}), \tag{4.143}$$

where x is the price series of the other asset. Typically, α, β and μ are manually selected in hindsight [Chan, 2013]. However, this practice is highly prone to the so-called *data-snooping bias*: these parameters can be tweaked so as to optimise the backtesting performance of the strategy. The ADA-BYPASS algorithm automatically tunes its underlying parameters, so it does not suffer from this caveat.

We tested ADA-BYPASS on a pair of exchange-traded funds (ETFs) consisting of the SPDR gold trust GLD and the gold-miners ETF GDX. This ETF pairing is a favourite in the financial industry, because the value of gold-mining companies is very much based on the value of gold. We downloaded the corresponding, daily adjusted closing prices from Yahoo! Finance, between May 22, 2006 and April 22, 2015.

[9]The hedge ratio of a particular asset is the number of units of that asset we should buy or sell in a portfolio. If the asset is a stock, then the number of units corresponds to the number of shares. A negative hedge ratio indicates we should sell that asset.

Table 4.2: Performance of the GDX-GLD pairs trade under ADA-BYPASS and SKF.

Metric	ADA-BYPASS	SKF
Sharpe ratio	**1.00**	0.70
Maximum drawdown (%)	**15.93**	73.05
Maximum drawdown duration (trading days)	**404**	567

Rather than maximising profits, most investors attempt to maximise risk-adjusted returns, as advocated by modern portfolio theory. The Sharpe ratio is the most widely used measure of risk-adjusted returns [Sharpe, 1966]. Besides the Sharpe ratio, the maximum drawdown and maximum drawdown duration are two other popular metrics to evaluate trading strategies. From Table 4.2, we can clearly discern that ADA-BYPASS beats SKF by a significant margin in terms of the aforementioned performance metrics.

4.3.9 Impact of different priors

In order to gauge the effects of our proposed Bayesian approach, we shall in this section check the impact of different means and standard deviations of the prior weight distributions[10]. Since the entire focus and pillar of this thesis are financial applications, we shall just rerun the statistical arbitrage experiment from the previous section under different means for the weight prior, but keeping the variance constant at 0. The relevant metrics are presented in Table 4.3.

Table 4.3: Performance of the GDX-GLD pairs trade under ADA-BYPASS, with different prior means μ (picked arbitrarily) and constant, zero variance.

Metric	$\mu = -10$	$\mu = -2$	$\mu = -1$	$\mu = 0$	$\mu = 1$	$\mu = 2$	$\mu = 10$
Sharpe ratio	1.01	0.98	0.91	1.00	0.94	1.06	1.01
Maximum drawdown (%)	14.72	15.51	15.69	15.93	16.55	16.00	13.74
Maximum drawdown duration (trading days)	379	628	536	404	342	359	319

The table shows that, within reason, the performance of the model is not critical upon the values of the weight prior mean (μ) tested. That is as we may have expected.

[10]We shall not, however, analyse the impact of different classes of prior distributions, as that would move us too far away from the PA regression context which, as we saw from Eqs. (4.27) and (4.28), implicitly imposes a Gaussian prior over its weights.

4.4 Concluding Remarks

In this chapter, we stated the shortcomings of online passive-aggressive (PA) algorithms and discussed ways to overcome them. In particular, we introduced generalised passive-aggressive learning which enables the use of *any* loss function within the PA framework (which, as a reminder, is restricted to the hinge loss for classification and the ϵ-insensitive loss for regression).

We also developed the first online Bayesian PA regression model within the state-space setting, along with a novel, online variational inference algorithm. This model is ideal for the probabilistic prediction of non-stationary and/or very large time series, in particular massive, time-varying data streams. Results on three real-world data sets show significant improvements in predictive performance over a more standard LGSSM.

One of the criticisms that could be directed at our work in this section is the absence of any evidence for why our proposed online hyperparameter tuning mechanism provides benefits, if any, over the most straightforward approach one could think of, i.e. cross-validation (CV). Although it would be possible to perform cross-validation with a rolling or sliding window, one of fundamental ideologies behind the thesis was to have online algorithms for portfolio management with the **fewest** to-be-manually-tuned hyperparameters as possible (so as to avoid data-snooping bias). Going down the rolling-window CV path goes against this fundamental ideology, as it introduces another layer of hyperparameters to be manually chosen by the practitioner (in addition to those she is trying to tune in the first place), namely 1) the size of the rolling window, 2) the size of the CV leave-out sets and 3) the CV frequency (i.e., how often do you perform CV?). For these reasons, we shall not consider rolling CV in this thesis altogether.

5

Adaptive Gradient Methods for Dynamic Online Optimisation

Contents

In the previous chapter, we saw how online regression can be enhanced through the use of Bayesian methods. Another important subfamily of online learning that would significantly benefit from a Bayesian approach is online convex optimisation. In this area, online gradient descent (OGD) is arguably the most popular algorithm.

However, OGD can often be difficult to use for practitioners, because its performance is very sensitive to the choice of learning rate parameter. In this chapter, we shall discuss how a Bayesian treatment of OGD can overcome this limitation, by providing a mechanism to infer the learning rate from the data. The resulting algorithm, named *maximum posterior gradient* (MAPGRAD), also allows us to cheaply compute uncertainty estimates of the learner's actions, based on an approximate posterior distribution over these.

In addition to the above, we shall demonstrate how the generalised passive-aggressive framework developed in the previous chapter can be adapted to the context of bandit convex optimisation. The resulting *passive-aggressive convex optimisation* (PACO) method helps alleviate the difficulties in tuning the learning rate within that context.

5.1 Introduction

Online convex optimisation (OCO) is a fundamental tool for solving a wide variety of machine learning problems, such as online routing, ad selection for search engines and spam filtering [Shalev-Shwartz, 2011; Hazan, 2016]. It can be formulated as a repeated game between a learner and an adversary. On round t of the game, the learner selects an action, i.e. a point $\mathbf{w}_t$ from a convex set Ω, while the adversary chooses a convex loss function $f_t : \Omega \to \mathbb{R}$. Subsequently, the learner incurs a loss of $f_t(\mathbf{w}_t)$ and is provided with feedback $\phi_t(\mathbf{w}_t, f_t)$ about the loss function $f_t(\cdot)$ selected by the adversary. In this work, we shall consider the following three canonical feedback structures:

1. *full-information feedback*, in which the entire function $f_t(\cdot)$ is revealed to the learner after her selection of $\mathbf{w}_t$;

2. *gradient feedback*, meaning the learner only receives gradient information $\mathbf{g}_t \equiv \nabla f_t(\mathbf{w}_t)$, i.e. the gradient of the loss function $f_t(\cdot)$ evaluated at the weight vector $\mathbf{w}_t$;

3. *bandit feedback*, characterised by the revelation of the incurred loss $f_t(\mathbf{w}_t)$ only.

An outline of the OCO problem under these feedback scenarios is given in Algorithm 6.

Algorithm 6 Online Convex Optimisation

1: **Input:** a convex set Ω
2: **for** $t = 1, 2, \ldots$ **do**
3: the learner chooses a vector $\mathbf{w}_t \in \Omega$
4: the adversary selects a loss function $f_t : \Omega \to \mathbb{R}$
5: the learner incurs a loss of $f_t(\mathbf{w}_t)$ and observes

- the loss function $f_t(\cdot)$ in the full-information setting
- the gradient $\mathbf{g}_t \equiv \nabla f_t(\mathbf{w}_t)$ in the gradient-feedback setting
- the loss value $f_t(\mathbf{w}_t)$ in the bandit-feedback setting

6: **end for**

The standard performance measure for OCO problems is the so-called *regret*, which is defined as the difference between the learner's cumulative loss and that of a relatively weak benchmark, namely the single best action in hindsight, or a *static clairvoyant*:

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) \; - \; \min_{\mathbf{w}\in\Omega} \sum_{t=1}^{T} f_t(\mathbf{w}). \tag{5.1}$$

Over the past decades, various online algorithms have been proposed to yield sublinear regret under a variety of feedback structures, with a focus on a class of either convex or strongly convex loss functions. The original work of Zinkevich [2003a] considered the class of convex functions and focused on the case of full-information feedback, providing an *online gradient descent* (OGD) algorithm with regret of order $\sqrt{T}$. Hazan et al. [2007] achieve regret of order $\log T$ for a class of strongly convex loss functions when the gradient of $f_t(\cdot)$, evaluated at $\mathbf{w}_t$, is observed. Additional algorithms were shown to be rate-optimal under further assumptions on the function class (see, e.g., [Kalai and Vempala, 2005; Hazan et al., 2007]), or other feedback structures such as multipoint access [Agarwal et al., 2010].

Though equipped with rich theories, the notion of regret fails to illustrate the performance of online algorithms in a dynamic setting, as a *static* comparator is used in Eq. (5.1). To overcome this limitation, there has been a recent surge of interest in analysing a more stringent metric, namely *dynamic regret* [Hall and Willett, 2013, 2015; Jadbabaie et al., 2015; Besbes et al., 2015; Mokhtari et al., 2016; Yang et al., 2016; Zhang et al., 2017; Gao et al., 2018], in which the cumulative loss of the learner is compared against that attained by a sequence $\mathbf{w}^*_{1:T} \equiv \mathbf{w}^*_1, \mathbf{w}^*_2, \ldots, \mathbf{w}^*_T$ of instantaneous-loss minimisers, i.e.

$$\mathbf{Reg}^d_T \equiv \sum_{t=1}^{T} f_t(\mathbf{w}_t) \; - \; \sum_{t=1}^{T} f_t(\mathbf{w}^*_t) = \sum_{t=1}^{T} f_t(\mathbf{w}_t) \; - \; \sum_{t=1}^{T} \min_{\mathbf{w}\in\Omega} \; f_t(\mathbf{w}), \tag{5.2}$$

where $\mathbf{w}^*_t \in \arg\min_{\mathbf{w}\in\Omega} \; f_t(\mathbf{w})$. In other words, Eq. (5.2) measures the quality of an algorithm, and the sequence $\mathbf{w}_{1:T}$ of actions it generates, by comparing its performance to a *dynamic clairvoyant* who knows the sequence of loss functions in advance, and hence selects the minimiser $\mathbf{w}^*_t$ at each step.

Compared to traditional regret in Eq. (5.1) (also termed *static regret*), dynamic regret is more aggressive in the sense that

$$\min_{\mathbf{w}\in\Omega} \sum_{t=1}^{T} f_t(\mathbf{w}) \; \geq \; \sum_{t=1}^{T} \min_{\mathbf{w}\in\Omega} \; f_t(\mathbf{w}), \tag{5.3}$$

which is a consequence of the fact that the maximum of a sum is at most the sum of maxima (see, e.g., `https://math.stackexchange.com/questions/740074/maximum-of-sum-is-at-m`
As the next example from [Besbes et al., 2015] shows, the dynamic clairvoyant used as benchmark in Eq. (5.2) can be a significantly harder target than the single best action defining the static clairvoyant in Eq. (5.1).

Example 5.1.1. *Assume an action set* $\Omega = [-1, 2]$. *Set*

$$f_t(w) = \begin{cases} w^2 & \text{if } t \leq T/2 \\ w^2 - 2w & \text{otherwise} \end{cases} \tag{5.4}$$

for any $w \in \Omega$. *Then, the single best action is suboptimal at each step, and*

$$\min_{w\in\Omega} \sum_{t=1}^{T} f_t(w) - \sum_{t=1}^{T} \min_{w\in\Omega} \; f_t(w) = \frac{T}{4}. \tag{5.5}$$

Hence, algorithms that achieve performance that is 'close' to the static clairvoyant in the static OCO setting may perform quite poorly in the dynamic OCO setting (in particular, they may, as the example above suggests, incur linear regret in that setting). These algorithms include, but are in no way limited to, the seminal online gradient descent (OGD) algorithm with a learning-rate schedule of $\eta_t = 1/\sqrt{t}$ [Zinkevich, 2003a]. Therefore, we shall refrain from comparing the impact of the methods introduced in this chapter to that schedule on (dynamic) regret. Nevertheless, for the avid reader, in Section 6.5.3 we compare the empirical performance of Zinkevich's OGD algorithm against that of the proposed methods in this thesis.

As in the case of static OCO — i.e., online convex optimisation under static regret — the OGD method has established itself as the candidate of choice for solving dynamic OCO problems, as demonstrated by its widespread use in the literature [Besbes et al., 2015; Mokhtari et al., 2016; Yang et al., 2016; Zhang et al., 2017; Gao et al., 2018]. However, OGD can often be frustrating to use for practitioners, because its performance is very sensitive to the choice of the learning rate parameter. While previous work in the dynamic OCO literature has proposed various ways to select the learning rate, these are based on theoretical considerations as to which assumptions of smoothness, convexity and temporal variation in the loss-function sequence are required to minimise the corresponding regret bounds, in some cases even causing the OGD algorithm to become unfeasible without the benefit of hindsight.

To overcome this practical hurdle, we develop in this chapter two novel adaptation techniques for tuning the learning rate parameter in an *online* and *data-dependent* fashion. Specifically, we propose the following algorithms:

- *Maximum Posterior Gradient* (MAPGRAD), in which we interpret OGD as a Bayesian hierarchical model, treat the learning rate as a *nuisance parameter* and marginalise it, and set the weight vector equal to its *maximum a posteriori* (MAP) value under the resulting marginal weight posterior;

- *Passive-Aggressive Convex Optimisation* (PACO), a generalisation of online passive-aggressive (PA) learning [Crammer et al., 2006] to general loss functions and feasibility regions other than $\mathbb{R}^n$, allowing this framework to be applied successfully to a dynamic OCO setting.

Since it requires knowledge of the loss $f_t(\mathbf{w}_t)$ incurred by the learner at each stage t, the PACO algorithm can only be used in the full-information and bandit settings. As for MAPGRAD, it is applicable under both the full-information and gradient feedback structures. Additionally, wherever needed, we shall assume that the loss functions are differentiable.

To the best of our knowledge, this is pioneering work as far as the dynamic OCO literature is concerned. Granted, beyond this literature, there has been a rich line of work on adaptive learning rate methods over the past few years[11] and, in the next section, we briefly review those that are somehow related to our work. However, we are not aware of any techniques that have approached the automatic tuning of the learning rate from a Bayesian angle, making our MAPGRAD algorithm one of a kind.

5.2 Related Work

The closest papers to our work, at least in spirit, are [Kessler et al., 2020], [Akyildiz et al., 2018] and [Blondel et al., 2014]. Below we discuss the similarities, and differences, between their work and ours. Other prominent, albeit unrelated, adaptive learning rate methods include ADAGRAD [Duchi et al., 2011] and exponential-moving-average variants thereof, namely RMSPROP [Tieleman and Hinton, 2012], ADADELTA [Zeiler, 2012] and ADAM [Kingma and Ba, 2015].

[Kessler et al., 2020] In the spirit of MAPGRAD, Kessler et al. [2020] have recently introduced a framework to cheaply build Bayesian neural networks from adaptive learning rate methods such as ADAGRAD and ADAM. Like ours, this framework is also based on a novel probabilistic interpretation of a generic update

[11] This continues to be an active area of research at the time of writing, especially in the deep learning community.

rule encompassing any adaptive gradient-based scheme, and as such is an elegant generalisation of MAPGRAD. Nevertheless, since the focus of their paper is on obtaining cheap uncertainty estimates for neural-network weights, Kessler et al. [2020] fail to discuss how to leverage their Bayesian approach to infer the learning rate parameter from the data.

[Akyildiz et al., 2018] This is another interesting paper that bears much resemblance to our MAPGRAD framework. The authors highlight a connection between the incremental proximal method (IPM) — to which online gradient descent provides a first-order approximation, as we shall see later — and stochastic filters. This connection is obtained as the result of a probabilistic interpretation of IPM. Interestingly, the authors demonstrate that in the case of linear regression, the probabilistic variant of IPM can be viewed as a Kalman filter. By analogy, for non-linear regressors, there is a direct correspondence between probabilistic IPM and the extended Kalman filter. Note however that, unlike MAPGRAD, the goal of the Bayesian treatment in [Akyildiz et al., 2018] is not to derive an adaptation mechanism for tuning the learning rate parameter.

[Blondel et al., 2014] An adaptive learning rate method that is closely related to our PACO algorithm is NN-PA, which was introduced in [Blondel et al., 2014]. Like PACO, NN-PA is a variant of online passive-aggressive learning that was proposed as a way of overcoming the learning-rate sensitivity of stochastic gradient descent in the particular field of non-negative matrix factorisation. In fact, NN-PA is a special case of PACO in which $\Omega = \mathbb{R}^n_+$ and the loss function is given by the ϵ-insensitive loss function, which is the loss used in the original PA framework of Crammer et al. [2006].

5.3 Problem Formulation

Having already laid out the key building blocks and ideas behind our problem formulation in Section 5.1, the purpose of the present section is to fill in any

gaps and make that exposition more precise where needed. Some repetition is expected but is kept to a minimum.

A dynamic OCO problem consists of a convex set $\Omega \subseteq \mathbb{R}^n$ and an a priori unknown sequence $\{f_t(\cdot)\}_{t\geq 1} \subset \mathcal{F}$, where $\mathcal{F}$ represents a class of sequences of convex loss functions from Ω onto $\mathbb{R}$. At any stage t, the decision maker selects a point $\mathbf{w}_t \in \Omega$, and then observes some feedback ϕ_t. As we mentioned earlier, we shall confine ourselves to the full-information, gradient and bandit feedback structures, in which $\phi_t = f_t$, $\phi_t = \mathbf{g}_t \equiv \nabla f_t(\mathbf{w}_t)$ and $\phi_t = f_t(\mathbf{w}_t)$, respectively, for each step t.

The efficacy of an algorithm — that is, a strategy for choosing a sequence of decisions $\mathbf{w}_t \in \Omega$ — is measured relative to a fictitious clairvoyant who knows the function sequence $\{f_t(\cdot)\}_{t\geq 1}$ in advance, and therefore is able to select, on each round t, a point $\mathbf{w}_t^*$ such that $\mathbf{w}_t^* \in \arg\min_{\mathbf{w}\in\Omega} f_t(\mathbf{w})$. The goal of the learner is to mimic the behaviour of the clairvoyant as closely as possible, that is to say, to find an algorithm whose efficacy is as similar as possible to that of the algorithm the clairvoyant would deploy. This corresponds to playing the strategy with the lowest possible dynamic regret, as defined in Eq. (5.2). Ideally, therefore, the learner would like to play the minimiser $\mathbf{w}_t^*$ of $f_t(\cdot)$ at each time step t, since this strategy would yield zero dynamic regret. This is not possible, however, so a surrogate function must be used in place of $f_t(\cdot)$. Loosely speaking, it would make sense to approximate the unobserved $f_t(\cdot)$ with the most recent loss function $f_{t-1}(\cdot)$ whenever the minimum of the latter is sufficiently close to that of the former:

$$\min_{\mathbf{w}\in\Omega} f_{t-1}(\mathbf{w}) \approx \min_{\mathbf{w}\in\Omega} f_t(\mathbf{w}). \tag{5.6}$$

To ascertain the conditions under which the approximation in Eq. (5.6) is valid, consider the change in the minimal loss from time step $t-1$ to time step t, which is defined by

$$\Delta f_t^* \equiv \min_{\mathbf{w}\in\Omega} f_t(\mathbf{w}) - \min_{\mathbf{w}\in\Omega} f_{t-1}(\mathbf{w}), \qquad t = 2, 3, \ldots, T. \tag{5.7}$$

Using the standard properties of infima and suprema, we can write

$$|\Delta f_t^*| \leq \max_{\mathbf{w}\in\Omega} |f_t(\mathbf{w}) - f_{t-1}(\mathbf{w})|, \qquad t = 2, 3, \ldots, T. \tag{5.8}$$

Summing both sides over t, we obtain

$$\sum_{t=2}^{T} |\Delta f_t^*| \leq \mathrm{Var}(f_1, \ldots, f_T) \equiv \sum_{t=2}^{T} \max_{\mathbf{w}\in\Omega} \; |f_t(\mathbf{w}) - f_{t-1}(\mathbf{w})|. \tag{5.9}$$

Roughly speaking, the variation functional $\mathrm{Var}(\cdot)$ measures the extent to which functions can change from one time step to the next, and adds this up over the horizon T. By virtue of Eq. (5.9), Eq. (5.6) is satisfied when the variation functional is bounded above, that is to say when there exists a scalar V_T such that

$$\mathrm{Var}(f_1, \ldots, f_T) \leq V_T. \tag{5.10}$$

We refer to V_T as the *variation budget* over $\mathcal{F}$. Note that this quantity is allowed to depend on the length of the horizon, and therefore measures the variation scale relative to the latter. In light of all the aforementioned considerations, we shall further constrain the set of admissible loss-function sequences as follows.

Assumption 5.3.1. *The sequence of loss functions $f_{1:T}$ belong to the* temporal uncertainty set $\mathcal{V}$*, defined by*

$$\mathcal{V} \equiv \Big\{ f_{1:T} \subset \mathcal{F} \;\Big|\; \sum_{t=1}^{T-1} \max_{\mathbf{w}\in\Omega} \; |f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})| \leq V_T \Big\}, \tag{5.11}$$

where $\{V_t\}_{t=1,2,\ldots}$ is a non-decreasing sequence of scalars such that $V_t \leq t$ for all t, with $V_1 = 0$ and $V_2 \geq 1$ for normalisation purposes.

It is worthwhile noting that the sequence $\{V_t\}_{t=1,2,\ldots}$ is allowed to grow linearly. While this may seem odd at first, [Besbes et al., 2015, Proposition 1] shows that if the variation budget is *linear* in T, then, as one may expect, it is *impossible* to achieve sublinear dynamic regret. Conversely, if V_T is *sublinear* in T, then algorithms that achieve sublinear dynamic regret *do exist*. With that in mind, hereon, we will focus on the case in which the variation budget is sublinear in T. Even though the variation budget places some restrictions on the possible evolution of loss functions, it still allows for many different temporal patterns, including continuous change, discrete shocks and a non-constant rate of change (see [Besbes et al., 2015] for

examples and illustrations of such variation instances). More importantly for our purposes, these restrictions justify the following action choices:

$$\mathbf{w}_{t+1} \in \arg\min_{\mathbf{w}\in\Omega} \; f_t(\mathbf{w}), \qquad t = 1, 2, \ldots, \tag{5.12}$$

with an arbitrary initial action $\mathbf{w}_1 \in \Omega$.

Before discussing how to implement strategy (5.12) within the different feedback structures considered in this chapter, it is worth emphasising the implications of the temporal uncertainty set (5.11) for dynamic regret.

5.3.1 Temporal variation and dynamic regret

In addition to allowing the learner to mimic the clairvoyant strategy in a fairly accurate way, the temporal uncertainty set ensures the existence of algorithms with sublinear dynamic regret rates. This is the primary reason why it was introduced in [Besbes et al., 2015].

Indeed, in the absence of restrictions on the variation in the adversary's choices, it would not be possible to achieve sublinear dynamic regret, because drastic fluctuations would render the problem intractable. For instance, Besbes et al. [2015] proved that if there is no restriction on the variation of loss functions, then the dynamic regret is linear in T, regardless of the strategy adopted by the learner. Let us illustrate this with a contrived example.

Example 5.3.1. *Suppose that for $t \in [T]$, after the learner has picked an action $w_t \in \mathbb{R}$, the adversary randomly chooses the loss function among $f_t(w) = (w-1)^2$ and $f_t(w) = (w+1)^2$. The expected instantaneous regret for round t in this example satisfies*

$$\begin{aligned}\mathbb{E}[f_t(w_t) - f_t(w_t^*)] &= \frac{1}{2}\left[(w_t-1)^2 - \min_w (w-1)^2\right] + \frac{1}{2}\left[(w_t+1)^2 - \min_w (w+1)^2\right] \\ &= \frac{1}{2}(w_t-1)^2 + \frac{1}{2}(w_t+1)^2 = w_t^2 + 1 \geq 1,\end{aligned} \tag{5.13}$$

whence $\mathbb{E}(\mathbf{Reg}_T^d) = \sum_{t=1}^T (w_t^2+1) \geq T$.

5.4 Full Information

In this section, we assume that on each round, the entire loss function is available to the learner after she makes her decision. Specifically, at each stage t, the sequence of events is as follows:

1. the learner submits an action $\mathbf{w}_t \in \Omega$;
2. the adversary chooses a loss function $f_t(\cdot)$ and reveals it to the learner;
3. the learner incurs a (known) loss of $f_t(\mathbf{w}_t)$.

As discussed in the previous section, the optimal strategy for the learner is to start with an arbitrary choice $\mathbf{w}_1 \in \Omega$, followed by choices $\mathbf{w}_{t+1}$ $(t \geq 1)$ such that

$$\mathbf{w}_{t+1} \in \arg\min_{\mathbf{w}\in\Omega} f_t(\mathbf{w}). \tag{5.14}$$

In the best case scenario, we are able to analytically solve the optimisation problem in Eq. (5.14). In this case, it is relatively straightforward to show that the dynamic regret associated with the action choices in Eq. (5.14) has an upper bound on the order of the variation in the sequence of loss functions. This is formally stated in Proposition 5.4.1.

Proposition 5.4.1. *Under the conditions set out in Assumption 5.3.1, with the exception that $V_t < t$ (rather than $V_t \leq t$) for all t, the action sequence*

$$\mathbf{w}_{t+1} \in \arg\min_{\mathbf{w}\in\Omega} f_t(\mathbf{w}), \qquad t \in [T-1], \tag{5.15}$$

with $\mathbf{w}_1$ being an arbitrary element of Ω, achieves the following dynamic regret bound:

$$\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}_t^*) \leq [f_1(\mathbf{w}_1) - f_1(\mathbf{w}_1^*)] + 2V_T. \tag{5.16}$$

Proof. For each $t \in [T-1]$ and each $\mathbf{w}_t^* \in \arg\min_{\mathbf{w}\in\Omega} f_t(\mathbf{w})$, we can write

$$\begin{aligned} f_{t+1}(\mathbf{w}_{t+1}) - f_{t+1}(\mathbf{w}_{t+1}^*) &= f_{t+1}(\mathbf{w}_t^*) - f_{t+1}(\mathbf{w}_{t+1}^*) \\ &= [f_{t+1}(\mathbf{w}_t^*) - f_t(\mathbf{w}_t^*)] + [f_t(\mathbf{w}_t^*) - f_{t+1}(\mathbf{w}_{t+1}^*)] \\ &\leq |f_{t+1}(\mathbf{w}_t^*) - f_t(\mathbf{w}_t^*)| + |\Delta f_{t+1}^*| \\ &\leq |f_{t+1}(\mathbf{w}_t^*) - f_t(\mathbf{w}_t^*)| + \max_{\mathbf{w}\in\Omega} |f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})|, \end{aligned} \tag{5.17}$$

where we have used the definition (5.7) in the first inequality, while the second inequality is a direct consequence of property (5.8).

By definition, the following inequality holds for any $\mathbf{w} \in \Omega$:

$$|f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})| \leq \max_{\mathbf{w}\in\Omega} |f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})|. \tag{5.18}$$

Setting $\mathbf{w} = \mathbf{w}_t^*$ and plugging the resulting inequality into Eq. (5.17) yields

$$f_{t+1}(\mathbf{w}_{t+1}) - f_{t+1}(\mathbf{w}_{t+1}^*) \leq 2 \max_{\mathbf{w}\in\Omega} |f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})|. \tag{5.19}$$

Summing both sides over t and using Assumption 5.3.1, we obtain

$$\sum_{t=1}^{T-1} [f_{t+1}(\mathbf{w}_{t+1}) - f_{t+1}(\mathbf{w}_{t+1}^*)] \leq 2 \sum_{t=1}^{T-1} \max_{\mathbf{w}\in\Omega} |f_{t+1}(\mathbf{w}) - f_t(\mathbf{w})| \leq 2V_T. \tag{5.20}$$

It immediately follows that

$$\begin{aligned} \sum_{t=1}^{T} [f_t(\mathbf{w}_t) - f_t(\mathbf{w}_t^*)] &= [f_1(\mathbf{w}_1) - f_1(\mathbf{w}_1^*)] + \sum_{t=1}^{T-1} [f_{t+1}(\mathbf{w}_{t+1}) - f_{t+1}(\mathbf{w}_{t+1}^*)] \\ &\leq [f_1(\mathbf{w}_1) - f_1(\mathbf{w}_1^*)] + 2V_T, \end{aligned} \tag{5.21}$$

which completes the proof. □

In most cases, however, the optima in Eq. (5.14) cannot be found by algebraic means alone, and numerical techniques are required. While there are numerous such techniques that proceed iteratively towards the optimum, including line-search, trust-region, conjugate-gradients and quasi-Newton methods, we shall only discuss gradient descent here, as it forms the basis for all of the algorithms that we shall develop in this chapter. For a detailed discussion of alternative methods, we refer the reader to [Nocedal and Wright, 2006].

5.4.1 Gradient descent

Gradient descent is the simplest and oldest of optimisation methods. It is an *iterative* procedure in the sense that it proceeds in iterations, each seeking to decrease the value of the objective function. The method is based on the following principle: locally, if we are at a point $\mathbf{v}_t^{(k)} \in \Omega$, we can decrease $f_t(\cdot)$ by taking a

step in the direction $-\mathbf{g}_t^{(k)}$, where $\mathbf{g}_t^{(k)}$ is the gradient of $f_t(\cdot)$ evaluated at $\mathbf{v}_t^{(k)}$, i.e. $\mathbf{g}_t^{(k)} \equiv \nabla f_t(\mathbf{v}_t^{(k)})$. Mathematically, this idea is expressed by means of the update rule

$$\mathbf{v}_t^{(k+1)} = \Pi_\Omega(\mathbf{v}_t^{(k)} - \eta \mathbf{g}_t^{(k)}), \qquad k \in [K], \tag{5.22}$$

for some fixed learning rate $\eta > 0$ and an integer $K \geq 1$ representing the number of iterations until convergence is achieved.

To see why gradient descent works, assume $\Omega = \mathbb{R}^n$ and apply Taylor's theorem to expand $f_t(\mathbf{v}_t^{(k+1)})$ around $\mathbf{v}_t^{(k)}$, which for small η yields

$$f_t(\mathbf{v}_t^{(k+1)}) = f_t(\mathbf{v}_t^{(k)} - \eta \mathbf{g}_t^{(k)}) \approx f_t(\mathbf{v}_t^{(k)}) - \underbrace{\eta \|\mathbf{g}_t^{(k)}\|_2^2}_{>0} < f_t(\mathbf{v}_t^{(k)}), \tag{5.23}$$

meaning the value of $f_t(\cdot)$ has decreased by moving from the old iterate $\mathbf{v}_t^{(k)}$ to the new one $\mathbf{v}_t^{(k+1)}$. If η is non-infinitesimal, it is always possible that we will overshoot the true minimum. Making η very small guards against this, but means that the optimisation process will take a very long time to reach a minimum.

The gradient descent algorithm adapted to the problem of dynamic OCO in a full-information setting is presented in Algorithm 7. It is worthwhile noting that the same algorithm has been independently proposed by Zhang et al. [2017], under the name 'online multiple gradient descent'. We refer the interested reader to the detailed regret analysis provided therein.

Algorithm 7 Gradient Descent for Dynamic Online Convex Optimisation

Input: convex action space $\Omega \subseteq \mathbb{R}^n$, learning rate $\eta > 0$, tolerance $\epsilon > 0$
Initialisation: arbitrary initial action $\mathbf{w}_1 \in \Omega$, $k = 1$
for $t = 1, 2, \ldots$ **do**
  play $\mathbf{w}_t$ and receive loss function $f_t : \Omega \to \mathbb{R}$
  set $\mathbf{v}_t^{(1)} = \mathbf{w}_t$
  repeat

$$\mathbf{v}_t^{(k+1)} = \Pi_\Omega(\mathbf{v}_t^{(k)} - \eta \mathbf{g}_t^{(k)}), \qquad \mathbf{g}_t^{(k)} \equiv \nabla f_t(\mathbf{v}_t^{(k)})$$
$$k = k + 1$$

  until $|f_t(\mathbf{v}_t^{(k+1)})| \leq \epsilon$
  obtain next action as $\mathbf{w}_{t+1} = \mathbf{v}_t^{(K)}$
end for

Since the inner update rule in Algorithm 7 has the form of *offline* gradient descent, and in this chapter we only care about *online* gradient descent, we will refrain from discussing Bayesian inference-based tools to automatically adjust the learning rate in Algorithm 7. In practice, one uses iteration-dependent learning rates η_k, chosen via line-search methods [Nocedal and Wright, 2006].

5.5 Gradient Feedback

Full information may not be available. In general, only some partial information about the loss functions $f_t(\cdot)$ is available. In this section, we assume that the gradient value $\mathbf{g}_t \equiv \nabla f_t(\mathbf{w}_t)$ is the only feedback that the adversary provides to the learner at each stage t after the latter has submitted her choice. Because we cannot query the gradient of the loss function $f_t(\cdot)$ as many times as we would wish or need to, Algorithm 7 is not applicable in this context.

Previous work on dynamic OCO under gradient feedback has popularised the OGD method, an online variant of standard gradient descent introduced in [Cesa-Bianchi et al., 1996] for prediction problems where the loss functions are convex Bregman divergences, and generalised in [Zinkevich, 2003a] to arbitrary convex functions and problems that may or may not involve prediction. As in the case of offline gradient descent, the learning rate has a significant impact on the performance of OGD, and hence tuning it has proved to be an arduous task. So far in the dynamic OCO literature, theoretical learning rate schedules have been applied, motivated by the assumption that the variation budget V_T and/or certain parameters of the loss functions, such as smoothness or strong convexity constants, are known.

However, in practice, such parameters are unknown, causing existing dynamic OCO methods to be impracticable. It is therefore important to equip the OGD algorithm with a tuning mechanism that operates in an *online* and *data-driven* manner, so that no prior knowledge about V_T or the smoothness/convexity constants of the loss functions is required. To this end, we provide in this section a Bayesian treatment of OGD allowing us to infer the optimal value of the learning rate from observed data, in this case the sequence $\mathbf{g}_{1:t}$ of gradients. The resulting algorithm,

which we shall refer to as *maximum posterior gradient*, or MAPGRAD for short, prescribes a sequence $\eta_{1:t}$ of learning rates such that η_t is the estimate of the learning rate associated with the maximum a posteriori estimate of the weights under their true posterior distribution at time step t.

In order to develop a Bayesian treatment of the OGD method, we first need to interpret it from a probabilistic perspective, which in turn requires understanding which criterion it implicitly optimises. This is the purpose of the next subsection.

5.5.1 The objective function in online gradient descent

Recall that under Assumption 5.3.1, the goal of the learner for round $(t+1)$ in a dynamic OCO scenario is to pick an action $\mathbf{w}_{t+1}$ such that

$$\mathbf{w}_{t+1} \in \operatorname*{arg\,min}_{\mathbf{w}\in\Omega} \; f_t(\mathbf{w}). \tag{5.24}$$

Since in the feedback structure considered here the learner does not observe the loss function $f_t(\cdot)$, she needs to consider a surrogate loss function which we denote by $\widehat{f}_t(\cdot)$. In Proposition 5.5.1, we show that the surrogate loss

$$\widehat{f}_t(\mathbf{w}) \equiv f_t(\mathbf{w}_t) + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) + \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \tag{5.25}$$

induces the OGD update

$$\mathbf{w}_{t+1} = \Pi_\Omega(\mathbf{w}_t - \eta\mathbf{g}_t), \qquad t = 1, 2, \ldots, \tag{5.26}$$

with $\mathbf{w}_1$ being arbitrarily chosen from Ω, and where $\eta > 0$ is the learning rate and Π_Ω denotes the Euclidean projection[12] onto the nearest point in the set Ω, i.e.

$$\Pi_\Omega(\mathbf{v}) \equiv \operatorname*{arg\,min}_{\mathbf{w}\in\Omega} \; \|\mathbf{w} - \mathbf{v}\|_2. \tag{5.27}$$

This means that the OGD update can be interpreted as minimising a first-order Taylor approximation of the loss function $f_t(\cdot)$ around the current action $\mathbf{w}_t$, added to a proximal term $\|\mathbf{w} - \mathbf{w}_t\|_2^2/(2\eta)$.

[12]We defer the discussion of Euclidean projections to Section A.2.2, wherein we also cover how to evaluate them for a relatively wide variety of sets.

Proposition 5.5.1. *Consider the update in Eq.* (5.26)*. Given the iterate* $\mathbf{w}_t$*, the instantaneous gradient* $\mathbf{g}_t = \nabla f_t(\mathbf{w}_t)$ *and the positive constant* η*, the proximal operator*[13]

$$\widetilde{\mathbf{w}}_{t+1} \equiv \arg\min_{\mathbf{w}\in\Omega} \left\{ f_t(\mathbf{w}_t) + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) + \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\} \tag{5.28}$$

is equivalent to the iterate $\mathbf{w}_{t+1}$ *generated by Eq.* (5.26)*, for each integer* $t \geq 1$*.*

Proof. We prove the above statement by contradiction. Suppose that $\widetilde{\mathbf{w}}_{t+1} \neq \mathbf{w}_{t+1}$. Combining Eq. (5.26) and Definition (5.27), we obtain

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\Omega} \|\mathbf{w} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2. \tag{5.29}$$

Thus, by construction, $\mathbf{w}_{t+1}$ is the smallest such $\mathbf{w}$ in Ω or, in mathematical terms,

$$\|\mathbf{w}_{t+1} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2 \leq \|\mathbf{x} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2, \qquad \forall\, \mathbf{x} \in \Omega. \tag{5.30}$$

Since Ω is convex, the projection $\Pi_\Omega(\mathbf{w}_t - \eta\mathbf{g}_t)$ has a unique solution, which is $\mathbf{w}_{t+1}$. Therefore, by setting $\mathbf{x} = \widetilde{\mathbf{w}}_{t+1}$ in Eq. (5.30), the inequality becomes strict, i.e.

$$\|\mathbf{w}_{t+1} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2 < \|\widetilde{\mathbf{w}}_{t+1} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2. \tag{5.31}$$

Taking the square of both sides, cancelling out common terms and then dividing both sides by 2η yields

$$\frac{1}{2\eta}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\mathbf{w}_{t+1} - \mathbf{w}_t) < \frac{1}{2\eta}\|\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t). \tag{5.32}$$

On the other hand, the definition of $\widetilde{\mathbf{w}}_{t+1}$ in Eq. (5.28) implies that

$$\frac{1}{2\eta}\|\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t) \leq \frac{1}{2\eta}\|\mathbf{x} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\mathbf{x} - \mathbf{w}_t), \quad \forall\, \mathbf{x} \in \Omega, \tag{5.33}$$

where we have omitted the additive term $f_t(\mathbf{w}_t)$ because it appears on both sides of the inequality. Setting $\mathbf{x} = \mathbf{w}_{t+1}$ in Eq. (5.33), we obtain

$$\frac{1}{2\eta}\|\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\widetilde{\mathbf{w}}_{t+1} - \mathbf{w}_t) < \frac{1}{2\eta}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|_2^2 + \mathbf{g}_t \cdot (\mathbf{w}_{t+1} - \mathbf{w}_t), \tag{5.34}$$

which clearly contradicts Eq. (5.32).

Consequently, our initial premise that $\widetilde{\mathbf{w}}_{t+1} \neq \mathbf{w}_{t+1}$ cannot be true, which means that the OGD update in Eq. (5.26) coincides with that from Eq. (5.28). □

[13] We refer the reader to [Parikh and Boyd, 2013] for further details on proximal operators.

In what follows, we give careful consideration to the choice of the learning rate η from a Bayesian standpoint. To this end, we first present a novel probabilistic interpretation of OGD based on its implicit criterion, i.e. the optimisation problem in Eq. (5.28). An additional advantage of this interpretation is that it can be used to cheaply compute uncertainty estimates of the learner's actions.

5.5.2 Probabilistic interpretation of online gradient descent

We saw in Proposition 5.5.1 that OGD implicitly solves the problem

$$\min_{\mathbf{w}\in\Omega} \left\{ f_t(\mathbf{w}_t) + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) + \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \right\}, \qquad t = 1, 2, \ldots \tag{5.35}$$

This problem can be equivalently formulated as

$$\begin{aligned}
&\min_{\mathbf{w}\in\mathbb{R}^n} \left\{ f_t(\mathbf{w}_t) + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) + \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 + \iota_\Omega(\mathbf{w}) \right\} \\
&= \max_{\mathbf{w}\in\mathbb{R}^n} \left\{ -f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \iota_\Omega(\mathbf{w}) \right\} \\
&= \max_{\mathbf{w}\in\mathbb{R}^n} \log \left[\exp \left\{ -f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \iota_\Omega(\mathbf{w}) \right\} \right],
\end{aligned} \tag{5.36}$$

where $\iota_{\mathcal{C}}(x)$ denotes the set indicator function, taking the value 0 if $x \in \mathcal{C}$ and $+\infty$ otherwise. This formulation highlights the fact that the OGD update in Eq. (5.26) coincides with the *maximum a posteriori* (MAP) estimate of $\mathbf{w}$ under the posterior distribution $p(\mathbf{w}|\mathcal{D}_t, \mathbf{w}_t, \eta)$ defined by

$$p(\mathbf{w}|\mathcal{D}_t, \mathbf{w}_t, \eta) \propto \exp \left\{ -f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \iota_\Omega(\mathbf{w}) \right\}, \tag{5.37}$$

where $\mathcal{D}_t$ is the t-th example implicit in the observed gradient value $\mathbf{g}_t$[14]. Simple manipulation involving completing the square with respect to $\mathbf{w}$ in the exponent gives us

$$\begin{aligned}
&- f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t) - \frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \iota_\Omega(\mathbf{w}) \\
&= \left[\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t) \right] - \frac{1}{2\eta}\|\mathbf{w} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2^2 - \iota_\Omega(\mathbf{w}).
\end{aligned} \tag{5.38}$$

[14]For instance, if $f_t(\cdot)$ is the square loss function, i.e. $f_t(\mathbf{w}) = (y_t - \mathbf{w} \cdot \mathbf{x}_t)^2$, where $\mathbf{x}_t$ and y_t are the input vector and output at time t, respectively, then we have $\mathcal{D}_t = (\mathbf{x}_t, y_t)$.

Taking the exponential of the right-hand side, back-substituting it into Eq. (5.37) and normalising the resulting quantity, we obtain

$$p(\mathbf{w}|\mathcal{D}_t, \mathbf{w}_t, \eta) = \frac{\cancel{\exp\left\{\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t)\right\}} \exp\left\{-\frac{1}{2\eta}\|\mathbf{w} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2^2 - \iota_\Omega(\mathbf{w})\right\}}{\cancel{\exp\left\{\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t)\right\}} \int \exp\left\{-\frac{1}{2\eta}\|\mathbf{w} - (\mathbf{w}_t - \eta\mathbf{g}_t)\|_2^2 - \iota_\Omega(\mathbf{w})\right\} \mathrm{d}\mathbf{w}}$$
$$= \frac{\mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})}{\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}}, \tag{5.39}$$

where $\mathbb{1}_\mathcal{C}(x) \equiv \exp\{-\iota_\mathcal{C}(x)\}$ is the characteristic function of the set $\mathcal{C}$. We recognise this as a *truncated Gaussian distribution* over Ω, which we shall denote by

$$p(\mathbf{w}|\mathcal{D}_t, \mathbf{w}_t, \eta) = \mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n) \equiv \frac{\mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})}{\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}}. \tag{5.40}$$

We have thus established that the OGD update (5.26) coincides with the most probable value of $\mathbf{w}$ given the current weights $\mathbf{w}_t$ and datum $\mathcal{D}_t$, in other words the maximiser of the logarithm of Eq. (5.40). To complete the specification of the probabilistic model behind OGD, we need to understand which likelihood function and prior distribution induce the posterior (5.40).

Likelihood and prior

Casting a frequentist into a MAP framework is achieved by taking a Bayesian decision-theoretic approach in which the loss function is based on the log likelihood of the data and the penalty is associated with a prior distribution over the parameters of interest. For example, in the LASSO algorithm [Tibshirani, 1996], the least-squares loss function is associated with a Gaussian likelihood (whose log is equal, up to a constant of proportionality, to the least-squares loss), while there exists duality between the ℓ_1 penalty and the Laplace prior.

Following a similar line of reasoning, we may deduce that the likelihood corresponding to OGD is such that

$$f_t(\mathbf{w}) = -\log p(\mathcal{D}_t|\mathbf{w}) \tag{5.41}$$

or, equivalently,

$$p(\mathcal{D}_t|\mathbf{w}) = \exp\Big\{-f_t(\mathbf{w})\Big\}. \tag{5.42}$$

Technically speaking, this quantity is a *pseudo*-likelihood because it is unnormalised with respect to $\mathcal{D}_t$. We work with this rather than with its normalised counterpart because it leads to the traditional OGD update (i.e. Eq. (5.26)), noting that the same approach has been previously adopted in Bayesian treatments of other frequentist algorithms, such as the support vector machine [Polson and Scott, 2011; Henao et al., 2014].

As for the prior distribution $p(\mathbf{w}|\eta)$, by inspection of Eq. (5.36) after setting $t = 1$, it becomes apparent that

$$p(\mathbf{w}|\eta) \propto \exp\Big\{ -\frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_1\|_2^2 - \iota_\Omega(\mathbf{w})\Big\}, \tag{5.43}$$

which we recognise as the unnormalised density of a truncated Gaussian distribution of the form

$$p(\mathbf{w}|\eta) = \mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_1,\, \eta\mathbf{I}_n). \tag{5.44}$$

Online gradient descent as approximate inference

It is important to note that Eq. (5.40) is an *approximate, parametric* posterior since it is not conditioned on the entire old data set $\mathcal{D}_{1:t}$, as is the case for the true posterior $p(\mathbf{w}|\mathcal{D}_{1:t}, \eta)$. Given the nature of the probabilistic model induced by OGD, this true posterior cannot be evaluated analytically, as we shall demonstrate in a moment. We shall further see that OGD bypasses this intractability by linearising the loss function around the current weight vector, which gives rise to the truncated Gaussian approximation from Eq. (5.40).

In the context of sequential inference, the recursive formulation of Bayes' rule states that the posterior distribution at any stage acts as the prior distribution for the subsequent data point. Using our notation, this is expressed by

$$p(\mathbf{w}|\mathcal{D}_{1:t}, \eta) = \frac{p(\mathcal{D}_t|\mathbf{w})p(\mathbf{w}|\mathcal{D}_{1:t-1}, \eta)}{\int p(\mathcal{D}_t|\mathbf{w})p(\mathbf{w}|\mathcal{D}_{1:t-1}, \eta)\,\mathrm{d}\mathbf{w}}. \tag{5.45}$$

Eq. (5.45) does not have the form of an online algorithm because it requires knowledge of the *entire old data set* $\mathcal{D}_{1:t-1}$. The basic idea, proposed by Opper [1998], to turn this into an online algorithm is to replace the true posterior $p(\mathbf{w}|\mathcal{D}, \eta)$

by a simpler parametric distribution $p(\mathbf{w}|\boldsymbol{\theta}, \eta)$, where $\boldsymbol{\theta}$ is a set of parameters capturing a major part of the information about the previous data and having to be updated at each step. Using the old approximating posterior $p(\mathbf{w}|\boldsymbol{\theta}_t, \eta)$, we can perform an update of the form (5.45) once the new data point $\mathcal{D}_t$ arrives, like so:

$$\widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, \eta) = \frac{p(\mathcal{D}_t|\mathbf{w})p(\mathbf{w}|\boldsymbol{\theta}_t, \eta)}{\int p(\mathcal{D}_t|\mathbf{w})p(\mathbf{w}|\boldsymbol{\theta}_t, \eta)\,\mathrm{d}\mathbf{w}}. \tag{5.46}$$

By inspection of the OGD criterion in Eq. (5.36), we see that the approximating posterior employed by the algorithm takes the form

$$p(\mathbf{w}|\boldsymbol{\theta}_t, \eta) \propto \exp\Big\{ -\frac{1}{2\eta}\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \iota_\Omega(\mathbf{w})\Big\}, \tag{5.47}$$

in other words $\boldsymbol{\theta}_t = \mathbf{w}_t$ and

$$p(\mathbf{w}|\boldsymbol{\theta}_t, \eta) = \mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n). \tag{5.48}$$

From this and Eq. (5.42), it follows that in the case of OGD, we have

$$\widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, \eta) = \frac{\exp\{-f_t(\mathbf{w})\}\mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)}{\int \exp\{-f_t(\mathbf{w})\}\mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\,\mathrm{d}\mathbf{w}}. \tag{5.49}$$

In all but limited special cases, the pseudo-likelihood contribution $\exp\{-f_t(\mathbf{w})\}$ is not a simple squared exponential, causing $\widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, \eta)$ to not belong to the parametric family $p(\mathbf{w}|\boldsymbol{\theta}, \eta)$. To address this issue, OGD linearises the loss function $f_t(\cdot)$ about the current weight estimates $\mathbf{w}_t$ using a first-order Taylor series expansion, i.e.

$$f_t(\mathbf{w}) \approx f_t(\mathbf{w}_t) + \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t), \qquad t = 1, 2, \ldots \tag{5.50}$$

Applying this expansion to the numerator in Eq. (5.49), along with the definition of the truncated Gaussian and the square-completion result from Eq. (5.38), the latter can be approximated as

$$\begin{aligned}
&\exp\Big\{ -f_t(\mathbf{w})\Big\}\mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n) \approx \exp\Big\{ -f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t)\Big\}\mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n) \\
&= \exp\Big\{ -f_t(\mathbf{w}_t) - \mathbf{g}_t \cdot (\mathbf{w} - \mathbf{w}_t)\Big\}\, \frac{\mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})}{\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}} \\
&= \Big[\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}\Big]^{-1} \exp\Big\{\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t)\Big\}\mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w}).
\end{aligned} \tag{5.51}$$

Plugging this result back into Eq. (5.49), we then have

$$\begin{aligned}
&\widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, \eta) \\
&\approx \frac{\cancel{[\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}]^{-1}} \exp\left\{\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t)\right\}\mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})}{\cancel{[\int \mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}]^{-1}} \exp\left\{\frac{\eta}{2}\|\mathbf{g}_t\|_2^2 - f_t(\mathbf{w}_t)\right\} \int \mathcal{N}(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n)\mathbb{1}_\Omega(\mathbf{w})\,\mathrm{d}\mathbf{w}} \\
&= \mathcal{N}_\Omega(\mathbf{w}|\mathbf{w}_t - \eta\mathbf{g}_t,\, \eta\mathbf{I}_n) \\
&= p(\mathbf{w}|\mathcal{D}_t, \mathbf{w}_t, \eta), \qquad (5.52)
\end{aligned}$$

and so we recover the approximate posterior (5.40) implicitly maximised by OGD.

5.5.3 Inference of the learning rate

Now that we have seen how OGD can be used as an approximate inference algorithm, we proceed to discussing how to infer the learning rate η (or $\alpha \equiv \eta^{-1}$ for convenience) from the data set. Since we are not interested in the value of this hyperparameter per se, effectively a *nuisance variable*, the proper Bayesian approach is to *marginalise*[15] it in order to perform inference. This justifies the application of the MAP method[16] as described in, for example, [MacKay, 1996]. In this method, the true posterior distribution of the weights is found by marginalising the hyperparameter α. The true posterior is then maximised with respect to the weight parameters. Integrating over a nuisance parameter is very much like estimating this parameter from the data [Bretthorst, 1988; Box and Tiao, 1992].

The MAP method

The MAP method starts by integrating out α to obtain the marginal, or what might be considered the 'true', prior over the weights:

$$p(\mathbf{w}) = \int p(\mathbf{w}|\alpha)p(\alpha)\,\mathrm{d}\alpha. \qquad (5.53)$$

[15]The process of marginalisation refers to 'integrating out' uncertainty. For example, given $p(x, \theta) = p(x|\theta)p(\theta)$, we may obtain $p(x)$ by marginalising over the unknown parameter θ, such that $p(x) = \int p(x|\theta)p(\theta)\,\mathrm{d}\theta$.

[16]which stands for *maximum a posteriori*; this use of the term 'MAP' may not coincide precisely with its general usage.

We can then write down the true posterior directly (except for its normalising constant):

$$p(\mathbf{w}|\mathcal{D}_{1:t}) \propto p(\mathcal{D}_{1:t}|\mathbf{w})p(\mathbf{w}). \tag{5.54}$$

This posterior can be maximised to find the (hyperparameter-free) MAP weight vector for round $(t+1)$, which we shall denote by $\mathbf{w}_{t+1}^{\mathrm{MP}}$.

It will prove convenient to show that $\mathbf{w}_{t+1}^{\mathrm{MP}}$ is also a maximum of the conditional posterior $p(\mathbf{w}|\mathcal{D}_{1:t}, \alpha)$, with α set to its expected value $\langle\alpha\rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}$ under the conditional hyperparameter prior[17]

$$p(\alpha|\mathbf{w}) = \frac{p(\mathbf{w}|\alpha)p(\alpha)}{p(\mathbf{w})}. \tag{5.55}$$

We begin by noting that the first-order optimality conditions (FOCs) for maximising $\log p(\mathbf{w}|\mathcal{D}_{1:t})$ with respect to $\mathbf{w}$ (which is tantamount to maximising $p(\mathbf{w}|\mathcal{D}_{1:t})$) require that the gradient of $\log p(\mathbf{w}|\mathcal{D}_{1:t})$, evaluated at the optimal point $\mathbf{w}_{t+1}^{\mathrm{MP}}$, is equal to the zero vector, i.e.

$$\left[\nabla_{\mathbf{w}} \log p(\mathcal{D}_{1:t}|\mathbf{w}) + \nabla_{\mathbf{w}} \log p(\mathbf{w})\right]\Big|_{\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}}} = \mathbf{0}_{n\times 1}. \tag{5.56}$$

Similarly, by virtue of Bayes' rule, the optimum of the conditional log-posterior must satisfy the following FOCs:

$$\nabla_{\mathbf{w}} \log p(\mathcal{D}_{1:t}|\mathbf{w}) + \nabla_{\mathbf{w}} \log p(\mathbf{w}|\alpha) = \mathbf{0}_{n\times 1}. \tag{5.57}$$

A comparison between Eqs. (5.56) and (5.57) reveals that the maxima of the true posterior $p(\mathbf{w}|\mathcal{D}_{1:t})$ will coincide with those of the conditional posterior $p(\mathbf{w}|\mathcal{D}_{1:t}, \alpha)$ if the gradients of their respective log-priors coincide at $\mathbf{w} = \mathbf{w}_{t+1}^{\mathrm{MP}}$, given $\alpha = \langle\alpha\rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}$. In other words, it suffices to prove that

$$\nabla_{\mathbf{w}} \log p(\mathbf{w})\Big|_{\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}}} = \nabla_{\mathbf{w}} \log p(\mathbf{w}|\alpha = \langle\alpha\rangle_{p(\alpha|\mathbf{w})})\Big|_{\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}}}. \tag{5.58}$$

From Eq. (5.44), we know that

$$p(\mathbf{w}|\alpha) = \mathcal{N}_{\Omega}(\mathbf{w}|\mathbf{w}_1, \alpha^{-1}\mathbf{I}_n) = \frac{\mathcal{N}(\mathbf{w}|\mathbf{w}_1, \alpha^{-1}\mathbf{I}_n)\mathbb{1}_{\Omega}(\mathbf{w})}{\int \mathcal{N}(\mathbf{w}|\mathbf{w}_1, \alpha^{-1}\mathbf{I}_n)\mathbb{1}_{\Omega}(\mathbf{w})\,\mathrm{d}\mathbf{w}}. \tag{5.59}$$

[17] As a reminder, $\langle x\rangle_{p(x)}$ denotes the expectation of the random variable x taken with respect to the distribution $p(x)$.

Ignoring all the terms that do not depend on $\mathbf{w}$, this allows us to write

$$\nabla_{\mathbf{w}} \log p(\mathbf{w}|\alpha = \langle\alpha\rangle_{p(\alpha|\mathbf{w})})\Big|_{\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}}} = -\langle\alpha\rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}(\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1) - \nabla\iota_{\Omega}(\mathbf{w}_{t+1}^{\mathrm{MP}}), \tag{5.60}$$

where, as a reminder, $\iota_{\Omega}(\mathbf{w})$ is the set indicator function (which takes the value 0 if $\mathbf{w} \in \Omega$ and $+\infty$ otherwise), and $\nabla\iota_{\Omega}(\mathbf{w}_{t+1}^{\mathrm{MP}})$ denotes a subgradient thereof evaluated at $\mathbf{w}_{t+1}^{\mathrm{MP}}$. It remains to derive $\nabla_{\mathbf{w}} \log p(\mathbf{w})|_{\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}}}$. Combining Eqs. (5.53) and (5.59), and using Leibniz' integral rule, we obtain

$$\begin{aligned}
\nabla_{\mathbf{w}} \log p(\mathbf{w}) &= \frac{1}{p(\mathbf{w})} \int [\nabla_{\mathbf{w}} \mathcal{N}_{\Omega}(\mathbf{w}|\mathbf{w}_1,\, \alpha^{-1}\mathbf{I}_n)] p(\alpha)\, \mathrm{d}\alpha \\
&= \frac{1}{p(\mathbf{w})} \int [-\alpha(\mathbf{w} - \mathbf{w}_1) - \nabla\iota_{\Omega}(\mathbf{w})]\, \mathcal{N}_{\Omega}(\mathbf{w}|\mathbf{w}_1,\, \alpha^{-1}\mathbf{I}_n) p(\alpha)\, \mathrm{d}\alpha \\
&= \int [-\alpha(\mathbf{w} - \mathbf{w}_1) - \nabla\iota_{\Omega}(\mathbf{w})]\, \underbrace{\frac{p(\mathbf{w}|\alpha)p(\alpha)}{p(\mathbf{w})}}_{=p(\alpha|\mathbf{w})}\, \mathrm{d}\alpha \\
&= -\langle\alpha\rangle_{p(\alpha|\mathbf{w})}(\mathbf{w} - \mathbf{w}_1) - \nabla\iota_{\Omega}(\mathbf{w}).
\end{aligned} \tag{5.61}$$

Setting $\mathbf{w} = \mathbf{w}_{t+1}^{\mathrm{MP}}$ and comparing the resulting expression against Eq. (5.60) shows, as asserted, that $\mathbf{w}_{t+1}^{\mathrm{MP}}$ is also a maximum of the conditional posterior $p(\mathbf{w}|\mathcal{D}_{1:t}, \alpha)$ for the particular value of $\alpha = \langle\alpha\rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}$.

Optimal learning rate

To be able to compute the optimal (inverse) learning rate, $\langle\alpha\rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}$, we must specify a hyperprior over α. This quantity is an example of a *scale* parameter, and a suitable prior thereover is a Gamma distribution (see, e.g., [Berger, 1985]):

$$p(\alpha) = \mathcal{G}(\alpha|a, b) \equiv \frac{b^a}{\Gamma(a)} \alpha^{a-1} \exp\Big\{-b\alpha\Big\}, \tag{5.62}$$

where $\Gamma(a) \equiv \int_0^{\infty} x^{a-1} \exp\{-x\}\, \mathrm{d}x$ is the gamma function. Under this hyperprior, the joint prior over $\mathbf{w}$ and α is given by

$$p(\mathbf{w}, \alpha) = p(\mathbf{w}|\alpha)p(\alpha) = \mathcal{N}_{\Omega}(\mathbf{w}|\mathbf{w}_1,\, \alpha^{-1}\mathbf{I}_n)\, \mathcal{G}(\alpha|a, b). \tag{5.63}$$

We can rearrange this equation so that it reads $p(\mathbf{w}, \alpha) = p(\alpha|\mathbf{w})p(\mathbf{w})$. Letting $\mathcal{Z}_{\mathbf{w}}(\alpha)$ be the normalising constant of $p(\mathbf{w}|\alpha)$, this gives

$$\begin{aligned} p(\mathbf{w}, \alpha) &= \mathcal{Z}_{\mathbf{w}}^{-1}(\alpha)(2\pi\alpha^{-1})^{-n/2} \exp\left\{-\frac{\alpha}{2}\|\mathbf{w}-\mathbf{w}_1\|_2^2\right\} \mathbb{1}_\Omega(\mathbf{w}) \frac{b^a}{\Gamma(a)} \alpha^{a-1} \exp\Big\{-b\alpha\Big\} \\ &= \mathcal{Z}_{\mathbf{w}}^{-1}(\alpha)\alpha^{a+n/2-1} \exp\left\{-\left(b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right)\alpha\right\} \frac{b^a}{(2\pi)^{n/2}\Gamma(a)} \mathbb{1}_\Omega(\mathbf{w}) \\ &= \mathcal{Z}_{\mathbf{w}}^{-1}(\alpha) \frac{\left(b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right)^{a+n/2}}{\Gamma(a+n/2)} \alpha^{a+n/2-1} \exp\left\{-\left(b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right)\alpha\right\} \\ &\quad \times \frac{b^a\Gamma(a+n/2)}{(2\pi)^{n/2}\Gamma(a)} \left(b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right)^{-(a+n/2)} \mathbb{1}_\Omega(\mathbf{w}). \end{aligned} \tag{5.64}$$

From this, we infer that

$$p(\alpha|\mathbf{w}) \propto \mathcal{Z}_{\mathbf{w}}^{-1}(\alpha)\, \mathcal{G}\left(\alpha \,\middle|\, a + \frac{n}{2},\ b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right) \tag{5.65}$$

and

$$p(\mathbf{w}) \propto \frac{b^a\Gamma(a+n/2)}{(2\pi)^{n/2}\Gamma(a)} \left(b + \frac{\|\mathbf{w}-\mathbf{w}_1\|_2^2}{2}\right)^{-(a+n/2)} \mathbb{1}_\Omega(\mathbf{w}). \tag{5.66}$$

We are now forced to adopt some form of approximation, and do so by assuming no constraints on $\mathbf{w}$. In reality, there are constraints, namely $\mathbf{w} \in \Omega$, but if we can determine the optimal learning rate for unconstrained $\mathbf{w}$, we can offset this assumption by projecting the resulting unconstrained weights onto the constraint surface Ω, as is done in the OGD update (5.26). For the case of unconstrained weights, we have $\mathcal{Z}_{\mathbf{w}}(\alpha) \equiv 1$, and so based on Eq. (5.65),

$$\langle\alpha\rangle_{p(\alpha|\mathbf{w})} = \frac{a + n/2}{b + \|\mathbf{w}-\mathbf{w}_1\|_2^2/2} = \frac{n + 2a}{\|\mathbf{w}-\mathbf{w}_1\|_2^2 + 2b}, \tag{5.67}$$

which is reminiscent of an expectation-maximisation update.

We would then proceed by repeatedly applying Eq. (5.67), concurrent with the weight updates from Eq. (5.26), until some suitable convergence criteria have been satisfied. Instead, however, we slightly depart from Eq. (5.26) by considering *lazy* OGD updates, which leads to a closed-form solution for $\langle\alpha\rangle_{p(\alpha|\mathbf{w})}$ under a non-informative hyperprior, as we shall demonstrate below. This is of utmost practical importance in an online setting, as the iterative re-estimation of α might take longer to converge than the time interval between the arrival of data points.

Lazy updates

If we pretend that there are no constraints on the weights, which is equivalent to assuming that $\Omega = \mathbb{R}^n$, the OGD update rule (5.26) becomes

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t = \mathbf{w}_1 - \eta \sum_{\tau=1}^{t} \mathbf{g}_\tau, \tag{5.68}$$

where the second equality is obtained by induction. Now, to rectify the incorrectness of our assumption, we perform a projection of the above update rule onto Ω, giving

$$\mathbf{w}_{t+1} = \Pi_\Omega \left(\mathbf{w}_1 - \eta \sum_{\tau=1}^{t} \mathbf{g}_\tau \right). \tag{5.69}$$

This is called a *lazy* projection because we do not project the iterates $\mathbf{w}_t$ onto Ω at each stage (as in Eq. (5.26)), but instead keep track of the running sum of loss gradients and only project it at decision time. The corresponding algorithm is therefore known as *online gradient descent with lazy projections* (see, e.g., [Shalev-Shwartz, 2011] for further details).

As we mentioned earlier, an interesting closed-form expression for the optimal learning rate arises when we consider lazy OGD updates together with a non-informative hyperprior over α. Such a hyperprior is characterised by $a = b = 0$, and so from Eq. (5.67), we have

$$\langle \alpha \rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})} = \frac{n}{\|\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1\|_2^2}. \tag{5.70}$$

Plugging this expression into Eq. (5.68) yields

$$\mathbf{w}_{t+1}^{\mathrm{MP}} = \mathbf{w}_1 - \frac{\|\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1\|_2^2}{n} \sum_{\tau=1}^{t} \mathbf{g}_\tau. \tag{5.71}$$

Subtracting $\mathbf{w}_1$ from both sides and taking the squared Euclidean norm of the resulting expressions, we obtain

$$\|\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1\|_2^2 = \frac{\|\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1\|_2^4}{n^2} \left\| \sum_{\tau=1}^{t} \mathbf{g}_\tau \right\|_2^2 \iff \frac{\|\mathbf{w}_{t+1}^{\mathrm{MP}} - \mathbf{w}_1\|_2^2}{n} = \frac{n}{\left\| \sum_{\tau=1}^{t} \mathbf{g}_\tau \right\|_2^2} \tag{5.72}$$

so the optimal learning rate, in the MAP sense, is given by

$$\eta_t^{\mathrm{MP}} \equiv \langle \alpha \rangle_{p(\alpha|\mathbf{w}=\mathbf{w}_{t+1}^{\mathrm{MP}})}^{-1} = \frac{n}{\left\| \sum_{\tau=1}^{t} \mathbf{g}_\tau \right\|_2^2}. \tag{5.73}$$

It is worthwhile noting that η_t^{MP} bears some resemblance to the learning rate employed by the ADAGRAD algorithm of Duchi et al. [2011], although these authors took a completely different and non-Bayesian approach.

5.5.4 Maximum posterior gradient

The online, approximately Bayesian algorithm we propose for dynamic OCO under gradient feedback, termed MAPGRAD and summarised in Algorithm 8, is a hyperparameter-free, two-step procedure for adapting both the learner's actions $\mathbf{w}$ and the learning rate η, after the presentation of each example.

Algorithm 8 MAPGRAD: Maximum Posterior Gradient

Input: convex and closed action space $\Omega \subseteq \mathbb{R}^n$
Initialisation: arbitrary initial action $\mathbf{w}_1 \in \Omega$, initial gradient sum $\mathbf{s}_0^g = \mathbf{0}_{n\times 1}$
for $t = 1, 2, \ldots$ **do**
  play $\mathbf{w}_t$ and receive the gradient $\mathbf{g}_t \equiv \nabla f_t(\mathbf{w}_t)$
  update the gradient sum:

$$\mathbf{s}_t^g = \mathbf{s}_{t-1}^g + \mathbf{g}_t$$

  compute the learning rate:

$$\eta_t = \frac{n}{\|\mathbf{s}_t^g\|_2^2}$$

  compute the next action:

$$\mathbf{w}_{t+1} = \Pi_\Omega(\mathbf{w}_t - \eta_t \mathbf{g}_t)$$

end for

Note that Algorithm 8 is based on the agile OGD update rule from Eq. (5.26) rather than its lazy counterpart in Eq. (5.69). This is because we relied on Eq. (5.69) only to be able to derive an analytical expression for the optimal learning rate, and thereby bypass the need for iterative re-estimation rules that in practice might take longer to converge than the time interval between consecutive data points.

5.5.5 Improving uncertainty estimates

As well as providing a principled approach to tuning the learning rate in an online and data-driven manner, the MAPGRAD framework can also be used to quantify the uncertainty that surrounds the appropriate choice for the learner's actions $\mathbf{w}$.

By plugging the optimal learning rate (5.73) into the OGD posterior (5.40), we see that in the case of $\Omega = \mathbb{R}^n$, this uncertainty is measured via an *isotropic* covariance matrix conditioned on the gradient sequence $\mathbf{g}_{1:t}$, of the form

$$\Sigma_t^{\mathrm{MP}} = \frac{n}{\left\|\sum_{\tau=1}^{t} \mathbf{g}_\tau\right\|_2^2} \mathbf{I}_n. \tag{5.74}$$

The isotropic nature of this uncertainty estimate stems from the fact that the OGD algorithm approximates the Hessian of the loss functions $f_t(\cdot)$ as $\alpha \mathbf{I}_n$. Indeed, if we consider the second-order Taylor expansion of $f_t(\cdot)$ around $\mathbf{w}_t$, i.e.

$$f_t(\mathbf{w}) \approx f_t(\mathbf{w}_t) + \nabla f_t(\mathbf{w}_t)^{\mathrm{T}}(\mathbf{w} - \mathbf{w}_t) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_t)^{\mathrm{T}} \nabla^2 f_t(\mathbf{w}_t)(\mathbf{w} - \mathbf{w}_t), \tag{5.75}$$

and replace the terms $\nabla f_t(\mathbf{w}_t)$ and $\nabla^2 f_t(\mathbf{w}_t)$ with $\mathbf{g}_t$ and $\alpha \mathbf{I}_n$, respectively, we recover the OGD criterion (5.35).

Although the uncertainty estimates in Eq. (5.74) can be obtained at no computational overhead, we might expect them to perform poorly in practice, as they fail to account for the geometry of the loss functions. More accurate uncertainty estimates can be readily obtained from a Bayesian treatment of adaptive learning-rate methods equipped with better curvature estimates, such as ADAGRAD [Duchi et al., 2011] and ADAM [Kingma and Ba, 2015]. For instance, ADAGRAD approximates the Hessian via the square root of the cumulative outer product of subgradients, i.e.

$$\nabla^2 f_t(\mathbf{w}_t) \approx \sum_{\tau=1}^{t} \mathbf{g}_\tau \mathbf{g}_\tau^{\mathrm{T}}. \tag{5.76}$$

The beauty of our framework is that it can be easily extended to account for uncertainty estimates of the above type. A more detailed discussion of such extensions is beyond the scope of this thesis, and the reader is referred to [Kessler et al., 2020].

5.6 Bandit Feedback

So far in this chapter, we have assumed that either the entire loss function, or the gradient thereof evaluated at the current action, is observed by the decision maker at each stage. For some applications, however, these assumptions will not

be valid. Examples include the multi-armed bandit problem, dynamic pricing and Bayesian optimisation. To accommodate such exceptions, we turn now to the bandit feedback structure, in which the only feedback available to the learner is the actual incurred loss $f_t(\mathbf{w}_t)$, and she must choose a descent direction to follow based only on this minimal piece of information.

The bandit setting has previously been studied in both the static and dynamic OCO literatures. The simplest, and historically earliest, application of the online gradient descent algorithm to this setting was the work of Flaxman et al. [2005], in which the authors used point evaluations of convex functions to approximately estimate the unknown gradient. Later, Agarwal et al. [2010] extended the work of Flaxman et al. [2005] by considering a multipoint feedback structure, in which the learner can query each loss function at multiple points.

In the context of dynamic OCO under bandit feedback, the OGD algorithm was used in [Yang et al., 2016] and, more recently, in [Gao et al., 2018]. However, in both these papers, the learning rates were chosen under the assumption that the variation budget V_T is known to the player *before* the game begins. From a practical standpoint, this is a very unrealistic assumption. To rectify this shortcoming, we introduce in this section a novel adaptive online learning-rate method for dynamic OCO in the bandit setting, built upon the generalised passive-aggressive (GPA) model developed in Section 4.2. We name the resulting algorithm PACO, which stands for *passive-aggressive convex optimisation*.

5.6.1 A one-point gradient estimate

The first step in building an algorithm for bandit OCO is to estimate the unknown gradient at the current action, $\nabla f_t(\mathbf{w}_t)$. Although we cannot explicitly compute $\nabla f_t(\mathbf{w}_t)$, it is possible to find an *observable* random vector $\widehat{\mathbf{g}}_t$ that satisfies

$$\mathbb{E}(\widehat{\mathbf{g}}_t) \approx \nabla f_t(\mathbf{w}_t). \tag{5.77}$$

Thus, $\widehat{\mathbf{g}}_t$ can be seen as an estimator for the gradient.

How can we find an appropriate $\widehat{\mathbf{g}}_t$? In the multi-armed bandit problem, this is achieved by means of the importance sampling technique within the Hedge algorithm [Shalev-Shwartz, 2011; Freund and Schapire, 1997]. However, in more general cases, importance sampling cannot be applied because there is a continuum of points to sample, and also because the value of a function at a given point (a scalar) carries no information about its gradient (a vector). Thus, it is important to incorporate some 'directionality' that would allow us to generate a vector from a scalar. This is precisely the key idea behind the simultaneous stochastic approximation (SSA) method of Spall [1997], which estimates the gradient by artificially sampling the function not at the point of interest, but at a nearby, randomly chosen point.

To illustrate this method, it is convenient to begin with the one-dimensional case, i.e. functions defined over the real line. Specifically, suppose that we seek to estimate the derivative of a function $f : \mathbb{R} \to \mathbb{R}$ at some target point $\hat{w}$ using a single evaluation thereof. By definition, this derivative can be approximated by the difference quotient

$$f'(\hat{w}) \approx \frac{f(\hat{w}+\delta) - f(\hat{w}-\delta)}{2\delta}, \tag{5.78}$$

with the approximation becoming exact as $\delta \to 0^+$. Of course, this estimate requires two function evaluations (at $\hat{w}+\delta$ and $\hat{w}-\delta$), but it also suggests the following approach: simply make a uniform random draw from $\{\pm 1\}$ and sample $f(\cdot)$ at $\hat{w} \pm \delta$. Letting $u \in \{\pm 1\}$ denote the outcome of this draw, the difference quotient above can be rewritten as

$$\frac{f(\hat{w}+\delta) - f(\hat{w}-\delta)}{2\delta} = \frac{1}{\delta}\left[\frac{1}{2}f(\hat{w}+\delta) - \frac{1}{2}f(\hat{w}-\delta)\right] = \frac{1}{\delta}\mathbb{E}[f(\hat{w}+\delta u)u], \tag{5.79}$$

i.e. as the expectation of the random variable $f(\hat{w}+\delta u)u$. Thus, going back to Eq. (5.78), we see that $f'(\hat{w})$ can be approximated to $\mathcal{O}(\delta)$ by the single-shot estimator $\delta^{-1}f(\hat{w}+\delta u)u$.

Extending this construction to functions defined on $\mathbb{R}^n$ gives rise to the estimator

$$\frac{n}{\delta}f(\widehat{\mathbf{w}}+\delta\mathbf{u})\mathbf{u}, \tag{5.80}$$

where $\mathbf{u}$ is now drawn uniformly from the n-dimensional unit sphere centred at the origin, i.e. from

$$\mathbb{S}^n = \Big\{ \mathbf{x} \in \mathbb{R}^n \,\Big|\, \|\mathbf{x}\|_2 = 1 \Big\}, \tag{5.81}$$

and the factor n has been included for the purposes of dimensional scaling[18]. As in the one-dimensional case, the bias of the estimator vanishes in the limit $\delta \to 0^+$. More precisely, for $\delta > 0$ and K-Lipschitz continuous functions $f(\cdot)$, we have

$$\mathbb{E}_{\mathbf{u} \in \mathbb{S}^n} \left[\frac{n}{\delta} f(\widehat{\mathbf{w}} + \delta \mathbf{u}) \mathbf{u} \right] = \nabla f_\delta(\widehat{\mathbf{w}}), \tag{5.82}$$

where $f_\delta(\cdot)$ is a δ-smoothed approximation of $f(\cdot)$ such that $|f_\delta(\widehat{\mathbf{w}}) - f(\widehat{\mathbf{w}})| \leq K\delta$. Specifically,

$$f_\delta(\mathbf{w}) \equiv \mathbb{E}_{\mathbf{v} \in \mathbb{B}^n}[f(\mathbf{w} + \delta \mathbf{v})] \tag{5.83}$$

is simply the average value of $f(\cdot)$ over $\mathbb{B}^n_\delta$, an n-dimensional ball of radius δ centred at the origin, that is

$$\mathbb{B}^n_\delta = \Big\{ \mathbf{x} \in \mathbb{R}^n \,\Big|\, \|\mathbf{x}\|_2 \leq \delta \Big\}. \tag{5.84}$$

In other words, the one-point estimator in Eq. (5.80) provides a reasonable approximation to $\nabla f(\widehat{\mathbf{w}})$, up to a bias of at most $\mathcal{O}(\delta)$.

The above is the cornerstone of the 'gradient descent without a gradient' methodology proposed in [Flaxman et al., 2005]. In particular, the main idea behind this method is to apply a projected gradient descent scheme to generate an online sequence of *pivot points* $\widehat{\mathbf{w}}_t$. These variables are *not* played by the decision maker, but they act as base points ('pivots') to select an action at each stage based on the perturbation model

$$\mathbf{w}_t = \widehat{\mathbf{w}}_t + \delta \mathbf{u}_t, \tag{5.85}$$

[18]Heuristically, the reason for this is that there are n independent directions to sample, each with 'probability' $1/n$. Formally, this is a consequence of Stokes' theorem, a fundamental result in differential geometry which is used to prove the validity of the gradient estimator [Flaxman et al., 2005, Lemma 2.1.].

where $\mathbf{u}_t$ is drawn i.i.d. from the unit sphere $\mathbb{S}^n$. In so doing, the learner can follow the zeroth-order technique outlined above to estimate the gradient of $f_t(\cdot)$ at $\widehat{\mathbf{w}}_t$ as

$$\widehat{\mathbf{g}}_t = \frac{n}{\delta} f_t(\mathbf{w}_t)\mathbf{u}_t. \tag{5.86}$$

Clearly, this is not an estimate of the loss gradient at the current action $\mathbf{w}_t$. However, since the distance between the pivot and the learner's chosen action is of order $\mathcal{O}(\delta)$, the difference is not too big: optimistically, it should be small enough to generate a reasonable descent step.

5.6.2 Passive-aggressive convex optimisation

In this section, we present our PACO algorithm. Like most bandit algorithms, PACO works by estimating the 'missing information' and then passing on this estimate to a full-information algorithm. In this case, the 'missing information' are the gradients $\nabla f_t(\mathbf{w}_t)$. The underlying full-information algorithm is the GPA algorithm we introduced in Section 4.2 and shall briefly review below.

Recall that the GPA algorithm is a generalisation of the online passive-aggressive (PA) framework of Crammer et al. [2006] to loss functions other than the ϵ-insensitive loss. The algorithm initialises the weights at zero and computes subsequent weights according to

$$\mathbf{w}_{t+1} = \operatorname*{arg\,min}_{\mathbf{w}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|_2^2 \quad \text{s.t.} \quad \widetilde{f}_t^{(1)}(\mathbf{w}) \leq \epsilon, \qquad t = 1, 2, \ldots, \tag{5.87}$$

where $\widetilde{f}_t^{(1)}(\cdot)$ denotes the first-order Taylor approximation of the loss function $f_t(\cdot)$ around the contemporaneous action $\mathbf{w}_t$, i.e.

$$\widetilde{f}_t^{(1)}(\mathbf{w}) \equiv f_t(\mathbf{w}_t) + \mathbf{g}_t^{\mathrm{T}}(\mathbf{w} - \mathbf{w}_t), \qquad \mathbf{g}_t \in \partial f_t(\mathbf{w}_t). \tag{5.88}$$

Because $\epsilon \geq 0$, GPA relies on the following assumption which will also be required for PACO.

Assumption 5.6.1. *The loss functions $f_t(\cdot)$ are always non-negative, i.e.*

$$\mathrm{Range}(f_t) = \mathbb{R}_{\geq 0} \equiv \Big\{x \in \mathbb{R} \,|\, x \geq 0\Big\}, \qquad \forall\, t = 1, 2, \ldots \tag{5.89}$$

Note that the optimisation problem in Eq. (5.87) is an alternative formulation of the OGD criterion (5.35). By specifying an upper bound on the loss, GPA bypasses in a simple and elegant way the difficulty in attempting to find a suitable value for the learning rate. Certainly, the choice of the latter has been replaced with a choice of the insensitivity parameter ϵ, but one may argue that it is more natural and intuitive for a practitioner to pinpoint her maximum-loss threshold rather than the more abstract learning rate. Another nice property of the GPA method is that it admits a unique closed-form solution which, as we have shown in Section 4.2, takes the form

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\max\{0,\, f_t(\mathbf{w}_t) - \epsilon\}}{\|\mathbf{g}_t\|_2^2}\mathbf{g}_t. \tag{5.90}$$

In essence, the PACO algorithm is just a variant of GPA in which the true gradient $\mathbf{g}_t$ has to be replaced with an estimate $\widehat{\mathbf{g}}_t$ thereof, as the learner does not have access to the former quantity. So instead of using $\widetilde{f}_t^{(1)}(\cdot)$, PACO relies on the approximation

$$\widehat{f}_t^{(1)}(\mathbf{w}) \equiv f_t(\mathbf{w}_t) + \widehat{\mathbf{g}}_t^{\mathrm{T}}(\mathbf{w} - \widehat{\mathbf{w}}_t), \tag{5.91}$$

where $\widehat{\mathbf{g}}_t$ is given by Eq. (5.86) and $\widehat{\mathbf{w}}_t$ is a pivot point whose perturbation yields action $\mathbf{w}_t$ (see Eq. (5.85)). At each stage t, the algorithm then solves

$$\min_{\mathbf{w}\in\Omega_\delta} \frac{1}{2}\|\mathbf{w} - \widehat{\mathbf{w}}_t\|_2^2 \qquad \text{s.t.} \qquad \widehat{f}_t^{(1)}(\mathbf{w}) \leq \epsilon, \tag{5.92}$$

where the 'δ-shrunk' sub-region Ω_δ is defined as

$$\Omega_\delta \equiv \Big\{(1-\delta)\mathbf{w} \,\Big|\, \mathbf{w} \in \Omega\Big\}, \qquad \delta \in (0,1). \tag{5.93}$$

The rationale for minimising over Ω_δ instead of Ω is that the chosen action $\mathbf{w}_t = \widehat{\mathbf{w}}_t + \delta\mathbf{u}_t$ may lie outside the feasible region Ω if the pivot point $\widehat{\mathbf{w}}_t$ lies too close to the boundary of Ω. One can keep $\widehat{\mathbf{w}}_t$ away from the boundary by projecting it onto the 'δ-shrunk' sub-region[19]. This projection introduces an additional error of order $\mathcal{O}(\delta)$, but since all other estimation errors are of the same order, it does not affect PACO's qualitative behaviour.

[19]For any $\mathbf{w} \in \Omega_\delta$ and any unit vector $\mathbf{u}$, it holds that $(\mathbf{w} + \delta\mathbf{u}) \in \Omega$ [Flaxman et al., 2005, Observation 2].

Like OGD, PACO solves its optimisation problem (5.92) in two steps, namely an update step whose solution is such that

$$\widehat{\mathbf{v}}_{t+1} = \underset{\mathbf{v}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{v} - \widehat{\mathbf{w}}_t\|_2^2 \qquad \text{s.t.} \qquad \widehat{f}_t^{(1)}(\mathbf{w}) \leq \epsilon, \tag{5.94}$$

followed by a projection step yielding the corresponding pivot point:

$$\widehat{\mathbf{w}}_{t+1} = \underset{\mathbf{w}\in\Omega_\delta}{\arg\min}\ \|\mathbf{w} - \widehat{\mathbf{v}}_{t+1}\|_2^2. \tag{5.95}$$

By analogy with Eq. (5.90), we have

$$\widehat{\mathbf{v}}_{t+1} = \widehat{\mathbf{w}}_t - \frac{\max\{0,\ f_t(\mathbf{w}_t) - \epsilon\}}{\|\widehat{\mathbf{g}}_t\|_2^2}\widehat{\mathbf{g}}_t, \tag{5.96}$$

whence

$$\widehat{\mathbf{w}}_{t+1} = \Pi_{\Omega_\delta}\left[\widehat{\mathbf{w}}_t - \frac{\max\{0,\ f_t(\mathbf{w}_t) - \epsilon\}}{\|\widehat{\mathbf{g}}_t\|_2^2}\widehat{\mathbf{g}}_t\right]. \tag{5.97}$$

Note that we update the pivot points and not the selected actions, since the latter are updated via the perturbation model in Eq. (5.85). The entire procedure is detailed in Algorithm 9 which is subject to the additional condition set out below.

Assumption 5.6.2. *The decision set Ω contains the unit ball centred at the origin, i.e. $\mathbb{B}_1^n \subseteq \Omega$.*

This is a standard assumption in the study of bandit OCO algorithms. In case it does not hold, we can always map Ω to a lower-dimensional space. Technically, we must also assume that Ω is a closed set, so that the projection operation is well-defined.

Algorithm 9 PACO: Passive-Aggressive Convex Optimisation

Input: convex and closed action space $\Omega \subseteq \mathbb{R}^n$ containing the unit ball $\mathbb{B}_1^n$, exploration parameter $\delta \in (0, 1)$, insensitivity parameter $\epsilon \geq 0$
Initialisation: initial pivot point $\widehat{\mathbf{w}}_1 = \mathbf{0}_{n\times 1}$
for $t = 1, 2, \ldots$ **do**
    draw $\mathbf{u}_t$ uniformly from $\mathbb{S}^n$
    play $\mathbf{w}_t = \widehat{\mathbf{w}}_t + \delta \mathbf{u}_t$ and receive loss $f_t(\mathbf{w}_t) \geq 0$
    estimate the gradient of the current action as

$$\widehat{\mathbf{g}}_t = \frac{n}{\delta} f_t(\mathbf{w}_t)\mathbf{u}_t$$

    update the pivots:

$$\widehat{\mathbf{w}}_{t+1} = \Pi_{\Omega_\delta}\left[\widehat{\mathbf{w}}_t - \frac{\max\{0,\ f_t(\mathbf{w}_t) - \epsilon\}}{\|\widehat{\mathbf{g}}_t\|_2^2}\widehat{\mathbf{g}}_t\right], \qquad \Omega_\delta \equiv \{(1-\delta)\mathbf{w} \,|\, \mathbf{w} \in \Omega\}$$

end for

While Algorithm 9 still requires the specification of the exploration parameter δ, it has removed one degree of freedom from bandit OCO algorithms based on the OGD method, namely the cumbersome task of choosing an appropriate learning rate. It is important to note that δ controls the bias-variance trade-off of the algorithm. When it is too large, the gradient estimator in Algorithm 9 will improve in terms of precision (inverse variance), but deteriorate in terms of accuracy. Conversely, for small δ, the estimated gradient will be relatively unbiased, but immensely variable. When using Algorithm 9, we will therefore have to carefully balance these two conflicting objectives. Previous work on dynamic OCO in the bandit setting suggests setting $\delta = \mathcal{O}(1/T)$ [Yang et al., 2016; Gao et al., 2018].

5.7 Application Domain

The algorithms and methods presented in this chapter are expository in nature, but designed with a view to real-world applications. As we already mentioned, one clear application domain is in the area of online portfolio selection, to which the next chapter shall be entirely devoted. Other areas of application include but are not limited to the following:

- sequential compressed sensing of a dynamic scene and tracking dynamic social networks [Hall and Willett, 2013];
- dynamic texture analysis, solar flare detection, traffic surveillance, and tracking self-exciting point processes and network behaviour in the Enron email corpus [Hall and Willett, 2015];
- switching zero-sum games with uncoupled dynamics [Jadbabaie et al., 2015];
- tracking of a time-varying parameter with unknown dynamics [Mokhtari et al., 2016];
- online statistical learning [Zhang et al., 2017];
- decentralised tracking of dynamic parameters [Shahrampour and Jadbabaie, 2018].

More generally, our proposed algorithms can be used for any type of problem whose *offline* version can be posed as follows:

$$\begin{aligned} \underset{\mathbf{w}_1,\ldots,\mathbf{w}_T}{\text{minimise}} \quad & \sum_{t=1}^{T} f_t(\mathbf{w}_t) \\ \text{subject to} \quad & \mathbf{w}_t \in \Omega,\ t \in [T], \end{aligned} \tag{5.98}$$

where Ω is a convex compact set and $f_t : \Omega \to \mathbb{R}$ is a convex loss function. Clearly, these encompass any static OCO problem. Once we have cast the portfolio-choice problem in the form (5.98), we will be able to solve it in an *online* manner, using the methods introduced in this chapter.

6

Application to Online Portfolio Selection

Contents

Online portfolio selection (OLPS), a fundamental problem in computational finance, has attracted increasing interest from the artificial intelligence and machine learning communities in recent years. Empirical evidence shows that highs and

lows in the stock market are temporary phenomena, as daily equity prices are likely to follow a *mean-reverting* pattern. While existing OLPS strategies capturing this pattern achieve good empirical performance on many real data sets, they suffer from *data-snooping bias*—the danger that backtest performance is inflated relative to future performance because one has over-optimised a strategy's hyperparameters based on transient noise in the historical data.

To overcome this significant limitation, we adapt the (hyperparameter-free) full-information and gradient-feedback algorithms introduced in the previous chapter to the OLPS context. We shall see that the former induces a concentrated follow-the-loser procedure whereby, on each trading day, the entire wealth is allocated to the worst-performing stock, with the expectation it will benefit from the largest appreciation over the following trading day. For this reason, we shall refer to this strategy as *online maximum reversion*, or OLMAX for short. We shall then borrow ideas from the MAPGRAD framework to design self-tuning variants of top-performing OLPS algorithms, such as passive-aggressive mean reversion (PAMR) and online moving average reversion (OLMAR), and accordingly name them *adaptive passive-aggressive mean reversion* (AdaPAMR) and *adaptive online moving average reversion* (AdaOLMAR), respectively. Finally, we shall see that AdaPAMR and AdaOLMAR turn out to be useful in cross-asset allocation problems, in which reversion strategies such as PAMR, OLMAR and even OLMAX have a tendency to underperform in the presence of transaction costs.

6.1 Introduction

Portfolio selection is a fundamental computational finance problem extensively explored across several fields, ranging from traditional finance theory and quantitative finance, to machine learning and artificial intelligence [Li and Hoi, 2014, 2015]. Generally, the goal is to achieve a long-run target by sequentially allocating wealth across a set of assets. Two major schools of principles and theories for portfolio selection include *i)* mean-variance theory [Markowitz, 1952] which balances the expected return (mean) and risk (variance) of a portfolio, and is suitable for

single-period portfolio selection, and *ii)* Kelly investment [Kelly, 1956; Breiman, 1961; Thorp, 1969, 1971], which aims to maximise the expected log return of a portfolio and is directly applicable to multiperiod portfolio selection. Due to the sequential nature of a real-world portfolio selection task, many recent OLPS techniques often follow the second approach.

One important property exploited by many existing studies [Borodin et al., 2004; Li et al., 2011, 2012, 2015] is the *mean reversion* property, which assumes that poor performing assets will perform well in subsequent periods and vice-versa. Although some recently proposed mean-reversion algorithms [Li et al., 2011, 2012, 2015] have achieved promising results on many real-world equity data sets, their performance relies on user-defined hyperparameters and is therefore fraught with *data-snooping bias*, i.e. the risk that there is a significant divergence between the historical and future performance of a strategy because its hyperparameters have been overfitted to the historical data. This bias is pervasive in the business of predictive statistical models of historical data, but is especially serious in finance because of the limited amount of available data. High-frequency data, while abundant in supply, is useful only for high-frequency models.

A useful rule of thumb to avoid the data-snooping pitfall is that the less independent data one has, the fewer adjustable parameters one should employ in a trading model. In this spirit, we adapt the dynamic online convex optimisation algorithms from Chapter 5 to the online portfolio selection setting, as these come with the benefit of being hyperparameter-free, among others. The basic idea is to cast the OLPS problem in terms of the minimisation of dynamic regret. This yields two families of simple, scalable and efficient algorithms for online portfolio selection, depending on the amount of information queried about the loss functions at each stage (full-information or gradient feedback). We shall demonstrate the superiority of the first family, OLMAX, in exploiting daily mean reversion patterns in stock markets, based on the same equity data sets that are widely used in the OLPS literature. As for the second family, derived from the MAPGRAD algorithm (Algorithm 8), we shall see in a simple experiment that it is more suitable for the

management of portfolios composed of different asset classes, and not just equities, especially in the presence of transaction costs.

6.2 Preliminaries

6.2.1 Notation

Before stating the online portfolio selection problem, we introduce some useful notation to be used throughout the chapter. As before, vectors are denoted by lower-case bold Roman letters such as $\mathbf{v}$, and all vectors are assumed to be column vectors. The exceptions are $\mathbf{0}$ and $\mathbf{1}$, which denote the zero vector and a vector of ones, respectively, whose dimensions are to be inferred from the context. A superscript T denotes the transpose of a vector, so that $\mathbf{v}^{\mathrm{T}}$ will be a row vector.

One common operation is the product or division of a vector $\mathbf{v}$ by a scalar s, whereby each element v_i of the vector is multiplied or divided by s, i.e. $[s\mathbf{v}]_i = s \times v_i$ and $\left[\frac{s}{\mathbf{v}}\right]_i = \frac{s}{v_i}$. With a slight abuse of notation, for two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, we use $\frac{\mathbf{v}_1}{\mathbf{v}_2}$ to denote the element-wise division of $\mathbf{v}_1$ by $\mathbf{v}_2$, meaning $\left[\frac{\mathbf{v}_1}{\mathbf{v}_2}\right]_i = v_{1,i}/v_{2,i}$. Similarly, $\mathbf{v}_1 \bigotimes \mathbf{v}_2$ denotes the element-wise product of $\mathbf{v}_1$ times $\mathbf{v}_2$, that is, $[\mathbf{v}_1 \bigotimes \mathbf{v}_2]_i = v_{1,i} \times v_{2,i}$. The same definition applies to the summation $\mathbf{v}_1 + \mathbf{v}_2$ and subtraction $\mathbf{v}_1 - \mathbf{v}_2$ operations. The dot/inner product between two vectors is defined as $\mathbf{a} \cdot \mathbf{b} \equiv \mathbf{a}^{\mathrm{T}}\mathbf{b} = \sum_{i=1}^{m} a_i \times b_i$. We shall use the two notations $\mathbf{a} \cdot \mathbf{b}$ and $\mathbf{a}^{\mathrm{T}}\mathbf{b}$ interchangeably.

Lastly, let $\mathbb{N}$ be the set of the positive integers. For $l \in \mathbb{N}$, we define $[l] \equiv \{1, 2, 3, \ldots, l\}$.

6.2.2 The online portfolio selection problem

We now formally describe the online portfolio selection (OLPS) problem. Consider an investment task over a financial market with m assets and a T-period horizon. On the t-th period, the asset prices are represented by a *closing-price vector* $\mathbf{p}_t \in \mathbb{R}_+^m$. The price changes are represented by a *price-relative vector* $\mathbf{x}_t \in \mathbb{R}_+^m$ such that $x_{t,i} \equiv \frac{p_{t,i}}{p_{t-1,i}}$ for each asset $i \in [m]$. Thus, an investment in asset i on the t-th period increases

by a factor of $x_{t,i}$. Let us denote by $\mathbf{x}_{t_1:t_2} \equiv \{\mathbf{x}_{t_1}, \mathbf{x}_{t_1+1}, \ldots, \mathbf{x}_{t_2}\}$ a sequence of price-relative vectors ranging from period t_1 to t_2. Therefore, $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ represents the sequence of price-relative vectors over the entire T-period horizon.

An investment on the t-th period is specified by a *portfolio vector* $\mathbf{b}_t = (b_{t,1}, \ldots, b_{t,m})$, where $b_{t,i}$ represents the proportion of wealth invested in asset i. Typically, we assume the portfolio is self-financed and that no margin/short sale is allowed, so that each entry of a portfolio is non-negative and all entries add up to one:

$$\mathbf{b}_t \in \Delta_m \equiv \left\{\mathbf{b}_t : \mathbf{b}_t \in \mathbb{R}_+^m, \sum_{i=1}^m b_{t,i} = 1\right\}. \tag{6.1}$$

The investment procedure is represented by a *portfolio strategy* such that $\mathbf{b}$ is initialised at the equally-weighted portfolio by convention, that is $\mathbf{b}_1 = \mathbf{1}/m$, followed by a sequence of mappings $\mathbf{b}_t : \mathbb{R}_+^{m(t-1)} \to \Delta_m$ for $t = 2, 3, \ldots$, where $\mathbf{b}_t = \mathbf{b}_t(\mathbf{x}_{1:t-1})$ is the t-th portfolio given the historical market sequence $\mathbf{x}_{1:t-1} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}$. We denote by $\mathbf{b}_{1:T} = \{\mathbf{b}_1, \ldots, \mathbf{b}_T\}$ the strategy for T periods.

On the t-th period, a portfolio $\mathbf{b}_t$ produces a *portfolio period return* s_t, that is, the wealth increases by a factor of $s_t \equiv \mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t = \sum_{i=1}^m b_{t,i}x_{t,i}$. We shall assume all profits are reinvested, which implies that wealth grows in a multiplicative fashion. Thus, after T periods, a portfolio strategy $\mathbf{b}_{1:T}$ produces a *portfolio cumulative wealth* of S_T, which amounts to the initial wealth times a factor of $\prod_{t=1}^T s_t$, that is

$$S_T = S_0 \prod_{t=1}^T s_t = S_0 \prod_{t=1}^T \mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t, \tag{6.2}$$

where S_0 denotes the initial wealth and can be set to \$1 without any loss of generality.

Finally, we formally formulate the OLPS procedure, and outline its algorithmic framework in Algorithm 10. In this task, a portfolio manager is a decision maker whose goal is to produce a portfolio strategy $\mathbf{b}_{1:T}$ so as to maximise her cumulative wealth S_T. She computes the portfolios sequentially. On each period t, the manager has access to the sequence of all previous price-relative vectors $\mathbf{x}_{1:t-1}$. Then, she computes a new portfolio $\mathbf{b}_t$ before observing the next price-relative vector $\mathbf{x}_t$, based on a decision criterion that varies among different managers. The portfolio $\mathbf{b}_t$ is scored based on the portfolio period return s_t. This procedure is repeated until

the end of the T-period investment horizon, and the portfolio strategy is finally scored according to its cumulative wealth S_T.

It is important to note that we have made several general and common assumptions in the above description:

1. Transaction costs: there are no commission fees or taxes;

2. Market liquidity: one can buy and sell any desired amount, even fractional, at the last closing price of any given trading period;

3. Market impact: no portfolio strategy shall influence the market, or the prices of other assets.

Note that although these assumptions are commonly made in portfolio theory, they are non-trivial in practice. We shall further analyse and discuss their implications and effects later in the empirical sections.

Algorithm 10 Online Portfolio Selection Framework

Initialisation: $\mathbf{b}_1 = \left(\frac{\mathbf{1}}{m}, \ldots, \frac{\mathbf{1}}{m}\right)$, $S_0 = 1$
for $t = 1, 2, \ldots, T$ **do**
  the market reveals the price-relative vector $\mathbf{x}_t$
  the portfolio manager realises a profit (or loss) of $s_t = \mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t$, and her wealth changes to $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t)$
  if $t < T$ **then**
    the portfolio manager updates her portfolio from $\mathbf{b}_t$ to $\mathbf{b}_{t+1} \in \Delta_m$
  end if
end for

6.2.3 Dynamic OCO formulation

Our ultimate goal is to be able to apply the techniques developed in Chapter 5 to online portfolio selection. To this end, we need to cast the OLPS problem described above into a dynamic online convex optimisation (OCO) problem, which is the purpose of this subsection.

Consider a clairvoyant investor who knows the sequence $\mathbf{x}_{1:T}$ of price-relative vectors in advance. Given her perfect foresight and a sequence $u_{1:T}$ of non-decreasing

utility functions of the form $u_t(\mathbf{b}) = u(\mathbf{b}^{\mathrm{T}}\mathbf{r}_t)$, where $\mathbf{r}_t \equiv \mathbf{x}_t - \mathbf{1}$, she selects the maximiser

$$\mathbf{b}_t^* \equiv \operatorname*{arg\,max}_{\mathbf{b}\in\Delta_m} \; u(\mathbf{b}^{\mathrm{T}}\mathbf{r}_t) = \operatorname*{arg\,max}_{\mathbf{b}\in\Delta_m} \; \mathbf{b}^{\mathrm{T}}\mathbf{r}_t = \operatorname*{arg\,max}_{\mathbf{b}\in\Delta_m} \; (1 + \mathbf{b}^{\mathrm{T}}\mathbf{r}_t) = \operatorname*{arg\,max}_{\mathbf{b}\in\Delta_m} \; \mathbf{b}^{\mathrm{T}}\mathbf{x}_t, \tag{6.3}$$

where the second equality follows from the fact that $u(\cdot)$ is monotonically increasing, while the last one is a consequence of the constraint $\mathbf{b}^{\mathrm{T}}\mathbf{1} = 1$. Given the resulting clairvoyant strategy $\mathbf{b}_{1:T}^*$, we define the clairvoyant portfolio returns R_t^* as

$$R_t^* \equiv \mathbf{b}_t^* \cdot \mathbf{r}_t, \qquad t \in [T]. \tag{6.4}$$

From a dynamic OCO perspective, the aim of a non-clairvoyant investor is to replicate the fictitious clairvoyant's decisions to the best possible extent. While there are many different interpretations of what 'best' means in this context, we shall take it to mean 'least *tracking error*', resulting in a dynamic regret of the form

$$\mathbf{TE}_T^d \equiv \sum_{t=1}^{T} R_t^* - \sum_{t=1}^{T} \mathbf{b}_t \cdot \mathbf{r}_t = \sum_{t=1}^{T} (\mathbf{b}_t^* \cdot \mathbf{r}_t - \mathbf{b}_t \cdot \mathbf{r}_t) = \sum_{t=1}^{T} (\mathbf{b}_t^* \cdot \mathbf{x}_t - \mathbf{b}_t \cdot \mathbf{x}_t), \tag{6.5}$$

where the last equality is a direct consequence of $\mathbf{b}_t^*$, $\mathbf{b}_t \in \Delta_m$. Comparing (6.5) and the definition (5.2) of dynamic regret, we see that the loss function $f_t(\cdot)$ underlying (6.5) is given by

$$f_t(\mathbf{b}) = -\mathbf{b}^{\mathrm{T}}\mathbf{x}_t, \qquad t \in [T]. \tag{6.6}$$

For the non-clairvoyant portfolio manager, Eq. (6.5) implies that she picks a strategy so as to *maximise* her cumulative return or, equivalently,

$$\begin{aligned} \underset{\mathbf{b}_1,\ldots,\mathbf{b}_T}{\text{minimise}} \quad & \sum_{t=1}^{T} f_t(\mathbf{b}_t) \\ \text{subject to} \quad & \mathbf{b}_t \in \Delta_m, \; t \in [T]. \end{aligned} \tag{6.7}$$

In other words, Eq. (6.5) gives rise to a *follow-the-winner* approach, which is characterised by sequential transfers of wealth from the worst to the best performing assets (see Chapter 3 for details). Moreover, the empirical justification of (6.5) is sound, as the concept of tracking error is widely used in active portfolio management to measure how closely an actively managed portfolio (e.g. a mutual fund) follows the index (e.g. a stock index such as the S&P500) to which it is benchmarked. We can clearly draw an analogy between this benchmark and the clairvoyant's portfolio.

6.3 Online Maximum Reversion

6.3.1 Motivation

Empirical results [Li et al., 2011, 2012, 2015] show that mean reversion, which assumes that assets with lacklustre performance on a given trading period can be expected to perform well on subsequent periods, may better fit equity markets. By exploiting the mean-reverting pattern of daily stock prices, the CWMR (confidence-weighted mean reversion), PAMR and OLMAR algorithms are able to achieve excellent results on various equity index data sets. Nonetheless, they all rely on a set of hyperparameters whose values need to be manually tuned by practitioners. Clearly, this practice invites data-snooping bias.

To avoid falling into the data-snooping trap, we propose *online maximum reversion* (OLMAX), a new family of 'hyperparameter-deprived' algorithms[20] for online portfolio selection that capitalise on single-period as well as multiperiod mean reversion in the stock market. The key to these algorithms is that they frame the OLPS problem as the dynamic OCO problem given in Eq. (6.7). Because the underlying loss function $f_t(\cdot)$ is *entirely* revealed at the end of each trading period t, they are then able to exploit the dynamic OCO methods for the full-information setting from Chapter 5. The only caveat concerns the *regularity* of the loss-function sequence, which we shall empirically validate shortly. Beforehand, however, we list in Table 6.1 the equity data sets that we shall use for this purpose as well as in the experimental section 6.4, to which we defer their description.

Table 6.1: Summary of the six equity data sets employed for the empirical evaluation.

Data set	Market	Region	Time frame	# Periods	# Assets
NYSE (O)	Stock	US	07/03/1962 – 12/31/1984	5651	36
NYSE (N)	Stock	US	01/01/1985 – 06/30/2010	6431	23
TSE	Stock	CA	01/04/1994 – 12/31/1998	1259	88
SP500	Stock	US	01/02/1998 – 01/31/2003	1276	25
MSCI	Stock	Global	04/01/2006 – 03/31/2010	1043	24
DJIA	Stock	US	01/14/2001 – 01/14/2003	507	30

[20] By 'hyperparameter-deprived', we mean algorithms that have no or, at worst, fewer hyperparameters than existing comparable strategies.

As we discussed in the previous chapter, we capture the 'niceness' of the adversarial sequence of loss functions through the notion of *temporal variability* based on the supremum norm, whose definition we reproduce here for completeness:

$$\mathrm{Var}(f_{1:T}) \equiv \sum_{t=2}^{T} \|f_t - f_{t-1}\|_\infty, \tag{6.8}$$

where for any bounded functions g and h from $\mathcal{X}$ into $\mathbb{R}$, we denote $\|g-h\|_\infty \equiv \sup_{x\in\mathcal{X}} |g(x)-h(x)|$. In Figure 6.1, we plotted the temporal variability for each of the six stock data sets listed in Table 6.1. The plots reveal continuous changes at a non-constant rate. Moreover, we can see that the variation across all data sets is a sublinear function of the horizon length T. Hence, one may achieve sublinear tracking error relative to the dynamic clairvoyant [Besbes et al., 2015].

Figure 6.1: Temporal variability $\mathrm{Var}(f_{1:T}) = \sum_{t=2}^{T} \|f_t - f_{t-1}\|_\infty$ for the six equity data sets from Table 6.1. The solid red line represents the identity line (i.e. $y = x$).

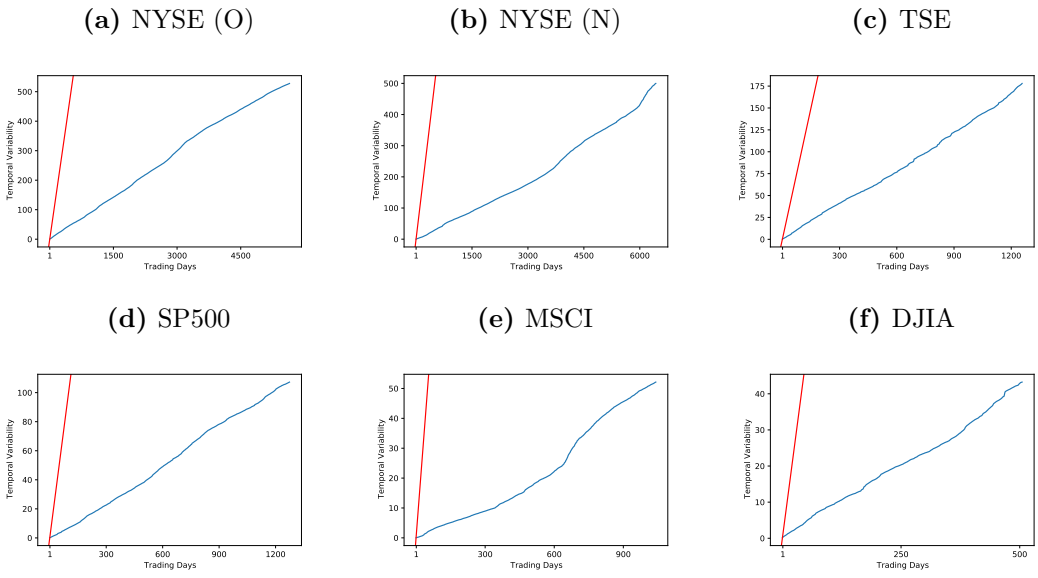
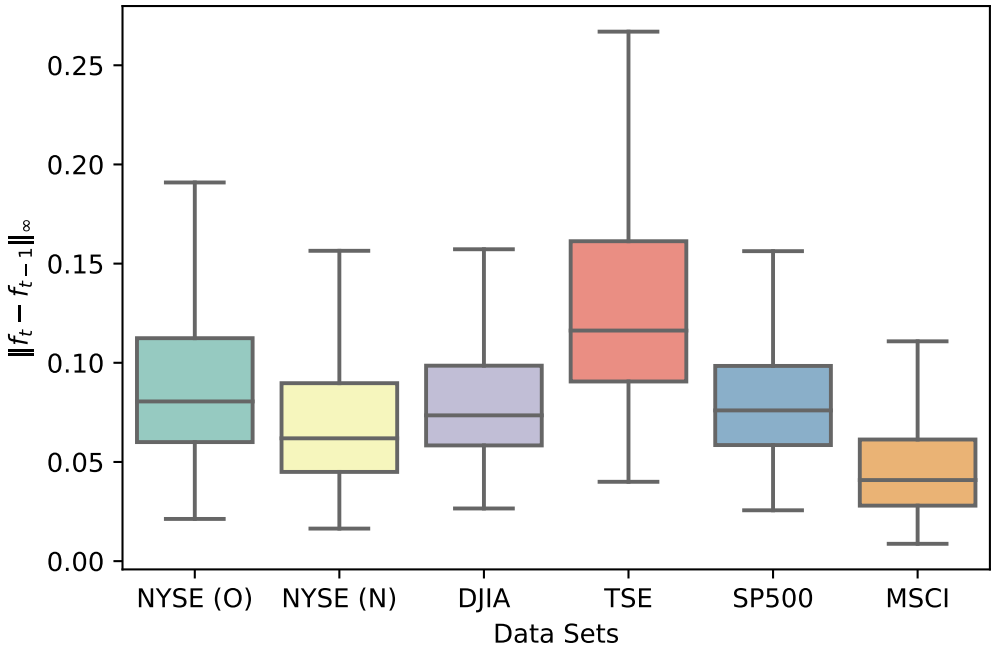


Figure 6.2 allows us to get a sense of the tracking-error magnitudes for the six equity data sets. Given the tight interquartile ranges and relatively low medians, we should not expect these to be very significant.

Figure 6.2: Box plot of temporal-variability components $\|f_t - f_{t-1}\|_\infty$ across the six equity data sets presented in Table 6.1.



6.3.2 Formulations

Single-period mean reversion

Our single-period mean reversion algorithm, termed SingleOLMAX, is based on the single-period mean reversion ideas described in [Li et al., 2012, Section 4.1]. The goal is to solve the problem in Eq. (6.7), with $f_t(\mathbf{b}) = \mathbf{b}^{\mathrm{T}}\mathbf{x}_t$ instead of Eq. (6.6). The reason for this is because we seek to capitalise on the appreciation of poor-performing stocks, expecting their bad performance to be transient and to ultimately 'revert to the mean', which calls for a *follow-the-loser* investing approach.

We saw in the previous subsection that the temporal variability of the loss-function sequence is sublinear in T, for all the stock data sets. As shown in Proposition 5.4.1, the following portfolio strategy will therefore achieve a sublinear tracking error:

$$\mathbf{b}_{t+1} \in \operatorname*{arg\,min}_{\mathbf{b}\in\Delta_m} \mathbf{b}^{\mathrm{T}}\mathbf{x}_t, \qquad t \in [T-1], \tag{6.9}$$

with the convention that $\mathbf{b}_1 = \mathbf{1}/m$. To solve this linear program, it is convenient to first sort the components of $\mathbf{x}_t$ in ascending order, so that

$$x_{t,1} \leq x_{t,2} \leq \ldots \leq x_{t,i} \leq \ldots \leq x_{t,m}. \tag{6.10}$$

We then check the sign of $x_{t,1}$. If the latter is non-negative, then the minimal value of $\mathbf{b}^{\mathrm{T}}\mathbf{x}_t$ is achieved by setting $b_{t+1,i} = 0$ for all $i \in [m]$, which, after performing the projection onto Δ_m, results in the equally-weighted portfolio $\mathbf{1}/m$. Conversely, if $x_{t,1} < 0$, then the optimal weights are such that $b_{t+1,1} = 1$ and $b_{t+1,k} = 0$ for all $k > 1$. Wrapping up the two cases, we obtain the following strategy:

$$b_{t+1,i} = \begin{cases} 0 & \text{if } x_{t,1} \geq 0 \\ \delta_{i1} & \text{otherwise} \end{cases}, \qquad \forall\, t \in [T-1],\, \forall\, i \in [m], \tag{6.11}$$

where δ_{ij} denotes the Kronecker delta. This choice is quite obvious: we allocate our entire budget to the investment with the lowest price relative.

The procedure is outlined in Algorithm 11. Note that it is completely hyperparameter-free and thereby not prone to the data-snooping bias, unlike its cousin PAMR.

Algorithm 11 SingleOLMAX: Single-Period Online Maximum Reversion

Initialisation: initial portfolio $\mathbf{b}_1 = \frac{\mathbf{1}}{m}$, initial wealth $S_0 = 1$
for $t = 1, 2, \ldots, T$ **do**
  observe stock price relatives $\mathbf{x}_t$
  update wealth: $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t)$
  if $t < T$ **then**
    sort $\mathbf{x}_t$ in ascending order
    update the portfolio:

$$b_{t+1,i} = \begin{cases} 0 & \text{if } x_{t,1} \geq 0 \\ \delta_{i1} & \text{otherwise} \end{cases}, \qquad i \in [m]$$

  end if
end for

Multiperiod mean reversion

One common criticism of the single-period mean reversion assumption is that it fails to account for price reversals that occur over longer periods (see, e.g., [Li

et al., 2015]). To overcome this limitation, we follow Li et al. [2015], and modify the SingleOLMAX criterion in Eq. (6.9) as follows:

$$\mathbf{b}_{t+1} \in \underset{\mathbf{b}\in\Delta_m}{\arg\max}\ \mathbf{b}\cdot\widetilde{\mathbf{x}}_{t+1}, \qquad t\in[T-1], \tag{6.12}$$

where $\widetilde{\mathbf{x}}_{t+1}$ is a forecast of the $(t+1)$-th price-relative vector based upon the sequence $\mathbf{x}_{1:t}$ of observed price relatives. From a dynamic OCO standpoint, we can heuristically justify Eq. (6.12) via

$$\underset{\mathbf{b}\in\Delta_m}{\arg\max}\ \mathbf{b}\cdot\widetilde{\mathbf{x}}_{t+1} \approx \mathbf{b}^*_{t+1}, \tag{6.13}$$

where $\mathbf{b}^*_{t+1}$ is the clairvoyant's optimal portfolio for period $(t+1)$, i.e. $\mathbf{b}^*_{t+1} = \arg\max_{\mathbf{b}\in\Delta_m}\ \mathbf{b}\cdot\mathbf{x}_{t+1}$. By analogy with Eq. (6.9), after sorting $\widetilde{\mathbf{x}}_{t+1}$ in ascending order, we can write the unique solution to the optimisation problem in Eq. (6.12) as

$$b_{t+1,i} = \begin{cases} 0 & \text{if } \widetilde{x}_{t+1,m} \le 0 \\ \delta_{im} & \text{otherwise} \end{cases}, \qquad \forall\, t\in[T-1],\ \forall\, i\in[m]. \tag{6.14}$$

We now discuss how to form the predictions $\widetilde{\mathbf{x}}_{t+1}$. The basic idea is to exploit the reversion of the current prices $\mathbf{p}_t$ to $\widetilde{\mathbf{p}}_{t+1} = \mathrm{MA}_t$, where MA_t denotes the moving average of price vectors at the end of period t. Following Li et al. [2015], we shall consider two types of moving average, namely a simple moving average (SMA)

$$\mathrm{SMA}_t = \frac{1}{w}\sum_{\tau=t-w+1}^{t}\mathbf{p}_\tau, \tag{6.15}$$

where w denotes the size of the lookback window, and an exponential moving average (EMA)

$$\mathrm{EMA}_t = \alpha\mathbf{p}_t + (1-\alpha)\mathrm{EMA}_{t-1}, \tag{6.16}$$

where $\alpha\in(0,1)$ is the smoothing factor. The corresponding price-relative forecasts are respectively:

$$\widetilde{\mathbf{x}}^{\mathrm{SMA}}_{t+1} \equiv \frac{\mathrm{SMA}_t}{\mathbf{p}_t} = \frac{1}{w}\left(\frac{\mathbf{p}_t}{\mathbf{p}_t} + \frac{\mathbf{p}_{t-1}}{\mathbf{p}_t} + \ldots + \frac{\mathbf{p}_{t-w+1}}{\mathbf{p}_t}\right) = \frac{1}{w}\left(\mathbf{1} + \frac{1}{\mathbf{x}_t} + \ldots + \frac{1}{\bigotimes_{h=0}^{w-2}\mathbf{x}_{t-h}}\right), \tag{6.17}$$

$$\widetilde{\mathbf{x}}^{\mathrm{EMA}}_{t+1} \equiv \frac{\mathrm{EMA}_t}{\mathbf{p}_t} = \frac{\alpha\mathbf{p}_t + (1-\alpha)\mathrm{EMA}_{t-1}}{\mathbf{p}_t} = \alpha + (1-\alpha)\frac{\widetilde{\mathbf{x}}^{\mathrm{EMA}}_t}{\mathbf{x}_t}. \tag{6.18}$$

We present the resulting two variants of our proposed multiperiod online maximum reversion (MultiOLMAX) method in Algorithm 12. Compared to their OLMAR counterparts in [Li et al., 2015], they rely on a single hyperparameter instead of two, namely the window size w for the SMA variant and the smoothing factor α for the EMA variant[21].

Algorithm 12 MultiOLMAX: Multiperiod Online Maximum Reversion

Input: Window size $w \geq 2$ (SMA variant), smoothing factor $0 < \alpha < 1$ (EMA variant)
Initialisation: initial portfolio $\mathbf{b}_1 = \frac{\mathbf{1}}{m}$, initial wealth $S_0 = 1$
for $t = 1, 2, \ldots, T$ **do**
  observe stock price relatives $\mathbf{x}_t$
  update wealth: $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t)$
  if $t < T$ **then**
    predict next price-relative vector:

$$\widetilde{\mathbf{x}}_{t+1} = \begin{cases} \frac{1}{w}\left(\mathbf{1} + \frac{1}{\mathbf{x}_t} + \ldots + \frac{1}{\bigotimes_{h=0}^{w-2} \mathbf{x}_{t-h}}\right) & \text{(SMA variant)} \\ \alpha + (1-\alpha)\frac{\widetilde{\mathbf{x}}_t}{\mathbf{x}_t} & \text{(EMA variant)} \end{cases}$$

    sort $\widetilde{\mathbf{x}}_{t+1}$ in ascending order
    update the portfolio:

$$b_{t+1,i} = \begin{cases} 0 & \text{if } \widetilde{x}_{t+1,m} \leq 0 \\ \delta_{im} & \text{otherwise} \end{cases}, \qquad i \in [m].$$

  end if
end for

6.4 Experiments

In this section, we will present an extensive set of empirical studies, including our experimental test bed, protocols, comparison schemes, results and a detailed empirical analysis.

6.4.1 Experimental test bed on real data

To be able to compare the empirical performance of our OLMAX framework against that of other OLPS strategies, we focus on historical daily stock prices. Earlier

[21] OLMAR additionally depends on a reversion threshold ϵ.

studies have restricted their empirical analyses to equity data as they can be readily obtained from public domains such as Yahoo! Finance and Google Finance[22], which ensures experimental reproducibility. Summarised in Table 6.1, we employ six real and diverse data sets from several financial markets[23].

The first data set, NYSE (O), is a standard data set pioneered by Cover [1991] and used in subsequent works [Helmbold et al., 1998; Borodin et al., 2004; Agarwal et al., 2006; Györfi et al., 2006]. This data set contains 5,651 daily price relatives of 36 stocks[24] from the New York Stock Exchange (NYSE) over a 22-year period from July 3, 1962 until December 31, 1984.

The second data set, NYSE (N), is an extended version of NYSE (O) collected by Li et al. [2011]. It covers the period from January 1, 1985 to June 30, 2010, i.e. a total of 6,431 trading days[25]. Note that it is limited to 23 rather than 36 stocks owing to mergers and bankruptcies.

The third and fourth data sets, TSE and SP500, were collected by Borodin et al. [2004]. The former consists of 88 stocks listed on the Toronto Stock Exchange (TSE) and contains price relatives over a period of 1,259 trading days, ranging from January 4, 1994 through December 31, 1998. As for SP500, it consists of the 25 stocks in the S&P500 index with the largest market capitalisations. It ranges from January 2, 1998 to January 31, 2003 (1,276 trading days).

The fifth dataset is MSCI, which is a collection of global equity indices that constitute the MSCI World Index[26]. It is made of 24 indices that represent the equity markets of 24 countries around the world and spans of a total of 1,043 trading days, ranging from April 1, 2006 to March 31, 2010. The final dataset is

[22]Yahoo! Finance: `https://finance.yahoo.com/`; Google Finance: `https://www.google.com/finance`.

[23]The data sets from [Borodin et al., 2004] — NYSE (O), TSE, SP500 and DJIA — can be found at `http://www.cs.technion.ac.il/~rani/portfolios/`. As for the remaining data sets and their respective composition, they can be downloaded from `http://www.cais.ntu.edu.sg/~chhoi/olps`.

[24]According to Helmbold et al. [1998], the data set was originally collected by Hal Stern. The stocks are mainly those of large-cap companies listed on NYSE. However, we ignore the criteria that were used in selecting them.

[25]The data before 2007 were collected by Gábor Gelencsér `http://www.cs.bme.hu/~oti/portfolio/`, whereas Li et al. [2011] collected the remaining data from Yahoo! Finance.

[26]The constituents of this index are available at MSCI Barra (https://www.msci.com/).

Table 6.2: Summary of the performance metrics used in the evaluations.

Critetion	Performance metrics
Absolute return	Terminal wealth (S_T), Annual percentage yield (APY)
Risk	Annualised standard deviation, Maximum drawdown (MDD)
Risk-adjusted return	Annualised Sharpe ratio (SR)

the DJIA dataset [Borodin et al., 2004], consisting of the 30 components of the Dow Jones Industrial Average index. DJIA contains 507 trading days, ranging from January 14, 2001 to January 14, 2003.

The above test bed enables us to examine the behaviour of the proposed strategies under different market circumstances. For example, it covers several well-known crises in equity markets, such as the dot-com bubble from 1995 to 2000 and the subprime mortgage crisis from 2007 to 2009. The main purpose of the five stock data sets is to test the capabilities of the proposed algorithms on regional stock markets, while the MSCI index data set aims to assess their empirical performance on global indices.

6.4.2 Experimental setup and metrics

Our experimental setup is as follows. For MultiOLMAX (Algorithm 12), we denote by MultiOLMAX-S and MultiOLMAX-E the variants with SMA and EMA forecasts, respectively. To make our results comparable to those of Li et al. [2015], we adopt their hyperparameter configuration, namely $w = 5$ and $\alpha = 0.3$, across all six equity data sets. We shall assess the sensitivity of OLMAX to hyperparameter selection in Section 6.4.7.

Following the standard practice in the OLPS literature, we primarily compare different strategies in terms of terminal wealth and annualised Sharpe ratio [Sharpe, 1966]. In general, higher values of these metrics indicate better algorithms. To analyse a strategy's downside risk, we use the maximum drawdown (MDD) metric; the lower the MDD values, the less pronounced the strategy's downside. For convenience, we report all these metrics in Table 6.2.

There is somewhat of a disconnect of the metrics used for benchmarking and the notion of regret, in the sense that while the performance metrics from Table 6.2 are meaningful, the algorithms proposed in this thesis were devised in the spirit of regret. However, we refrained from measuring performance relative to regret as well because most of the OLPS techniques used for benchmarking purposes in this chapter do not optimise regret, but rather obey alternative optimisation criteria. Therefore, while we could have computed the regret of these OLPS methods and compare it against the regret of our proposed methods, we thought it would have been unfair towards these methods.

In the next subsection, we discuss one important practical issue in online portfolio selection, namely transaction costs. We defer the evaluation of the performance of OLMAX under transaction costs to Section 6.4.8.

6.4.3 Transaction costs

While our model in Section 6.2.2 is concise and easy to understand, it omits the important and unavoidable issue of *transaction costs*, which include commission fees and taxes imposed by brokers and governments, respectively[27].

Basically, there are two ways to handle the transaction-cost issue. The first is to omit transaction costs from the OLPS model in the first instance to then adopt the *proportional transaction cost* model [Blum and Kalai, 1999]. The second is to directly integrate the costs into the model [Györfi and Vajda, 2008]. Since the former approach is the most common in the OLPS literature, we shall follow it here. Specifically, at the beginning of period t, the portfolio manager rebalances to a new portfolio $\mathbf{b}_t$ from the last close-price adjusted portfolio $\widehat{\mathbf{b}}_{t-1}$, each component of which is calculated as $\widehat{b}_{t-1,i} = \frac{b_{t-1,i} \times x_{t-1,i}}{\mathbf{b}_{t-1}^{\mathrm{T}} \mathbf{x}_{t-1}}$. Letting $\gamma \in (0,1)$ be the transaction-cost rate, this rebalancing step incurs a transaction cost of $\frac{\gamma}{2} \sum_{i=1}^{m} |b_{t,i} - \widehat{b}_{t-1,i}|$, with the initial portfolio being set to the zero vector. Thus, the wealth after T periods can be expressed as

$$S_T^{\gamma} = S_0 \prod_{t=1}^{T} \left[(\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t) \times \left(1 - \frac{\gamma}{2} \sum_{i=1}^{m} |b_{t,i} - \widehat{b}_{t-1,i}| \right) \right]. \tag{6.19}$$

[27] Besides these, other factors such as bid-ask spreads contribute to transaction costs.

6.4.4 Comparison approaches

We shall compare the proposed algorithms against a number of benchmarks and representative strategies enumerated below, all of which provide extensive empirical evaluations in their respective studies. All parameters are set to the same values used in the corresponding papers[28]. Because this chapter has an empirical focus, we omit algorithms that place an emphasis on theoretical analysis and lack thorough experimentation.

1. Market: Market strategy, that is, the uniform buy-and-hold (BAH) strategy;

2. Best stock: Stock with the best performance in hindsight throughout the T-period horizon;

3. BCRP: Best constant rebalanced portfolio strategy in hindsight [Cover, 1991];

4. UP: Cover's universal portfolios, based on the implementation of Kalai and Vempala [2002], in which the parameters are such that $\delta_0 = 0.004$, $\delta = 0.005$, $m = 100$ and $S = 500$;

5. EG: Exponentiated gradient algorithm with the best learning rate $\eta = 0.05$, as suggested by Helmbold et al. [1998];

6. ONS: Online Newton step with the parameters suggested by Agarwal et al. [2006], i.e. $\eta = 0$, $\beta = 1$ and $\gamma = \frac{1}{8}$;

7. Anticor: BAH_{30}(Anticor), i.e. uniform buy-and-hold investment on the algorithms Anticor_w with $w = 2, 3, \ldots, 30$, which achieves the best performance among the three solutions proposed by Borodin et al. [2004];

8. B^{K}: Non-parametric kernel-based moving window strategy with $W = 5$, $L = 10$ and threshold $c = 1.0$, which exhibits the best empirical performance according to Györfi et al. [2006];

[28] We could tune these parameters to improve performance, but this is beyond the scope of this thesis.

9. B^{NN}: Non-parametric nearest-neighbour strategy with parameter values $W = 5$, $L = 10$ and $p_l = 0.02 + 0.5\frac{l-1}{L-1}$, as Györfi et al. [2008] suggested;

10. CORN: Correlation-driven non-parametric learning approaches [Li et al., 2010] with parameters $W = 5$, $P = 1$ and $\rho = 0.1$.

11. PAMR: Passive-aggressive mean reversion [Li et al., 2012] with parameter $\epsilon = 0.5$;

12. CWMR: Confidence-weighted mean reversion [Li et al., 2011] algorithm (exact version) with $\epsilon = 0.5$;

13. OLMAR-S: Online moving average reversion [Li et al., 2015] with SMA price-relative forecasts, and parameters $\epsilon = 10$ and $w = 5$;

14. OLMAR-E: Online moving average reversion [Li et al., 2015] with EMA price-relative forecasts, and parameters $\epsilon = 10$ and $\alpha = 0.3$.

6.4.5 Experimental results – terminal wealth

Table 6.3 reports the terminal wealth achieved by various approaches on the six equity data sets listed in Table 6.1. From this perspective, the OLMAX algorithms are among the top two performers across five data sets (the exception being MSCI), and they achieve the best performance on four of them. On the well-known benchmark NYSE (O) data set, OLMAX significantly outperforms the state of the art, and likewise on the TSE and SP500 data sets. Although most existing algorithms other than Anticor and OLMAR-S perform badly on the DJIA data set, OLMAX achieves the highest final wealth, which further motivates the main idea behind the framework (see Section 6.3.1). Besides, the use of exponential rather than simple moving averages provides a significant performance boost on the NYSE (O) and NYSE (N) data sets.

Table 6.3: Terminal wealth achieved by various OLPS strategies on the six equity data sets from Table 6.1. The top two results on each data set are highlighted in **bold**.

Methods	NYSE (O)	NYSE (N)	DJIA	TSE	SP500	MSCI
Market	14.50	18.06	0.76	1.61	1.34	0.91
Best stock	54.14	83.51	1.19	6.28	3.78	1.50
BCRP	250.60	120.32	1.24	6.78	4.07	1.51
UP	26.68	31.49	0.81	1.60	1.62	0.92
EG	27.09	31.00	0.81	1.59	1.63	0.93
ONS	109.19	21.59	1.53	1.62	3.34	0.86
B^K	1.08E+09	4.64E+03	0.68	1.62	2.24	2.64
B^{NN}	3.35E+11	6.80E+04	0.88	2.27	3.07	13.47
CORN	1.48E+13	5.37E+05	0.84	3.56	6.35	**26.10**
Anticor	2.41E+08	6.21E+06	**2.29**	39.36	5.89	3.22
PAMR	5.14E+15	1.25E+06	0.68	264.86	5.09	15.23
CWMR	6.49E+15	1.41E+06	0.68	332.62	5.90	17.28
OLMAR-S	3.68E+16	2.54E+08	2.12	424.80	5.83	16.39
OLMAR-E	**1.09E+18**	**5.10E+08**	1.20	**678.44**	8.63	**21.21**
SingleOLMAX	4.09E+15	5.09E+05	0.59	**1.56E+03**	8.32	7.30
MultiOLMAX-S	4.47E+16	3.86E+08	**2.51**	75.88	**15.30**	11.19
MultiOLMAX-E	**2.54E+18**	**4.80E+08**	1.22	313.09	**14.31**	13.44

In order to confirm that the above OLMAX results are not a statistical fluke, we compute the alphas and betas from the regression

$$r_t^{\text{OLMAX}} - r_f = \alpha + \beta(r_t^{\text{MKT}} - r_f) + \varepsilon_t, \tag{6.20}$$

where r_t^{OLMAX} denotes the return of an OLMAX portfolio at time t, r_f is the risk-free rate, and r_t^{MKT} represents a passive exposure to the stock market, proxied by the return on the BAH strategy. Table 6.4 displays the t-statistics and p-values of the estimated alphas for the MultiOLMAX-S variant. From this table, we see that the probability that the outstanding MultiOLMAX-S performance is due to luck is at most 1.31%. Though we omitted them here, the findings are similar for other OLMAX variants. Thus, we can claim with almost absolute certainty that OLMAX outperforms the state of the art on the equity data sets.

Table 6.4: Statistical tests of the performance achieved by the MultiOLMAX-S algorithm on the stock data sets. The acronym MER stands for 'mean excess return'.

Stat. Attr.	NYSE (O)	NYSE (N)	DJIA	TSE	SP500	MSCI
Size	5651	6431	507	1259	1276	1043
MER (MultiOLMAX-S)	0.0073	0.0035	0.0022	0.0050	0.0026	0.0025
MER (Market)	0.0005	0.0005	-0.0004	0.0004	0.0003	0.0000
Winning Ratio	57.95%	55.19%	53.77%	55.15%	52.51%	55.88%
α	0.0068	0.0031	0.0029	0.0046	0.0023	0.0026
β	1.2709	1.1407	1.2540	1.6497	1.2541	1.1921
t-stat	15.1105	7.5234	2.4888	2.7452	2.9378	5.1207
p-value	0.0000	0.0000	0.0131	0.0061	0.0034	0.0000

6.4.6 Experimental results – risk-adjusted returns

We now evaluate the volatility and maximum drawdown of different OLPS strategies, as well as their respective risk-adjusted returns as measured by the annualised Sharpe ratio [Sharpe, 1966]. Figure 6.3 displays these metrics for the six equity data sets. In addition to the proposed MultiOLMAX-S algorithm, we plot two benchmarks (Market and BCRP) and five state-of-the-art algorithms (Anticor, CORN, PAMR, CWMR and OLMAR[29]).

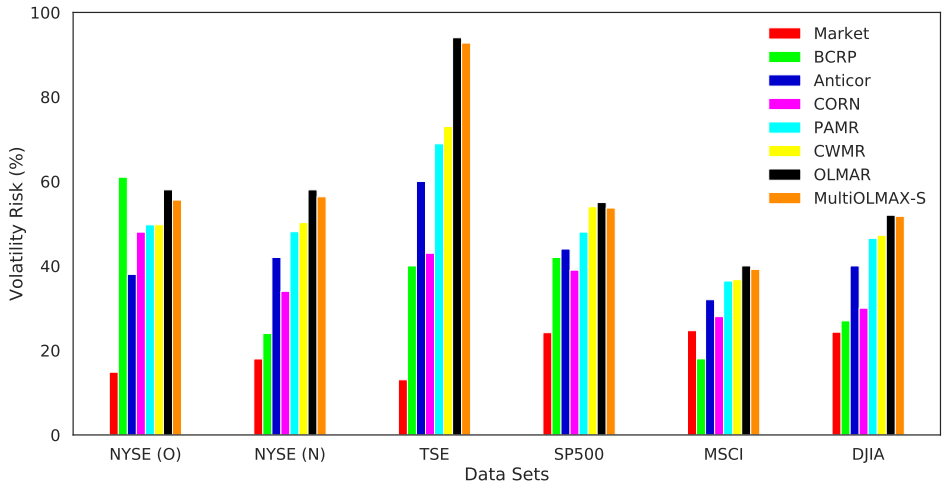
In the previous paragraph, we saw that MultiOLMAX-S achieves the highest cumulative return on the majority of equity data sets. However, high return is typically associated with high risk. This is corroborated by Figures 6.3(a) and (b), which show that the proposed method is one the riskiest in terms of volatility and maximum drawdown, respectively.

However, in spite of its high risk, MultiOLMAX-S yields a highly competitive Sharpe ratio, as illustrated by Figure 6.3(c). This encouraging finding demonstrates that the proposed method is able to strike a good balance between risk and return, even though we do not explicitly account for risk in its formulation.

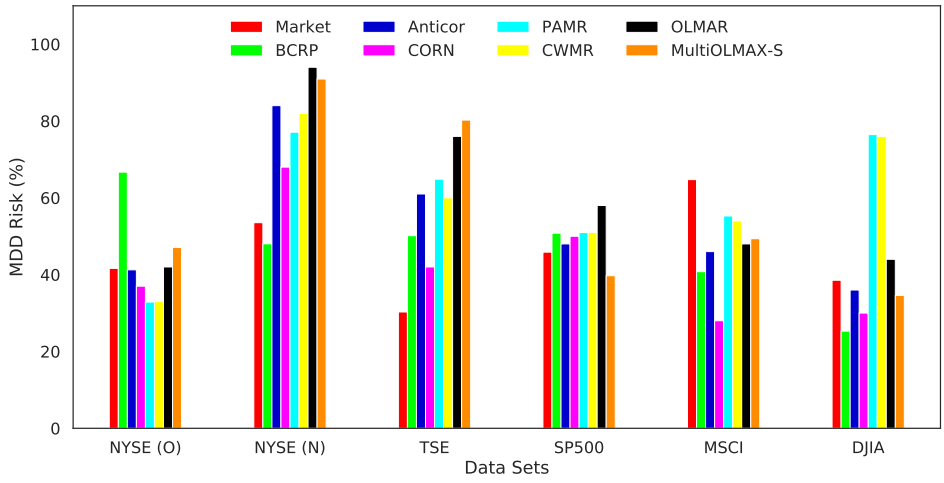
[29]Henceforth, the acronym 'OLMAR' will refer to the OLMAR-S strategy, i.e. the variant with SMA forecasts of price relatives.

Figure 6.3: Risk and risk-adjusted performance of various OLPS strategies on the six stock data sets. In each diagram, the rightmost bar represents the result of our proposed strategy.

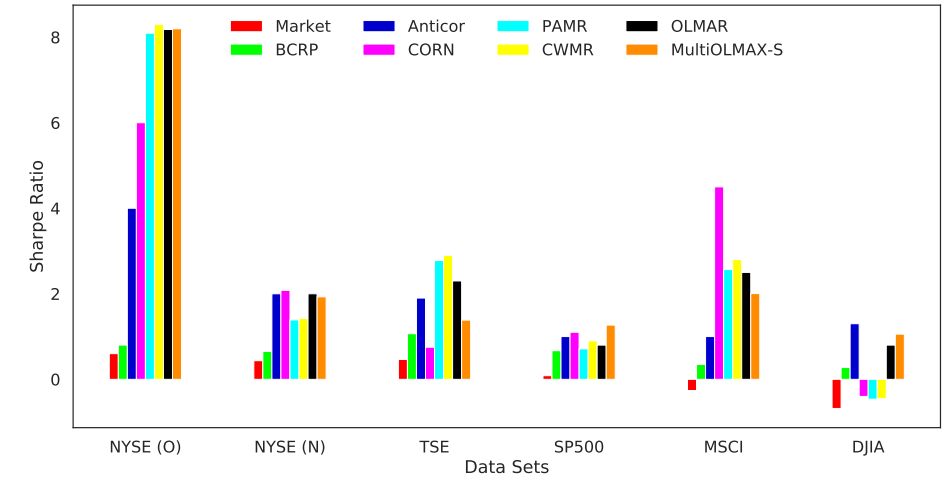
(a) Volatility Risk



(b) Drawdown Risk



(c) Sharpe Ratio



6.4.7 Hyperparameter sensitivity

Next we assess the sensitivity of OLMAX with respect to its hyperparameters, namely the lookback window w in the MultiOLMAX-S variant and the smoothing factor α in its MultiOLMAX-E counterpart. These are depicted in Figures 6.4 and 6.5, respectively.

From Figure 6.4, we see that as w increases, the performance deteriorates on the NYSE (O), NYSE (N) and, following an initial spike, the MSCI data sets. As for TSE and SP500, MultiOLMAX-S attains the highest terminal wealth for $w = 35$ and $w = 65$, respectively, while on DJIA we observe a multi-modal pattern. It is remarkable that for most choices of w, the total wealth generated by MultiOLMAX-S exceeds that of the Market and BCRP benchmarks by a significant margin.

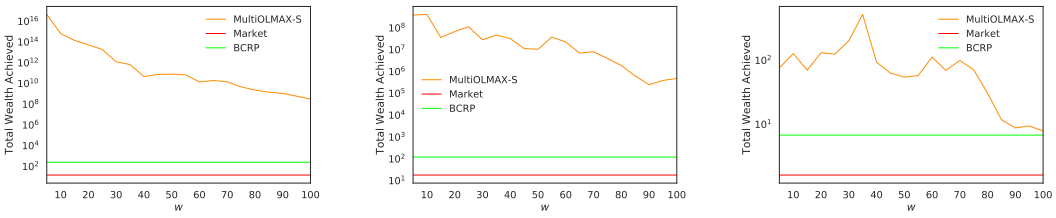
As for the MultiOLMAX-E variant, we observe in Figure 6.5 that on most data sets, terminal wealth is consistently high across a wide range of values for α, the two extreme endpoints $\alpha = 0$ and $\alpha = 1$ being the exception. We can see why this is so, as follows. If $\alpha = 1$, then all expected price relatives are always 1 and MultiOLMAX-E outputs $\mathbf{b}_{t+1} = \mathbf{b}_t$ on each trading period t. Since, by convention, we use the equally-weighted portfolio $\mathbf{1}/m$ for initialisation purposes, this amounts to rebalancing the MultiOLMAX-E portfolio to the equally-weighted one on each trading period, which explains the poor performance. On the other hand, when $\alpha = 0$, the expected vector of price relatives equals $\widetilde{\mathbf{x}}_{t+1} = \frac{1}{\bigotimes_{\tau=1}^{t} \mathbf{x}_\tau}$, i.e. it inversely relates to the entire history of price relatives, and should thereby not be expected to produce excellent results.

6.4.8 Practical performance under transaction costs

To evaluate the practical applicability of the OLMAX framework, we analyse its robustness to proportional transaction fees [Borodin et al., 2004]. Figure 6.6 illustrates the final wealth achieved by the MultiOLMAX-S method as a function of the transaction-cost rate γ, along with the corresponding results for the Market and BCRP benchmarks.

Figure 6.4: Sensitivity of MultiOLMAX-S with respect to its lookback window w.

(a) NYSE (O) **(b)** NYSE (N) **(c)** TSE



(d) SP500 **(e)** MSCI **(f)** DJIA

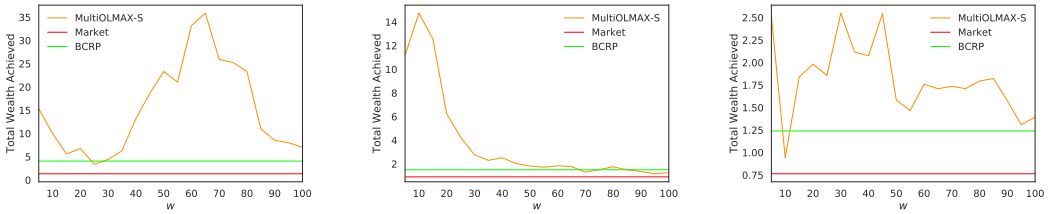
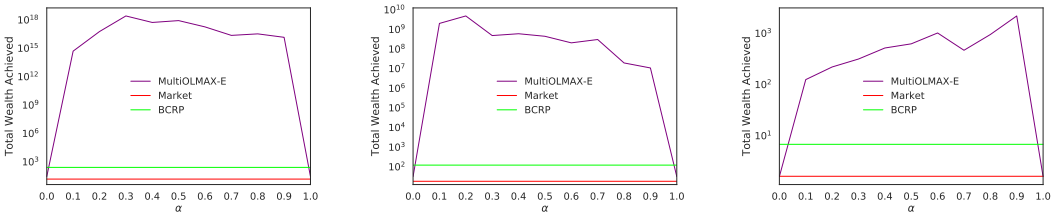


Figure 6.5: Sensitivity of MultiOLMAX-E with respect to its smoothing factor α.

(a) NYSE (O) **(b)** NYSE (N) **(c)** TSE



(d) SP500 **(e)** MSCI **(f)** DJIA

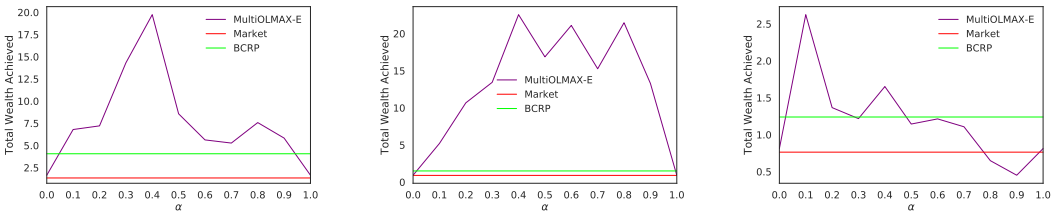
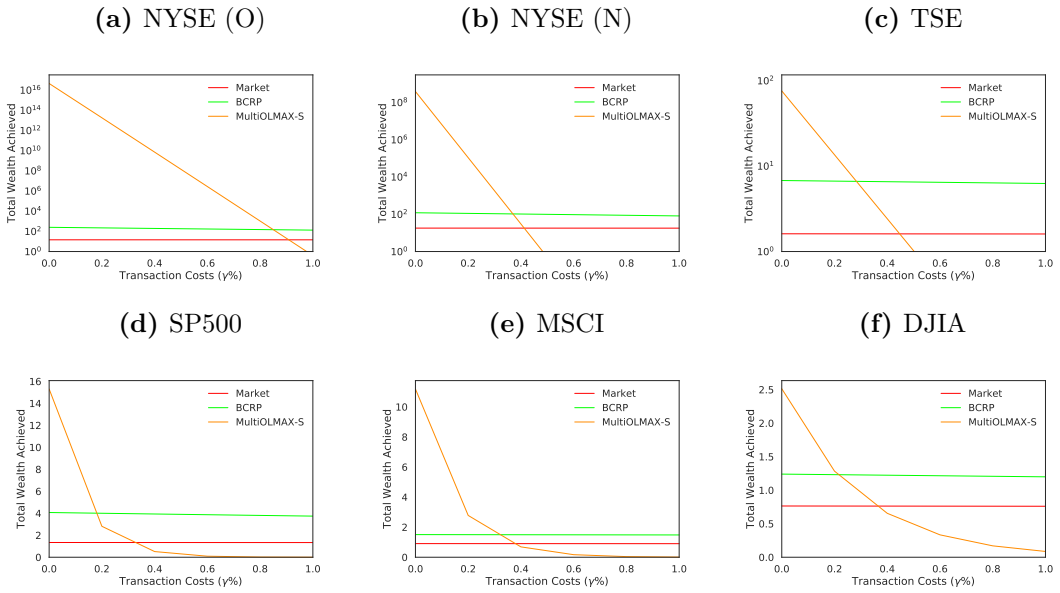


Figure 6.6: Scalability of the proposed strategy with respect to the transaction-cost rate (γ).



On one hand, Figure 6.6 clearly shows that MultiOLMAX-S can withstand reasonable transaction costs, as it often has high break-even capability relative to the market. On the other hand, it can outperform the benchmarks under various transaction-cost assumptions. Consequently, MultiOLMAX-S exhibits outstanding performance even when trading is not frictionless, which provides further support to its practical applicability.

6.5 Adaptive Online Mean Reversion

6.5.1 Motivation

Despite the effectiveness of the proposed OLMAX framework, one cannot help but notice that the corresponding experiments suffer from *selection bias*, in the sense that they fail to cover asset classes other than equities. The same caveat holds for the CWMR, PAMR and OLMAR strategies, among others. The purpose of this subsection is to demonstrate that, in a portfolio that is broadly diversified in terms of asset classes, the aforementioned algorithms no longer exhibit a stellar performance after adding arguably reasonable fees of 0.1% per transaction (corresponding to \$1 for every \$1,000 of bought/sold asset units). Additionally, we shall perform

a sensitivity analysis suggesting that this practical limitation is related to the data-snooping bias from which these strategies suffer.

The first issue that we face concerns the best way to form a diversified portfolio. Since this lies outside the scope of this thesis, we shall simply follow the approach of David Swensen[30], who suggests a portfolio composed of six different asset classes, as follows: US equities (30% weighting), international equities (15%), emerging markets (10%), Treasury inflation-protected securities (TIPS) (15%), US treasuries (15%) and real-estate investment trusts (REITs) (15%) [Swensen, 2005]. Since we are not concerned with portfolio construction, we will use a set of highly liquid exchange-traded funds (ETFs) as vehicles for these asset classes. These are listed in Table 6.5.

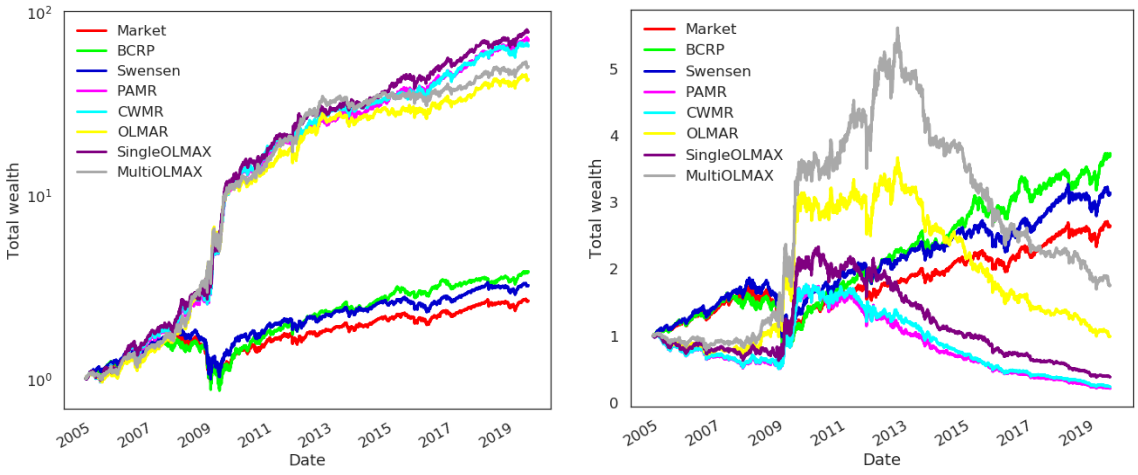
Table 6.5: Exchange-traded fund (ETF) dataset used in our empirical evaluation. Daily adjusted closing prices for these ETFs were obtained from Yahoo! Finance, for the period from January 1, 2005 until June 3, 2019 inclusive.

Ticker	Name	Asset Class	Region	Number of Holdings
VTI	Vanguard Total Stock Market ETF	Equity	US	3629
EFA	iShares MSCI EAFE ETF	Equity	Europe, Australia, Asia & the Far East	937
EEM	iShares MSCI Emerging Markets ETF	Equity	Emerging markets	857
TLT	iShares 20+ Year Treasury Bond ETF	Fixed Income	US	30
TIP	iShares TIPS Bond ETF	Fixed Income	US	38
VNQ	Vanguard Real Estate ETF	Equity	US	184

Figure 6.7 clearly illustrates the lack of robustness with respect to transaction costs of PAMR, CWMR, OLMAR, SingleOLMAX and MultiOLMAX[31]. Under the absence of transaction costs, these methods exhibit decent performance in terms of cumulative wealth, and even exceed the benchmarks (including Swensen's recommended portfolio) by a highly significant margin. However, the picture changes drastically when one introduces fees of 0.1% per transaction, and none of these algorithms manages to beat any of the benchmarks. Worse, they all stagnate around 2010 and their performance starts deteriorating in 2011 or a couple of years thereafter.

[30]David Swensen has been the chief investment officer of Yale University's endowment fund since 1985, and is highly regarded in the asset management community.

[31]Henceforth, the acronym 'MultiOLMAX' will be used as a shorthand for the MultiOLMAX-S strategy, whose price-relative forecasts are based on simple moving averages.

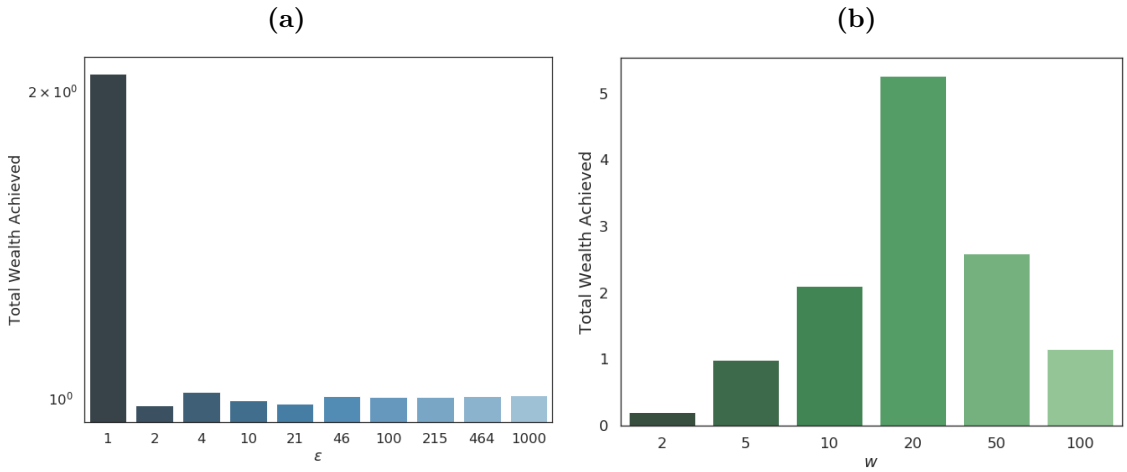


(a) Without fees **(b)** Subject to fees of 0.1%

Figure 6.7: Trends of cumulative wealth achieved by various strategies during the entire trading periods associated with the ETF data set (see Table 6.5). Panel **(a)** illustrates these trends under the assumption of no transaction costs, whereas panel **(b)** depicts them net of fees of 0.1% per transaction.

A deeper analysis reveals that the underperformance documented above may be due to data-snooping bias. Let us focus on OLMAR, for the sake of argument. In Figure 6.8, we see with hindsight that the optimal terminal wealth does not occur at the default values $\epsilon = 10$ and $w = 5$ of the underlying parameters, where, as a reminder, ϵ is the reversion threshold and w the lookback window used in the computation of the price-relative forecasts. For instance, if we keep ϵ fixed at 10, the optimal window yielding the highest final wealth is $w = 20$. Once we implement the OLMAR algorithm with these parameter values optimised with the benefit of hindsight, we are able to improve its *net* performance (i.e. net of transaction fees) beyond that of the benchmarks, as illustrated in Figure 6.9.

Figure 6.8: Parameter sensitivity of OLMAR on the ETF data set, with respect to **(a)** ϵ with fixed w ($w = 5$) and **(b)** w with fixed ϵ ($\epsilon = 10$).



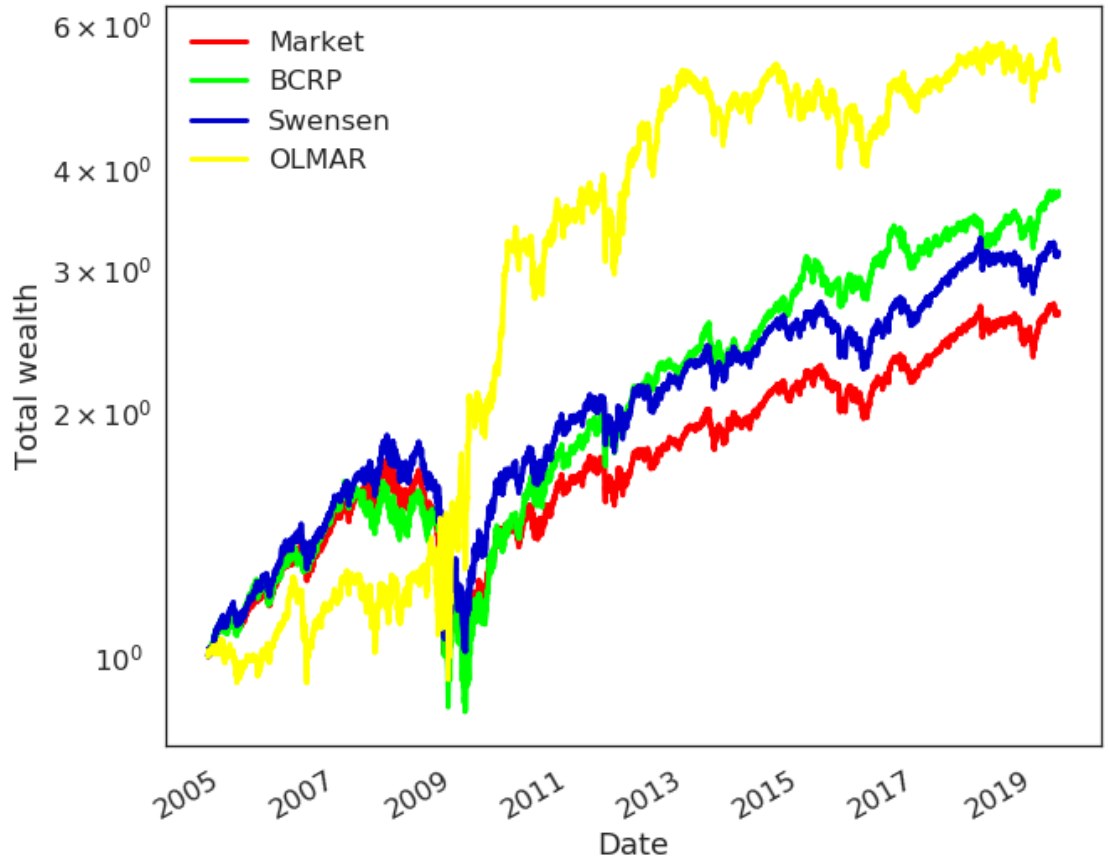


Figure 6.9: Cumulative wealth achieved by OLMAR optimised in hindsight (with parameters $\epsilon = 10$ and $w = 20$) vs benchmarks on the ETF data set, subject to transaction fees of 0.1%.

6.5.2 Proposed algorithms

Armed with the insights from Figures 6.8 and 6.9, we turn now to explore *adaptive* variants of PAMR and OLMAR that are governed by no and a smaller number of hyperparameters, respectively. To this end, we cast the underlying optimisation problems into a, roughly speaking, equivalent OGD-criterion representation (see Eq. (5.35)) whose learning rate can be made adaptive via the application of the MAPGRAD algorithm (i.e. Algorithm 8).

To start with, let us consider the PAMR strategy. As proposed in [Li et al., 2012], the standard variant is formulated as the following constrained optimisation problem:

$$\min_{\mathbf{b}\in\Delta_m} \frac{1}{2}\|\mathbf{b}-\mathbf{b}_t\|_2^2 \qquad \text{s.t.} \qquad \mathbf{b}\cdot\mathbf{x}_t \leq \epsilon, \tag{6.21}$$

for $t \in [T-1]$, with the convention $\mathbf{b}_1 = \mathbf{1}/m$. Instead of this problem, we will consider the following proximal problem:

$$\min_{\mathbf{b}\in\Delta_m} \left\{\eta(\mathbf{b}\cdot\mathbf{x}_t) + \frac{1}{2}\|\mathbf{b}-\mathbf{b}_t\|_2^2\right\} = \min_{\mathbf{b}\in\Delta_m} \left\{\mathbf{b}\cdot\mathbf{x}_t + \frac{1}{2\eta}\|\mathbf{b}-\mathbf{b}_t\|_2^2\right\}, \tag{6.22}$$

for some $\eta > 0$. Roughly speaking, the PAMR problem (6.21) and the proximal problem (6.22) have the same solutions for appropriate choices of the parameters ϵ and η. However, the formulation (6.22) has the form of the OGD criterion in Eq. (5.35), meaning the portfolio weights that solve (6.22) satisfy the OGD update rule (5.26):

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}(\mathbf{b}_t - \eta\mathbf{x}_t), \tag{6.23}$$

where, as a reminder, $\Pi_{\Delta_m}(\mathbf{b})$ signifies the point in Δ_m that is closest to $\mathbf{b}$. In order to marginalise the dependence on η of the portfolio updates in Eq. (6.23), we employ the MAPGRAD method outlined in Algorithm 8. The latter prescribes the data-dependent learning-rate schedule $\{\eta_t = m/\|\sum_{\tau=1}^t \mathbf{x}_\tau\|_2^2\}_{t=1}^{T-1}$ which, when plugged into Eq. (6.23), yields the strategy

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}\left(\mathbf{b}_t - \frac{m}{\|\sum_{\tau=1}^t \mathbf{x}_\tau\|_2^2}\,\mathbf{x}_t\right), \qquad t \in [T-1], \tag{6.24}$$

Algorithm 13 AdaPAMR: Adaptive Passive-Aggressive Mean Reversion

Initialisation: initial portfolio $\mathbf{b}_1 = \frac{\mathbf{1}}{m}$, initial wealth $S_0 = 1$, initial gradient sum $\boldsymbol{\theta}_0 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ **do**
  observe stock price relatives $\mathbf{x}_t$
  update wealth: $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}} \mathbf{x}_t)$
  if $t < T$ **then**
    update the gradient sum: $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mathbf{x}_t$
    compute the learning rate:

$$\eta_t = \frac{m}{\|\boldsymbol{\theta}_t\|_2^2}$$

    update the portfolio:

$$\mathbf{b}_{t+1} = \mathbf{b}_t - \eta_t \mathbf{x}_t$$

    normalise the portfolio:

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}(\mathbf{b}_{t+1}) = \operatorname*{arg\,min}_{\mathbf{b} \in \Delta_m} \|\mathbf{b} - \mathbf{b}_{t+1}\|_2^2$$

  end if
end for

subject to the initialisation condition $\mathbf{b}_1 = \mathbf{1}/m$. For obvious reasons, we shall refer to the latter as *adaptive passive-aggressive mean reversion* strategy, or AdaPAMR for short. The procedure is outlined in Algorithm 13.

The derivation of the adaptive OLMAR (AdaOLMAR) algorithm is similar, so our exposition thereof will be brief. We replace the OLMAR optimisation problem

$$\min_{\mathbf{b} \in \Delta_m} \frac{1}{2}\|\mathbf{b} - \mathbf{b}_t\|_2^2 \qquad \text{s.t.} \qquad \mathbf{b} \cdot \widetilde{\mathbf{x}}_{t+1} \geq \epsilon \tag{6.25}$$

with the proximal problem[32]

$$\min_{\mathbf{b} \in \Delta_m} \left\{ -\mathbf{b} \cdot \widetilde{\mathbf{x}}_{t+1} + \frac{1}{2\eta}\|\mathbf{b} - \mathbf{b}_t\|_2^2 \right\}, \tag{6.26}$$

whose unique solution is given by

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}(\mathbf{b}_t + \eta\, \widetilde{\mathbf{x}}_{t+1}). \tag{6.27}$$

[32]Note the negative sign preceding the loss term $\mathbf{b} \cdot \widetilde{\mathbf{x}}_{t+1}$. This is required because OLMAR's key constraint is $\mathbf{b} \cdot \widetilde{\mathbf{x}}_{t+1} \geq \epsilon$, i.e. it has a *greater-or-equal* sign, unlike PAMR's key constraint $\mathbf{b} \cdot \mathbf{x}_t \leq \epsilon$. However, it can easily be rewritten in this form, by multiplying both of its sides times -1, which gives $-\mathbf{b} \cdot \widetilde{\mathbf{x}}_{t+1} \leq \delta$, where $\delta \equiv -\epsilon$.

As before, we can automatically adjust the learning rate η by means of the MAPGRAD algorithm. Doing so produces the AdaOLMAR portfolio weights:

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}\left(\mathbf{b}_t + \frac{m}{\|\sum_{\tau=1}^{t}\widetilde{\mathbf{x}}_{\tau+1}\|_2^2}\,\widetilde{\mathbf{x}}_{t+1}\right), \qquad t \in [T-1]. \tag{6.28}$$

The entire AdaOLMAR strategy is presented in Algorithm 14. As is the case for OLMAR, there are two variants depending on whether one wants to use SMA or EMA-based price-relative forecasts.

Algorithm 14 AdaOLMAR: Adaptive Online Moving Average Reversion

Input: Window size $w \geq 2$ (SMA variant), smoothing factor $0 < \alpha < 1$ (EMA variant)
Initialisation: initial portfolio $\mathbf{b}_1 = \frac{\mathbf{1}}{m}$, initial wealth $S_0 = 1$, initial gradient sum $\boldsymbol{\theta}_0 = \mathbf{0}$
for $t = 1, 2, \ldots, T$ **do**
  observe stock price relatives $\mathbf{x}_t$
  update wealth: $S_t = S_{t-1} \times (\mathbf{b}_t^{\mathrm{T}}\mathbf{x}_t)$
  if $t < T$ **then**
    predict next price-relative vector:

$$\widetilde{\mathbf{x}}_{t+1} = \begin{cases} \frac{1}{w}\left(\mathbf{1} + \frac{1}{\mathbf{x}_t} + \ldots + \frac{1}{\bigotimes_{h=0}^{w-2}\mathbf{x}_{t-h}}\right) & \text{(SMA variant)} \\ \alpha + (1-\alpha)\frac{\widetilde{\mathbf{x}}_t}{\mathbf{x}_t} & \text{(EMA variant)} \end{cases}$$

    update the gradient sum: $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \widetilde{\mathbf{x}}_{t+1}$
    compute the learning rate:

$$\eta_t = \frac{m}{\|\boldsymbol{\theta}_t\|_2^2}$$

    update the portfolio:

$$\mathbf{b}_{t+1} = \mathbf{b}_t + \eta_t\,\widetilde{\mathbf{x}}_{t+1}$$

    normalise the portfolio:

$$\mathbf{b}_{t+1} = \Pi_{\Delta_m}(\mathbf{b}_{t+1}) = \operatorname*{arg\,min}_{\mathbf{b}\in\Delta_m} \|\mathbf{b} - \mathbf{b}_{t+1}\|_2^2$$

  end if
end for

6.5.3 Empirical evaluation

We now turn to implement the AdaPAMR and AdaOLMAR algorithms on the ETFs mentioned in Table 6.5. Figure 6.10 illustrates their respective cumulative

wealth net of a transaction fee of 0.1%, along with those achieved by the three benchmarks (Market, BCRP and the Swensen portfolio) and the non-adaptive reversion strategies (CWMR, PAMR, OLMAR, SingleOLMAX and MultiOLMAX). Furthermore, in order to empirically motivate the benefits of our approach, we include analogous variants of AdaPAMR and AdaOLMAR in which the learning rate is fixed to a schedule of $1/\sqrt{t}$, as in [Zinkevich, 2003a]. Perhaps at the expense of an abuse of terminology, we named these 'AdaPAMR with $1/\sqrt{t}$ annealing' and 'AdaOLMAR with $1/\sqrt{t}$ annealing', respectively.

In comparison to the benchmarks, both the AdaPAMR and AdaOLMAR methods would have realised big gains at the end of the subprime mortgage crisis and, unlike their non-adaptive counterparts, managed to maintain that positive momentum all along thereafter, especially AdaPAMR. This explains the large discrepancy between their respective terminal wealths and that of all the other strategies plotted in the chart. Additionally, we notice that the variants of AdaPAMR and AdaOLMAR with Zinkevich's learning-rate schedule of $1/\sqrt{t}$ barely beat the Swensen portfolio, making them unattractive choices.

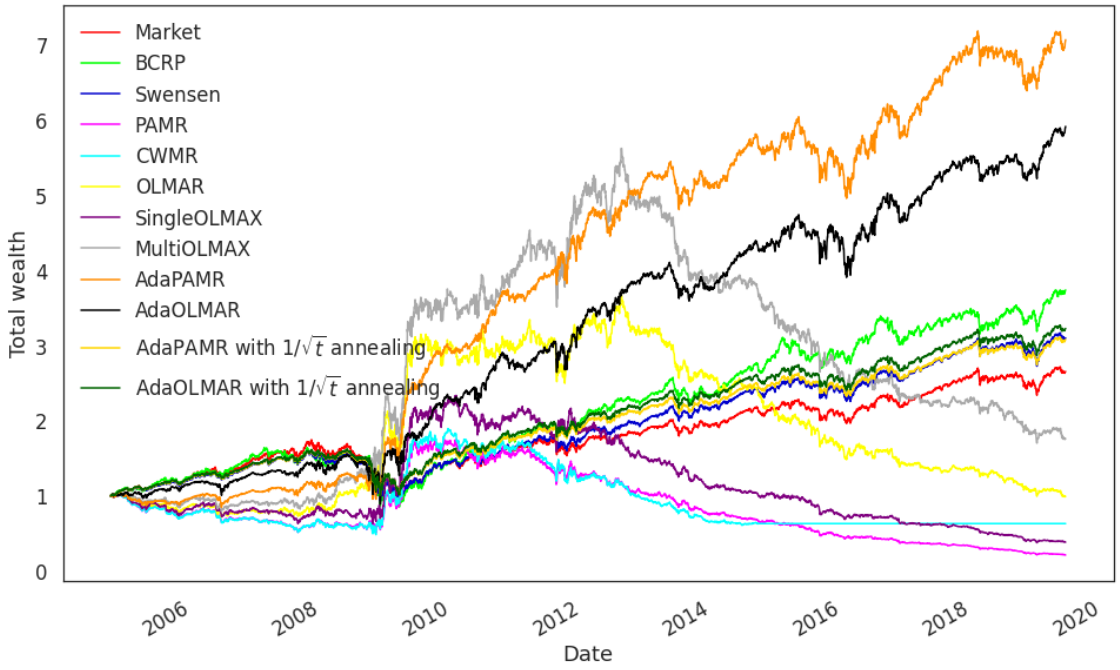


Figure 6.10: Cumulative wealth net of 0.1% fees achieved by various strategies during the entire trading periods associated with the ETF data set (see Table 6.5).

Table 6.6: Performance metrics achieved by various OLPS algorithms on the ETF data set from Table 6.5, subject to transaction fees of 0.1%. The top two results are highlighted in **bold**.

Algorithm	APY (%)	Volatility Risk (%)	MDD Risk (%)	Sharpe Ratio
Market	6.87	14.90	45.33	0.45
Best stock	8.83	18.47	55.45	0.46
BCRP	9.39	18.81	47.97	0.48
Swensen	8.12	17.33	47.31	0.45
UP	7.74	14.61	42.04	0.51
EG	7.83	14.61	41.72	0.52
ONS	10.14	15.22	**32.64**	0.63
PAMR	-9.95	24.54	87.74	-0.43
CWMR	-9.29	24.75	86.54	-0.39
OLMAR	-0.04	24.87	73.10	-0.00
SingleOLMAX	-6.29	24.61	83.81	-0.26
MultiOLMAX-S	3.93	24.62	69.71	0.16
MultiOLMAX-E	5.84	24.84	66.34	0.23
AdaPAMR with $1/\sqrt{t}$ annealing	8.50	**12.95**	37.07	**0.66**
AdaOLMAR with $1/\sqrt{t}$ annealing	8.93	**13.79**	38.16	0.65
AdaPAMR	**14.24**	18.87	**26.79**	**0.71**
AdaOLMAR	**12.86**	21.13	42.81	0.57

Table 6.6 reports the performance metrics of all the aforementioned algorithms on the ETF data set, in addition to the remaining ones enumerated in Section 6.4.4. As one would expect from Figure 6.10, AdaPAMR and AdaOLMAR dominate in terms of annual percentage yield. Furthermore, AdaPAMR is the least risky from a drawdown perspective, and exhibits the highest Sharpe ratio.

We end this chapter by recommending AdaPAMR as a general-purpose online portfolio selection algorithm that offers transaction-cost robustness and versatility to handle different asset classes and market regimes thanks to its hyperparameter-free nature.

Part III

Conclusions and Extensions

7
Summary Conclusions

Contents

This thesis has contributed novel modelling techniques for sequential data, with a focus on online portfolio selection. The contributions, set out in the three chapters comprising Part II, are unified under the title 'Bayesian Online Learning for Portfolio Management', and split into three approximately-homogeneous contribution chapters. Chapter 4 discussed generalisations of and probabilistic inference in a particular class of online model known as 'online passive-aggressive learning', in which a weight vector is passively maintained or aggressively updated from round to round depending on whether or not it correctly predicted the most recent example. Chapter 5 dealt with the topic of online convex optimisation in dynamic environments, and by incorporating insight from Chapter 4, contributed two novel online and data-dependent adaptation methods for tuning the learning rate in online gradient descent, arguably the most popular algorithm in this context. Finally, in Chapter 6, we demonstrated the superiority of our proposed methods

over existing ones in online portfolio selection experiments involving real-world data from the equity, fixed income and real-estate markets.

This chapter seeks to conclude the thesis by drawing together summary results and conclusions of the contributions set out in Part II.

7.1 Extending Passive-Aggressive Learning

Chapter 4 formulated some generalisations and a Bayesian treatment of online passive-aggressive (PA) algorithms, which have been used in various fields including finance [Li et al., 2012, 2015], recommendation systems [Blondel et al., 2014] and natural language processing [Shi and Zhu, 2017]. The generic PA update rule (as per Chapter 2, Eq. (2.9)) is intuitive, straightforward to implement and universally applicable to online classification, regression and uniclass tasks. However, it suffers from some drawbacks, namely *i*) it is only valid for the ϵ-insensitive loss function, *ii*) it does not provide any guidelines as to how to set the underlying hyperparameters, and *iii*) it is a point estimate, thereby failing to capture model and prediction uncertainty.

To address the first of these issues, we derived a generalised passive-aggressive learning framework, by replacing the ϵ-insensitive loss in the generic PA learning problem's constraint (i.e. the constraint in Eq. (2.8)) with a general loss function. By approximating this general loss function with its first-order Taylor polynomial around the current weight vector, we are able to apply the same solution methods used to arrive at the PA weight updates. The algorithms so defined overcome the specialisation of the original PA framework to the ϵ-insensitive loss while maintaining elegance and scalability. Moreover, the derivation is didactically useful in the sense that it provides inspiration and insight for other algorithms described later in the thesis, including the passive-aggressive convex optimisation algorithm 9.

A Bayesian treatment of PA learning was also discussed, motivated by a desire to circumvent the remaining aforementioned shortcomings, i.e. points *ii*) and *iii*), in addition to the framework's inability to take into account how the data are distributed. Inspired by the pseudo-likelihood and data-augmentation ideas from

[Polson and Scott, 2011], the mixture representation (4.33) is the key ingredient, since it significantly reduces the complexity of inference by bringing Bayesian tools for Gaussian linear models to bear on PA algorithms. We then contributed a novel online scheme for variational posterior inference in an augmented variable space, that additionally offers the benefit of automatic hyperparameter tuning. To make robust predictions, the approximate posterior of model weights, rather than point estimates as in traditional PA methods, is used. Once the approximate posterior is obtained from a given data set, we can regard it as the model prior when dealing with the arrival of new data, which makes the model suitable for online scenarios.

7.2 Adaptive Gradient Methods for Dynamic Online Optimisation

Online convex optimisation is a prevalent topic in machine learning and its application in dynamic environments has recently received considerable interest in the literature. Whilst the online gradient descent algorithm is appealing in this context due to its numerous advantages, adjusting its learning-rate parameter is an important unresolved problem which requires tuning in practice. Previous work has relied on theoretical learning-rate schedules, under the assumption that certain parameters of the loss function such as smoothness or strong convexity constants are known. However, in practice, such parameters are unknown and the burden of specification rests upon the practitioner.

We presented a novel method for dynamically updating the learning-rate parameter according to the gradients received so far, which we believe provides a significant contribution to the long-standing problem of tuning this parameter. Unlike the schemes previously used in the literature, ours is completely hyperparameter-free and can thus be readily deployed without any manual intervention. Moreover, its applicability extends beyond the field of online convex optimisation, in particular to neural-network training (see Section 8.2).

The aforementioned method was derived in three steps. First, we determined in Section 5.5.1 which optimisation problem online gradient descent implicitly

solves. Second, we provided a maximum a posteriori interpretation thereof in Section 5.5.2, which revealed that the algorithm implicitly imposes an isotropic Gaussian prior over the weights whose precision parameter coincides with the inverse learning rate. Third, in Section 5.5.3, we proceeded to integrate out this precision parameter so as to obtain the true weight posterior (i.e. the posterior conditioned on the data but not on the learning rate), whose optimisation yields hyperparameter-free online gradient descent updates.

Finally, we also discussed how to adapt the generalised passive-aggressive learning framework from Section 4.2 to the dynamic bandit setting. While we were able to readily carry out this adaptation, it still required estimating the latent gradients as a preliminary step, which we covered in Section 5.6.1.

Taken together, the two approaches summarised above represent a significant contribution to the literature on online convex optimisation in dynamic environments.

7.3 Application to Online Portfolio Selection

This chapter culminates our research on online Bayesian techniques for portfolio management. It presents two novel families of algorithms for online portfolio selection, namely 'online maximum reversion' and 'adaptive online mean reversion', in Sections 6.3 and 6.5, respectively. The proposed approaches are able to overcome the limitations of existing state-of-the-art online portfolio selection strategies, discussed in Section 6.1, and were shown to deliver superior performance on the equity data sets widely used in literature and a real-world cross-asset data set, respectively.

Online maximum reversion is built upon the assumption that price-relative vectors change slowly over time. This was corroborated for the stock-market data set in Section 6.3.1, and permits the use of the loss function at time t (parameterised by the tth price-relative vector) as a surrogate for the unobserved loss at time $t+1$. With this in mind, we approximate the optimal portfolio $\mathbf{b}^*_{t+1}$ by the minimiser $\mathbf{b}_{t+1}$ of the current loss function, as detailed in Eq. (6.9). This leads to a hyperparameter-free single-period online reversion algorithm that does not suffer from data-snooping

bias, unlike any of its competitors. We also explored a multiperiod variant to alleviate the data-snooping issue affecting the most powerful online mean-reversion strategy known to date, namely 'online moving average reversion'.

In addition to the data-snooping bias, existing online portfolio selection techniques are fraught with selection bias as they are restricted to equity markets and, to fix this limitation, we devised adaptive online mean reversion strategies. To this end, we started from the passive-aggressive mean reversion and online moving average reversion algorithms, and expressed their respective portfolio updates in the form of online gradient descent updates with data-driven, but hyperparameter-dependent, learning-rate schedules (see Eq. (6.23) and (6.27), respectively). To reduce this hyperparameter dependency, we then replaced these learning-rate schedules with their equivalent maximum posterior counterparts derived from Eq. (5.73). The resulting strategies are more amenable to asset classes beyond equities, as well as to reasonable transaction fees.

8
Further Work

Contents

It is interesting to consider how one may further extend the studies presented throughout the thesis, and this chapter seeks to lay the foundations for many future avenues of research.

8.1 Extending Passive-Aggressive Learning

Whilst the generalised passive-aggressive (GPA) framework of Chapter 4 is flexible by construction of the model and algorithms, there are some simple ways in which it can be enhanced. First of all, it is possible to augment the model by substituting the Mahalanobis for the Euclidean distance in the objective of Eq. (4.4), so as to explicitly account for the distribution of the data. A further extension of the model is possible by noting that it is formulated in terms of a deterministic

point-estimation problem governed by a set of user-defined hyperparameters: the approach fails to capture model/prediction uncertainty and induces dependence on hyperparameter settings.

8.1.1 Bayesian Generalised Passive-Aggressive Learning

We propose a unified Bayesian approach that encompasses the suggested enhancements. Our starting point is the soft GPA criterion in Eq. (4.10), but without the loss expansion in the constraint and with the Mahalanobis instead of the Euclidean distance in the objective. By analogy with Eq. (4.27), this modified criterion can be rewritten as

$$\min_{\mathbf{w}\in\mathbb{R}^n} \Big\{ C\ell_t(\mathbf{w}) + \frac{1}{2}(\mathbf{w}-\mathbf{w}_t)^{\mathrm{T}}\boldsymbol{\Sigma}_t^{-1}(\mathbf{w}-\mathbf{w}_t) \Big\}, \tag{8.1}$$

where $\boldsymbol{\Sigma}_t$ denotes the covariance matrix of the weight vector at round t.

Solving (8.1) is equivalent to finding the mode of the pseudo-posterior distribution $p(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C)$ defined by

$$\begin{aligned} p(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C) &\propto \exp\Big\{ -C\ell_t(\mathbf{w}) - \frac{1}{2}(\mathbf{w}-\mathbf{w}_t)^{\mathrm{T}}\boldsymbol{\Sigma}_t^{-1}(\mathbf{w}-\mathbf{w}_t) \Big\} \\ &\propto L_t(\mathbf{w}|C)p(\mathbf{w}|\boldsymbol{\theta}_t), \end{aligned} \tag{8.2}$$

where $\mathcal{D}_t$ is the datum implicit in $\ell_t(\cdot)$, and $\boldsymbol{\theta}_t \equiv \{\mathbf{w}_t, \boldsymbol{\Sigma}_t\}$. The data-dependent factor is a pseudo-likelihood contribution given by

$$L_t(\mathbf{w}|C) = \exp\Big\{ -C\ell_t(\mathbf{w}) \Big\}, \tag{8.3}$$

where the prefix 'pseudo' refers to the fact that this quantity is un-normalised with respect to the observation $\mathcal{D}_t$, which also justifies our use of the name 'pseudo-posterior'[33]. The second factor, which we recognise as a Gaussian of the form

$$p(\mathbf{w}|\boldsymbol{\theta}_t) = \mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \boldsymbol{\Sigma}_t), \tag{8.4}$$

acts as an *approximate posterior* replacing the true, but generally intractable and offline, posterior $p(\mathbf{w}|\mathcal{D}_{1:t})$. The parameter vector $\boldsymbol{\theta}_t$ can be thought of as a

[33]In principle, one could work with an actual likelihood contribution if L_t was replaced by its normalised value $\widetilde{L}_t$, but we work with L_t instead because it leads to the traditional GPA weight estimates. This is common practice in the Bayesian treatment of other frequentist supervised-learning methods (see, e.g., [Polson and Scott, 2011; Deng et al., 2016]).

'summary statistic' for past observations, and needs to be updated at each step to incorporate new information.

Variational inference

In all but limited special cases, the function L_t is not a simple squared exponential, resulting in a new pseudo-posterior $p(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C)$ of a non-standard form. Inevitably, therefore, approximations are required. When the dimension n of the weight vector $\mathbf{w}$ is large, finding an accurate approximation is, in general, non-trivial. Our particular interest here is to form an approximation $p(\mathbf{w}|\boldsymbol{\theta}_{t+1})$ in which the parameters $\boldsymbol{\theta}_{t+1}$ are chosen so as to ensure that $p(\mathbf{w}|\boldsymbol{\theta}_{t+1})$ is as similar as possible to $p(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C)$.

We consider a Kullback-Leibler (KL) approach based on fitting a Gaussian to $p(\mathbf{w}|\boldsymbol{\theta}_{t+1})$, which is the most natural choice since $p(\mathbf{w}|\boldsymbol{\theta}_t)$ is also Gaussian and $p(\mathbf{w}|\boldsymbol{\theta}_{t+1})$ serves as the prior for $p(\mathbf{w}|\mathcal{D}_{t+1}, \boldsymbol{\theta}_{t+1}, C)$. Defining

$$\widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C) = \frac{L_t(\mathbf{w}|C)\mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \boldsymbol{\Sigma}_t)}{Z}, \qquad Z = \int L_t(\mathbf{w}|C)\mathcal{N}(\mathbf{w}|\mathbf{w}_t,\, \boldsymbol{\Sigma}_t)\,\mathrm{d}\mathbf{w}, \tag{8.5}$$

and fitting a Gaussian $p(\mathbf{w}|\boldsymbol{\theta}_{t+1}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{t+1},\, \boldsymbol{\Sigma}_{t+1})$ based on minimising the KL divergence $\mathrm{KL}[p(\mathbf{w}|\boldsymbol{\theta}_{t+1}) \,||\, \widetilde{p}(\mathbf{w}|\mathcal{D}_t, \boldsymbol{\theta}_t, C)]$, we obtain the bound $\log Z \geq \mathcal{B}_t(\boldsymbol{\theta}_{t+1})$ with

$$\begin{aligned}\mathcal{B}_t(\boldsymbol{\theta}_{t+1}) \equiv &-\langle \log p(\mathbf{w}|\boldsymbol{\theta}_{t+1})\rangle - \frac{1}{2}\log|2\pi\boldsymbol{\Sigma}_t| \\ &- \frac{1}{2}\langle(\mathbf{w}-\mathbf{w}_t)^{\mathrm{T}}\boldsymbol{\Sigma}_t^{-1}(\mathbf{w}-\mathbf{w}_t)\rangle + \langle \log L_t(\mathbf{w}|C)\rangle,\end{aligned} \tag{8.6}$$

where $\langle\cdot\rangle$ denotes the expectation with respect to $p(\mathbf{w}|\boldsymbol{\theta}_{t+1})$. One then numerically finds the best parameters $\boldsymbol{\theta}_{t+1}$ that maximise the bound.

Since the entropy of a Gaussian is trivial, the only potentially problematic term in evaluating $\mathcal{B}_t$ is $\langle \log L_t(\mathbf{w}|C)\rangle$. An important class of functions for which $\langle \log L_t(\mathbf{w}|C)\rangle_{\mathcal{N}(\mathbf{w}|\mathbf{w}_{t+1},\, \boldsymbol{\Sigma}_{t+1})}$ is computationally tractable is when $L_t(\mathbf{w}|C) = L(\mathbf{w}^{\mathrm{T}}\mathbf{x}_t|C)$ for some fixed vector $\mathbf{x}_t$[34]. In this case, the projection $\mathbf{w}^{\mathrm{T}}\mathbf{x}_t$ is also Gaussian distributed and we have

$$\langle \log L(\mathbf{w}^{\mathrm{T}}\mathbf{x}_t|C)\rangle_{\mathcal{N}(\mathbf{w}|\mathbf{w}_{t+1},\, \boldsymbol{\Sigma}_{t+1})} = \langle \log L(y_{t+1}|C)\rangle_{\mathcal{N}(y_{t+1}|\mathbf{w}_{t+1}^{\mathrm{T}}\mathbf{x}_t,\, \mathbf{x}_t^{\mathrm{T}}\boldsymbol{\Sigma}_{t+1}\mathbf{x}_t)}, \tag{8.7}$$

[34]This is the case for linear hypotheses, in which $\mathbf{x}_t$ represents the input vector at time t, so that the loss function takes the form $\ell_t(\mathbf{w}) = \ell(\mathbf{w}^{\mathrm{T}}\mathbf{x}_t)$.

which can be readily computed using any one-dimensional integration routine. Explicitly, as a function of $\boldsymbol{\theta}_{t+1}$, we have

$$2\mathcal{B}_t(\boldsymbol{\theta}_{t+1}) = -\log|\boldsymbol{\Sigma}_{t+1}| + n + \log|\boldsymbol{\Sigma}_t| - \mathrm{Tr}[\boldsymbol{\Sigma}_t^{-1}(\boldsymbol{\Sigma}_{t+1} + (\mathbf{w}_{t+1} - \mathbf{w}_t)(\mathbf{w}_{t+1} - \mathbf{w}_t)^{\mathrm{T}})] + 2\langle \log L(y_{t+1}|C)\rangle_{\mathcal{N}(y_{t+1}|\mathbf{w}_{t+1}^{\mathrm{T}}\mathbf{x}_t,\, \mathbf{x}_t^{\mathrm{T}}\boldsymbol{\Sigma}_{t+1}\mathbf{x}_t)}. \tag{8.8}$$

Whilst, in general, the variational bounds are non-concave in their variational parameters, provided L is log-concave, then $\mathcal{B}_t(\mathbf{w}_{t+1},\, \boldsymbol{\Sigma}_{t+1})$ is jointly concave in $\mathbf{w}_{t+1}$ and $\boldsymbol{\Sigma}_{t+1}$. By using structured covariance matrices $\boldsymbol{\Sigma}_{t+1}$, the method is scalable to very high dimensional problems [Challis and Barber, 2011].

8.1.2 Online Bayesian Passive-Aggressive Classification

The case of online passive-aggressive classification, which is characterised by the hinge -loss function $\ell_t(\mathbf{w}) = \max\{0, 1 - y_t(\mathbf{w}\cdot\mathbf{x}_t)\}$, is to be treated differently. This is because, as shown in [Polson and Scott, 2011, Theorem 1], the pseudo-likelihood corresponding to the hinge loss can be represented as a location-scale mixture of Gaussians, by introducing latent variables λ_t such that

$$L_t(\mathbf{w}|\gamma) = \exp\Big\{-2\gamma\max\Big\{0, 1 - y_t(\mathbf{w}\cdot\mathbf{x}_t)\Big\}\Big\} = \int_0^\infty L_t(\mathbf{w}, \lambda_t|\gamma)\,\mathrm{d}\lambda_t, \tag{8.9}$$

where

$$L_t(\mathbf{w}, \lambda_t|\gamma) = \frac{\sqrt{\gamma}}{\sqrt{2\pi\lambda_t}}\exp\Big\{-\frac{[1 + \lambda_t - y_t(\mathbf{w}\cdot\mathbf{x}_t)]^2}{2\gamma^{-1}\lambda_t}\Big\}. \tag{8.10}$$

This result allows us to pair observation y_t with a latent variable λ_t in such a way that L_t is the *marginal* counterpart of a joint pseudo-likelihood function $L_t(\mathbf{w}, \lambda_t|\gamma)$ in which $\mathbf{w}$ appears as part of a quadratic form. This implies that $L_t(\mathbf{w}, \lambda_t|\gamma)$ is conjugate to a multivariate normal prior distribution, which in turn allows the optimality criterion of online PA classifcation to be expressed as a conditionally Gaussian linear model, for which approximate Bayesian inference is trivial. We refer the interested reader to [Polson and Scott, 2011] and Section 4.3 for further details on different inference schemes.

8.1.3 An Alternative Approach to Bayesian GPA Learning

The theory of normal variance-mean mixtures, alluded to in the previous paragraph, is highly flexible as it enables the derivation of a data-augmentation scheme for a class of common regularization problems. Polson and Scott [2013] demonstrate this method on several examples, including sparse quantile regression and binary logistic regression. Their work can be borrowed and adapted to develop a data-augmentation approach to Bayesian generalised passive-aggressive learning.

8.2 Adaptive Gradient Methods for Dynamic Online Optimisation

As we already mentioned in Section 5.2, the authors of [Kessler et al., 2020] give a probabilistic interpretation of adaptive optimisation algorithms (in particular, ADAM [Kingma and Ba, 2015]) so as to make probabilistic inferences regarding the weights of neural networks, and in particular, to obtain cheap uncertainty estimates thereof. Although their focus is not on developing a self-tuning scheme for the learning rate parameter, their main theoretical result, exposed in Eq. (14), can be additionally harnessed for this purpose, and it would be both of theoretical and practical importance to explore this avenue, borrowing results from Chapter 5.

8.3 Application to Online Portfolio Selection

Our online portfolio selection experiments could be enriched with some simple extensions. Firstly, we have thus far only considered trading stocks/ETFs on a daily basis. We may wish to tackle other asset classes and/or rebalancing frequencies (potentially intra-day) that also exhibit mean-reverting patterns. For example, there is overwhelming statistical evidence on the mean reversion of interest rates, which can be monetised by trading fixed-income instruments.

Long-short portfolios would form another interesting extension of our online portfolio selection framework, which is confined to long-only portfolios. Unlike the asset management industry, this type of portfolio is popular among hedge

funds whose bread and butter is *statistical arbitrage* (e.g., long-short equity), and so this avenue would be of extreme practical importance to such financial market participants.

Appendices

A

Mathematical Background

Contents

A.1 Probability and Information Theory

A.1.1 Joint, marginal and conditional probability

Let the n (discrete or continuous) random variables $y_1, \ldots, y_n$ have a joint joint probability probability $p(y_1, \ldots, y_n)$, or $p(\mathbf{y})$ for short[35]. Technically, one ought to distinguish between probabilities (for discrete variables) and probability densities for continuous variables. Throughout the thesis we commonly use the term 'probability' to refer to both. Let us partition the variables in $\mathbf{y}$ into two groups, $\mathbf{y}_A$ and $\mathbf{y}_B$,

[35] One can deal with more general cases where the density function does not exist by using the distribution function.

where A and B are two disjoint sets whose union is the set $\{1, \ldots, n\}$, so that $p(\mathbf{y}) = p(\mathbf{y}_A, \mathbf{y}_B)$. Each group may contain one or more variables.

The *marginal* probability of $\mathbf{y}_A$ is given by

$$p(\mathbf{y}_A) = \int p(\mathbf{y}_A, \mathbf{y}_B)\, \mathrm{d}\mathbf{y}_B. \tag{A.1}$$

The integral is replaced by a sum if the variables are discrete valued. Notice that if the set A contains more than one variable, then the marginal probability is itself a joint probability—whether it is referred to as one or the other depends on the context. If the joint distribution is equal to the product of the marginals, then the variables are said to be *independent*, otherwise they are *dependent*.

The *conditional* probability function is defined as

$$p(\mathbf{y}_A|\mathbf{y}_B) = \frac{p(\mathbf{y}_A, \mathbf{y}_B)}{p(\mathbf{y}_B)}, \tag{A.2}$$

defined for $p(\mathbf{y}_B) > 0$, as it is not meaningful to condition on an impossible event. If $\mathbf{y}_A$ and $\mathbf{y}_B$ are independent, then the marginal $p(\mathbf{y}_A)$ and the conditional $p(\mathbf{y}_A|\mathbf{y}_B)$ are equal.

Using the definitions of both $p(\mathbf{y}_A|\mathbf{y}_B)$ and $p(\mathbf{y}_B|\mathbf{y}_A)$ we obtain *Bayes' theorem*:

$$p(\mathbf{y}_A|\mathbf{y}_B) = \frac{p(\mathbf{y}_B|\mathbf{y}_A)p(\mathbf{y}_A)}{p(\mathbf{y}_B)}. \tag{A.3}$$

Since conditional distributions are themselves probabilities, one can use all of the above also when further conditioning on other variables. For example, in supervised learning, one often conditions on the inputs throughout, which would lead e.g. to a version of Bayes' rule with additional conditioning on X in all four probabilities in Eq. (A.3).

A.1.2 Change of variables

For a univariate continuous random variable x with distribution $p(x)$, the transformation $y = f(x)$, where $f(x)$ is a monotonic function, has distribution

$$p(y) = p(x)\left|\frac{df}{dx}\right|^{-1}, \qquad x = f^{-1}(y). \tag{A.4}$$

For multivariate $\mathbf{x}$ and bijection $\mathbf{f}(\mathbf{x})$, then $\mathbf{y} = \mathbf{f}(\mathbf{x})$ has distribution

$$p(\mathbf{y}) = p(\mathbf{x} = \mathbf{f}^{-1}(\mathbf{y})) \left| \det \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right) \right|^{-1}, \tag{A.5}$$

where the Jacobian matrix has elements

$$\left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right]_{i,\,j} = \frac{\partial f_i(\mathbf{x})}{\partial x_j}. \tag{A.6}$$

Sometimes, one needs to consider transformations between different dimensions. For example, if $\mathbf{z}$ has lower dimension than $\mathbf{x}$, then one may introduce additional variables $\mathbf{z}'$ to define a new multivariate $\mathbf{y} = (\mathbf{z},\, \mathbf{z}')$ with the same dimension as $\mathbf{x}$. Then one applies the transformation (A.5) to give the distribution on the joint variables $\mathbf{y}$, from which $p(\mathbf{z})$ can be obtained by marginalisation.

A.1.3 Entropy and Kullback-Leibler divergence

The *entropy* $\mathrm{H}[p(\mathbf{x})]$ of a distribution $p(\mathbf{x})$ is a non-negative measure of the amount of 'uncertainty' in the distribution, and is defined as

$$\mathrm{H}[p(\mathbf{x})] \equiv -\int p(\mathbf{x}) \log p(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{A.7}$$

The integral is substituted by a sum for discrete variables. Entropy is measured in *bits* if the log is to the base 2 and in *nats* in the case of the natural log.

The Kullback-Leibler (KL) divergence (or relative entropy) $\mathrm{KL}(p\|q)$ between two distributions $p(\mathbf{x})$ and $q(\mathbf{x})$ is defined as

$$\mathrm{KL}(p\|q) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}\, \mathrm{d}\mathbf{x}. \tag{A.8}$$

It is easy to show that $\mathrm{KL}(p\|q) \geq 0$, with equality if $p = q$ (almost everywhere). The KL divergence can be viewed as the extra number of nats needed on average to code data generated from a source $p(\mathbf{x})$ under the distribution $q(\mathbf{x})$ as opposed to $p(\mathbf{x})$.

A.2 Convex Optimisation

In this section, we give a gentle introduction to convex optimisation and present some basic algorithms for solving convex mathematical programs. A broad and significantly more detailed literature exists, and the reader is referred to Nemirovski and Yudin [1983]; Rockafellar [1997]; Fletcher [2000]; Boyd and Vandenberghe [2004]; Borwein and Lewis [2006]; Bubeck [2015], among others. We give here only the most elementary analysis, and focus on the techniques that are of use to us throughout the thesis.

A.2.1 Basic definitions and setup

The goal in this chapter is to minimise a continuous and convex function over a convex subset of the Euclidean space. Henceforth, let $\mathcal{K} \subseteq \mathbb{R}^d$ be a bounded, convex and compact set in the Euclidean space.

Definition A.2.1. *The* diameter *of $\mathcal{K}$ is defined as*

$$\operatorname{diam}(\mathcal{K}) \equiv \sup_{\mathbf{x},\mathbf{y}\in\mathcal{K}} \|\mathbf{x}-\mathbf{y}\|_2, \tag{A.9}$$

where $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}\cdot\mathbf{x}}$ is the Euclidean norm.

Definition A.2.2. *A set $\mathcal{K}$ is* convex *if for any $\mathbf{x},\mathbf{y} \in \mathcal{K}$, all the points on the line segment connecting $\mathbf{x}$ and $\mathbf{y}$ also belong to $\mathcal{K}$, i.e.*

$$\forall \alpha \in [0,1], \quad \alpha\mathbf{x} + (1-\alpha)\mathbf{y} \in \mathcal{K}. \tag{A.10}$$

Definition A.2.3. *A function $f : \mathcal{K} \to \mathbb{R}$ is* convex *if for any $\mathbf{x},\mathbf{y} \in \mathcal{K}$,*

$$\forall \alpha \in [0,1], \quad f[(1-\alpha)\mathbf{x} + \alpha\mathbf{y}] \leq (1-\alpha)f(\mathbf{x}) + \alpha f(\mathbf{y}). \tag{A.11}$$

Equivalently, if $f(\cdot)$ is differentiable, that is, its gradient $\nabla f(\mathbf{x})$ exists for all $\mathbf{x} \in \mathcal{K}$, then it is convex if and only if $\forall \mathbf{x},\mathbf{y} \in \mathcal{K}$,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot (\mathbf{y}-\mathbf{x}). \tag{A.12}$$

Definition A.2.4. *For convex and non-differentiable functions $f(\cdot)$, the* subgradient *of $f(\cdot)$ at $\mathbf{x}$ is defined to be any vector $\mathbf{z}$ that satisfies the inequality*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \mathbf{z} \cdot (\mathbf{y} - \mathbf{x}), \tag{A.13}$$

for all $\mathbf{y} \in \mathcal{K}$. We call subdifferential set *the set of subgradients of $f(\cdot)$ at $\mathbf{x}$, and denote it as $\partial f(\mathbf{x})$. Furthermore, if $f(\cdot)$ is differentiable at $\mathbf{x}$, then $\partial f(\mathbf{x})$ contains a single element, namely the gradient $\nabla f(\mathbf{x})$ of $f(\cdot)$ at $\mathbf{x}$.*

We denote by $G > 0$ an upper bound on the norm of the subgradients of $f(\cdot)$ over $\mathcal{K}$, i.e. $\|\nabla f(\mathbf{x})\|_2 \leq G$ for all $\mathbf{x} \in \mathcal{K}$. Such an upper bound implies that the function $f(\cdot)$ is Lipschitz continuous with parameter G.

Definition A.2.5. *The function $f : \mathcal{K} \to \mathbb{R}$ is said to be* Lipschitz continuous *with constant $G > 0$ if for all $\mathbf{x}, \mathbf{y} \in \mathcal{K}$, we have*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq G\|\mathbf{x} - \mathbf{y}\|_2. \tag{A.14}$$

The optimisation and machine learning communities study special types of convex functions that admit useful properties, which in turn allow for more efficient optimisation. Below, we define the most notable of these properties.

Definition A.2.6. *We say that a function $f : \mathcal{K} \to \mathbb{R}$ is* λ-strongly convex *if*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) + \frac{\lambda}{2}\|\mathbf{y} - \mathbf{x}\|_2^2, \qquad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{K}. \tag{A.15}$$

Definition A.2.7. *A function $f : \mathcal{K} \to \mathbb{R}$ is* L-smooth *if*

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|_2^2, \qquad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{K}. \tag{A.16}$$

The latter condition is equivalent to the Lipschitz continuity of the gradients of $f(\cdot)$, with Lipschitz constant $L > 0$, i.e.

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2, \qquad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{K}. \tag{A.17}$$

Throughout the thesis, we refer to λ-strongly convex functions simply as strongly convex functions. Similarly, we shall call smooth functions those functions satisfying Definition A.2.7.

If the function is twice-differentiable and admits a second derivative, known as the Hessian for a function of several variables, Definitions A.2.6 and A.2.7 are equivalent to the following condition on the Hessian, which we denote by $\nabla^2 f(\mathbf{x})$:

$$\alpha \mathbf{I} \preceq \nabla^2 f(\mathbf{x}) \preceq \beta \mathbf{I}, \tag{A.18}$$

where $A \preceq B$ means the matrix $B - A$ is positive semi-definite.

When the function $f(\cdot)$ is both strongly convex and smooth, we say that it is γ-well-conditioned, where γ is the ratio between strong convexity and smoothness, and is also known as the *condition number* of $f(\cdot)$:

$$\gamma = \frac{\lambda}{L} \leq 1. \tag{A.19}$$

Example A.2.1. *The following functions are both strongly convex and smooth:*

1. *A quadratic form* $f(\mathbf{x}) = \mathbf{x}^\top A \mathbf{x} - 2\mathbf{b}^\top \mathbf{x}$*, where* $a\mathbf{I} \preceq A \preceq b\mathbf{I}$*, for* $a > 0$ *and* $b < \infty$*;*

2. *The regularised logistic loss* $f(\mathbf{x}) = \log[1 + \exp(\mathbf{b}^\top \mathbf{x})] + \frac{\lambda}{2}\|\mathbf{x}\|_2^2$*, where* $\lambda > 0$*.*

A.2.2 Projections onto convex sets

Throughout the thesis, we shall make use of a projection operation onto a convex set, which is defined as the closest point inside that set to a given point. Formally:

$$\Pi_{\mathcal{K}}(\mathbf{y}) \equiv \operatorname*{arg\,min}_{\mathbf{x} \in \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_2. \tag{A.20}$$

When clear from the context, we shall omit the $\mathcal{K}$ subscript. It is left as an exercise to the reader to prove that the projection of a given point over a compact convex set exists and is unique.

The computational complexity of projections is a subtle issue that depends much on the characterisation of $\mathcal{K}$ itself. Most generally, $\mathcal{K}$ can be represented by a

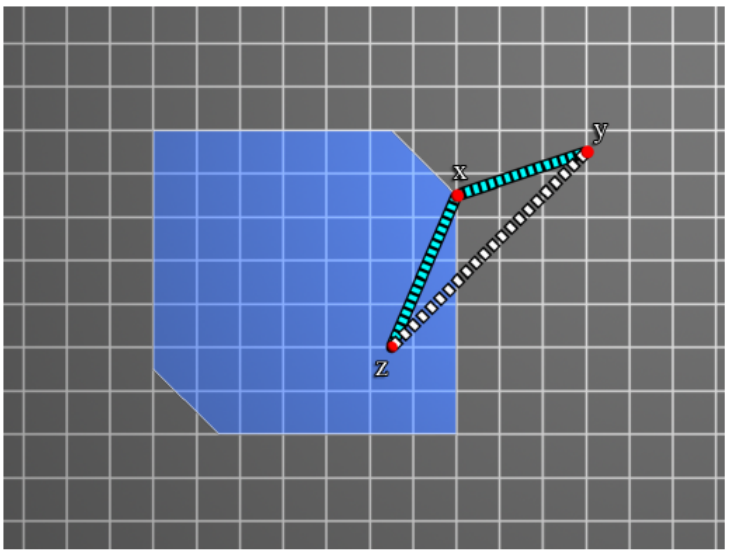


Figure A.1: An illustration of Pythagoras' theorem. Source: [Hazan, 2016].

membership oracle—an efficient procedure that is capable of deciding whether a given $\mathbf{x}$ belongs to $\mathcal{K}$ or not. In this case, projections can be computed in polynomial time. In certain special cases, projections can be computed very efficiently in near-linear time. A crucial property of projections that we shall make extensive use of is Pythagoras' theorem, which we state below for completeness.

Theorem A.2.1 (Pythagoras, circa 500 BC)**.** *Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a convex set, $\mathbf{y} \in \mathbb{R}^d$ and $\mathbf{x} = \Pi_{\mathcal{K}}(\mathbf{y})$. Then, for any $\mathbf{z} \in \mathcal{K}$, we have*

$$\|\mathbf{y} - \mathbf{z}\|_2 \geq \|\mathbf{x} - \mathbf{z}\|_2. \tag{A.21}$$

We note that there exists a more general version of Pythagoras' theorem. The above theorem and the definition of projections are true and valid not only for Euclidean norms, but for any norm. In addition, projections according to other distances that are not norms can be defined, in particular with respect to Bregman divergences, and an analogue of Pythagoras' theorem remains valid in such cases.

A.2.3 Introduction to optimality conditions

The standard curriculum of high-school mathematics contains the basic facts about when a function (usually in one dimension) attains a local optimum or saddle

point. The generalisation of these conditions to multiple dimensions is called the Karush-Kuhn-Tucker (KKT) conditions, and the reader is referred to Nemirovski and Yudin [1983]; Rockafellar [1997]; Fletcher [2000]; Boyd and Vandenberghe [2004]; Borwein and Lewis [2006]; Bubeck [2015] for an in-depth rigorous discussion of optimality conditions in general mathematical programming.

We shall only describe briefly and intuitively the main facts that are useful for our purposes. Naturally, we restrict ourselves to convex programming, and thus a local minimum of a convex univariate function is also a global minimum (because the second derivative of a convex function is non-negative everywhere).

The generalisation of the fact that a minimum of a convex differentiable function on $\mathbb{R}$ is a point in which its derivative is equal to zero is given by the multi-dimensional analogue that its gradient is equal to the zero vector:

$$\nabla f(\mathbf{x}) = \mathbf{0}_{d\times 1} \quad \iff \quad \mathbf{x} \in \operatorname*{arg\,min}_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}). \tag{A.22}$$

We will require a slightly more general, but equally intuitive, fact for constrained optimisation: at a minimum point of a constrained convex function, the inner product between the negative gradient and direction towards the interior of $\mathcal{K}$ is non-positive. This is depicted in Figure A.2, which shows that $-\nabla f(\mathbf{x}^*)$ defines a supporting hyperplane to $\mathcal{K}$. The intuition is that if the inner product were positive, one could improve the objective by moving in the direction of the projected negative gradient. This fact is stated formally in the following theorem.

Theorem A.2.2 (Karush-Kuhn-Tucker)**.** *Let $\mathcal{K} \subseteq \mathbb{R}^d$ be a convex set, $\mathbf{x}^* \in \arg\min_{\mathbf{x}\in\mathcal{K}} f(\mathbf{x})$. Then, for any $\mathbf{y} \in \mathcal{K}$, we have*

$$\nabla f(\mathbf{x}^*) \cdot (\mathbf{y} - \mathbf{x}^*) \geq 0. \tag{A.23}$$

A.3 Matrix Identities

The *matrix inversion lemma*, also known as the Woodbury, Sherman & Morrison formula (see e.g. [Press et al., 1992, p. 75]) states that

$$(\mathbf{Z} + \mathbf{U}\mathbf{W}\mathbf{V}^{\mathrm{T}})^{-1} = \mathbf{Z}^{-1} - \mathbf{Z}^{-1}\mathbf{U}(\mathbf{W}^{-1} + \mathbf{V}^{\mathrm{T}}\mathbf{Z}^{-1}\mathbf{U})^{-1}\mathbf{V}^{\mathrm{T}}\mathbf{Z}^{-1}, \tag{A.24}$$

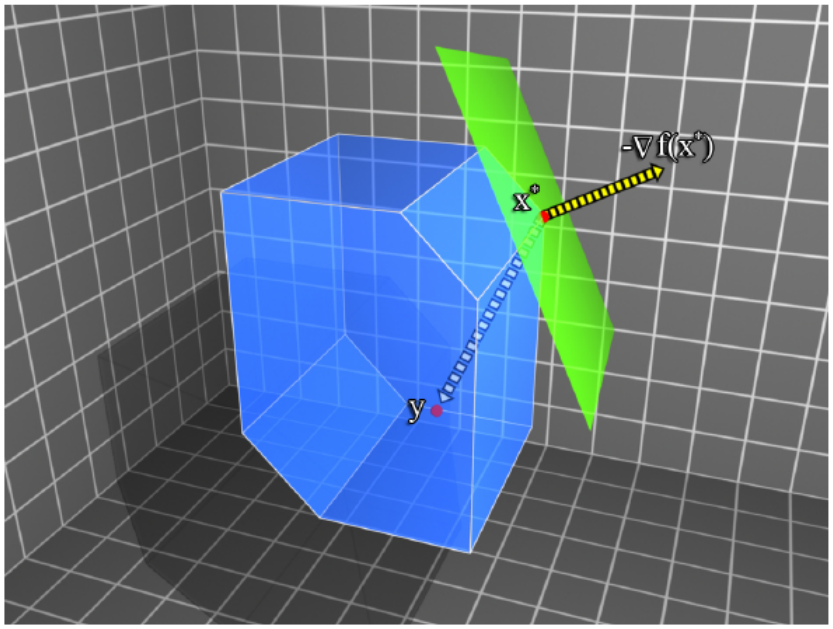


Figure A.2: Optimality conditions: negative gradient pointing outwards. Source: [Hazan, 2016].

assuming the relevant inverses all exist. Here $\mathbf{Z}$ is $n \times n$, $\mathbf{W}$ is $m \times m$ and $\mathbf{U}$ and $\mathbf{V}$ are both of size $n \times m$. Consequently, if $\mathbf{Z}^{-1}$ is known, and a low-rank (i.e. $m < n$) perturbation is made to $\mathbf{Z}$ as in the left-hand side of Eq. (A.24), considerable speedup can be achieved.

B

Probability Distributions

Contents

This appendix gives the definitions and some properties of the probability distributions used for inference in this thesis. For each distribution, we list some key statistics such as the expectation, the variance (or covariance), and the entropy $\mathrm{H}(x)$.

B.1 Uniform Distribution

The uniform distribution is often used as a non-informative prior, and assigns equal mass to all possible values in the distribution. When the variable is discrete, the probability is trivially set by dividing the total mass by the number of possible discrete values N,

$$p(n) = \frac{1}{N}, \tag{B.1}$$

but when the variable is continuous, it can be more difficult to specify a density. Fortunately, when the range of possible values is limited to some interval, the density is given as

$$\mathcal{U}(x|a,b) \equiv \frac{1}{b-a}\mathbb{1}_{(a,b)}(x) \tag{B.2}$$

which assigns equal mass within the interval and zero mass elsewhere. When a uniform distribution is required over all real values $x \in \mathbb{R}$, the distribution is *improper* since it is impossible to normalise the density.

The mean and variance of the continuous uniform distribution are respectively:

$$\mathbb{E}(x) = \frac{a+b}{2} \quad \text{and} \quad \text{Var}(x) = \frac{(b-a)^2}{12}. \tag{B.3}$$

An interesting property is that if x has distribution $\mathcal{U}(x|0,1)$, then $y \equiv a + (b-a)x$ will have distribution $\mathcal{U}(y|a,b)$.

B.2 Gamma Distribution

The Gamma is a probability distribution over a positive random variable $x > 0$ governed by a shape parameter a and a rate parameter b that are subject to the constraints $a > 0$ and $b > 0$ to ensure that the distribution can be normalised.

$$\mathcal{G}(x|a,b) = \frac{b^a}{\Gamma(a)} x^{a-1} \exp\Big\{-bx\Big\}\mathbb{1}_{(0,\infty)}(x) \tag{B.4}$$

$$\mathbb{E}(x) = \frac{a}{b} \tag{B.5}$$

$$\text{Var}(x) = \frac{a}{b^2} \tag{B.6}$$

$$\text{mode}(x) = \frac{a-1}{b} \qquad \text{for } a \geq 1, \tag{B.7}$$

where $\Gamma(a) = \int_0^\infty x^{a-1} \exp\Big\{-x\Big\}\,\mathrm{d}x$ is the gamma function.

The Gamma distribution is typically used as a prior for scale parameters (see e.g. [Berger, 1985]). An example of scale parameter would be the standard deviation σ of a univariate Gaussian distribution, after we have taken account of the location parameter μ. The uninformative prior is obtained as the special case $a = b = 0$, in which $p(x) = \mathcal{G}(x|a,b)$ takes the form $p(x) \propto 1/x$. Using the transformation rule (A.4) for densities, we see that this leads to a uniform prior over the logarithmic scale, i.e. $p(\log x) = \text{const}$.

B.3 Shifted Exponential Distribution

An important special case of the Gamma distribution is the exponential distribution, whose density is obtained by setting $a = 1$ in Eq. (B.4):

$$\mathcal{E}(x|\lambda) = \lambda \exp\Big\{ -\lambda x \Big\} \mathbb{1}_{(0,\infty)}(x), \tag{B.8}$$

with $\lambda = \beta$. The exponential distribution is used to describe the time between events in a Poisson point process, i.e. a process in which events occur continuously and independently at a constant average rate λ.

If we shift the origin of an exponentially distributed random variable, then we obtain the so-called shifted exponential distribution, whose density is given by

$$\mathcal{E}(x|\lambda,\theta) = \lambda \exp\Big\{ -\lambda(x-\theta) \Big\} \mathbb{1}_{(\theta,\infty)}(x), \tag{B.9}$$

where, as before, $\lambda > 0$ is the rate parameter, while $\theta \in \mathbb{R}$ is the location parameter. The shifted exponential distribution is widely used in the analysis of lifetime or response time data that arises in areas such as product performance evaluation and clinical trials [Bain, 1978; Lawless, 1982]. Using integration by parts, it is straightforward to show that its mean and variance are respectively:

$$\mathbb{E}(x) = \frac{1}{\lambda} + \theta, \qquad \mathrm{Var}(x) = \frac{1}{\lambda^2}. \tag{B.10}$$

Note that the variance coincides with that of the non-shifted exponential distribution, which is not surprising given that the variance is invariant to translations.

B.4 Gaussian Distribution

The Gaussian is the most widely used distribution for continuous variables. It is also known as the *normal* distribution. In the case of a single variable $x \in \mathbb{R}$, it is

governed by two parameters, the mean $\mu \in \mathbb{R}$ and the variance $\sigma^2 > 0$.

$$\mathcal{N}(x|\mu,\, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big\{ -\frac{1}{2\sigma^2}(x-\mu)^2 \Big\} \tag{B.11}$$

$$\mathbb{E}(x) = \mu \tag{B.12}$$

$$\mathrm{Var}(x) = \sigma^2 \tag{B.13}$$

$$\mathrm{mode}(x) = \mu \tag{B.14}$$

$$\mathrm{H}(x) = \frac{1}{2}\log\sigma^2 + \frac{1}{2}[1 + \log(2\pi)]. \tag{B.15}$$

The inverse of the variance $\beta = 1/\sigma^2$ is called the precision, and the square root of the variance, σ, is called the standard deviation.

For an n-dimensional column vector $\mathbf{x}$, the Gaussian is governed by an n-dimensional (column) mean vector $\boldsymbol{\mu}$ and an $n \times n$ covariance matrix $\boldsymbol{\Sigma}$ that must be symmetric and positive-definite.

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\, \boldsymbol{\Sigma}) = (2\pi)^{-n/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\Big\{ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) \Big\} \tag{B.16}$$

$$\mathbb{E}(\mathbf{x}) = \boldsymbol{\mu} \tag{B.17}$$

$$\mathrm{Cov}(\mathbf{x}) = \boldsymbol{\Sigma} \tag{B.18}$$

$$\mathrm{mode}(\mathbf{x}) = \boldsymbol{\mu} \tag{B.19}$$

$$\mathrm{H}(\mathbf{x}) = \frac{1}{2}\log|\boldsymbol{\Sigma}| + \frac{n}{2}[1 + \log(2\pi)]. \tag{B.20}$$

The inverse of the covariance matrix $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ is the precision matrix, which is also symmetric and positive definite. Averages of random variables tend to a Gaussian, by the central limit theorem, and the sum of two Gaussian variables is again Gaussian. The Gaussian is the distribution that maximises the entropy for a given variance (or covariance).

Any linear transformation of a Gaussian random variable is again a Gaussian. The marginal distribution of a multivariate Gaussian with respect to a subset of the variables is itself Gaussian, and similarly the conditional distribution is also Gaussian. See below for details.

B.4.1 Properties

The following properties [Rasmussen and Williams, 2006; Barber, 2012] are almost indispensable when dealing with Gaussian distributions.

Completing the square

A useful technique in manipulating Gaussians is the so-called technique of 'completing the square'. For example, the expression

$$\exp\Big\{-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{x}\Big\} \tag{B.21}$$

can be transformed as follows. First, we complete the square with respect to $\mathbf{x}$ in the exponent:

$$\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}-\mathbf{b}^{\mathrm{T}}\mathbf{x}=\frac{1}{2}(\mathbf{x}-\mathbf{A}^{-1}\mathbf{b})^{\mathrm{T}}\mathbf{A}(\mathbf{x}-\mathbf{A}^{-1}\mathbf{b})-\frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{b}. \tag{B.22}$$

Hence,

$$\exp\Big\{-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{x}\Big\}=\mathcal{N}(\mathbf{x}|\mathbf{A}^{-1}\mathbf{b},\,\mathbf{A}^{-1})\sqrt{|2\pi\mathbf{A}^{-1}|}\exp\Big\{\frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{b}\Big\}. \tag{B.23}$$

From this, one can derive

$$\int\exp\Big\{-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}+\mathbf{b}^{\mathrm{T}}\mathbf{x}\Big\}\,\mathrm{d}\mathbf{x}=\sqrt{|2\pi\mathbf{A}^{-1}|}\exp\Big\{\frac{1}{2}\mathbf{b}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{b}\Big\}. \tag{B.24}$$

Product of two Gaussians

The product of two Gaussians is another Gaussian, with a multiplicative factor:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1,\,\boldsymbol{\Sigma}_1)\,\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2,\,\boldsymbol{\Sigma}_2)=\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\,\boldsymbol{\Sigma})\frac{\exp\Big\{-\frac{1}{2}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^{\mathrm{T}}\mathbf{S}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)\Big\}}{\sqrt{|2\pi\mathbf{S}|}}, \tag{B.25}$$

where $\mathbf{S}\equiv\boldsymbol{\Sigma}_1+\boldsymbol{\Sigma}_2$ and the mean and covariance are given by

$$\boldsymbol{\mu}=\boldsymbol{\Sigma}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2)\qquad\text{and}\qquad\boldsymbol{\Sigma}=(\boldsymbol{\Sigma}_1^{-1}+\boldsymbol{\Sigma}_2^{-1})^{-1}. \tag{B.26}$$

Note that the resulting Gaussian has a precision (inverse covariance) equal to the sum of the precisions, and a mean equal to the convex sum of the means, weighted by the precisions.

To prove Eq. (B.25), simply write out the (lengthy) expressions by using the definition (B.16) of the Gaussian PDF, and expanding the terms inside the exponential to verify equality (hint: it may be helpful to expand $\mathbf{\Sigma}$ using the matrix inversion lemma in Eq. (A.24))

Affine transformations

Any affine transformation of a Gaussian random vector is itself Gaussian. More precisely, if $\mathbf{y} \equiv \mathbf{A}\mathbf{x} + \mathbf{b}$ and $\mathbf{x} \sim \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{\Sigma})$, where $\mathbf{x}$ is an n-dimensional column vector, $\mathbf{A}$ is a constant $m \times n$ matrix and $\mathbf{b}$ is an $m \times 1$ vector of constants, then $\mathbf{y}$ has multivariate Gaussian distribution with mean $\mathbf{A}\boldsymbol{\mu} + \mathbf{b}$ and covariance $\mathbf{A}\mathbf{\Sigma}\mathbf{A}^{\mathrm{T}}$. In other words,

$$\mathbf{x} \sim \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{\Sigma}) \quad \Longrightarrow \quad \mathbf{y} \equiv \mathbf{A}\mathbf{x} + \mathbf{b} \sim \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\mathbf{\Sigma}\mathbf{A}^{\mathrm{T}}). \tag{B.27}$$

Partitioned Gaussians

Perhaps the most important property of a multivariate Gaussian distribution is that its marginals and conditionals are both themselves Gaussian. That is, if we have a joint Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{\Sigma})$ with $\mathbf{\Lambda} \equiv \mathbf{\Sigma}^{-1}$ and we define the following partitions

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix}, \qquad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix} \tag{B.28}$$

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{aa} & \mathbf{\Sigma}_{ab} \\ \mathbf{\Sigma}_{ba} & \mathbf{\Sigma}_{bb} \end{bmatrix}, \qquad \mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_{aa} & \mathbf{\Lambda}_{ab} \\ \mathbf{\Lambda}_{ba} & \mathbf{\Lambda}_{bb} \end{bmatrix} \tag{B.29}$$

then the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ is given by

$$p(\mathbf{x}_a|\mathbf{x}_b) = \mathcal{N}(\mathbf{x}_a|\boldsymbol{\mu}_{a|b}, \mathbf{\Lambda}_{aa}^{-1}), \qquad \boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a - \mathbf{\Lambda}_{aa}^{-1}\mathbf{\Lambda}_{ab}(\mathbf{x}_b - \boldsymbol{\mu}_b) \tag{B.30}$$

and the marginal $p(\mathbf{x}_a)$ is given by

$$p(\mathbf{x}_a) = \mathcal{N}(\mathbf{x}_a|\boldsymbol{\mu}_a, \mathbf{\Sigma}_{aa}). \tag{B.31}$$

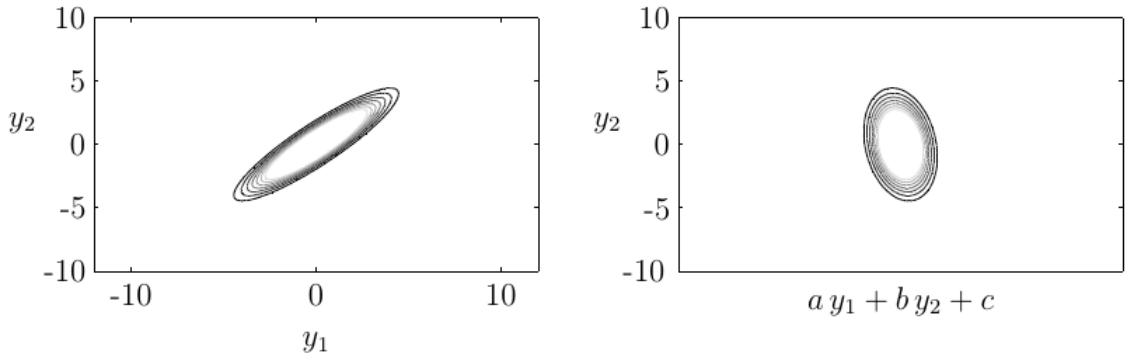


Figure B.1: Any variables over which we have a multivariate Gaussian are jointly Gaussian with any linear transformation of those variables. Source: [Osborne, 2010].

Marginal and conditional Gaussians

We now turn to another important property of the Gaussian distribution, illustrated in Figure B.1. Given a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution for $\mathbf{y}$ given $\mathbf{x}$ in the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\, \boldsymbol{\Lambda}^{-1}) \tag{B.32}$$

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b},\, \mathbf{L}^{-1}) \tag{B.33}$$

the marginal distribution of $\mathbf{y}$ and the conditional distribution for $\mathbf{x}$ given $\mathbf{y}$ are given by

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b},\, \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}}) \tag{B.34}$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Sigma}\{\mathbf{A}^{\mathrm{T}}\mathbf{L}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Lambda}\boldsymbol{\mu}\},\, \boldsymbol{\Sigma}) \tag{B.35}$$

where

$$\boldsymbol{\Sigma} = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}. \tag{B.36}$$

Gaussian average of a quadratic function

Last but not least, the following result is also useful:

$$\langle \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} \rangle_{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\, \boldsymbol{\Sigma})} = \boldsymbol{\mu}^{\mathrm{T}}\mathbf{A}\boldsymbol{\mu} + \mathrm{Tr}(\mathbf{A}\boldsymbol{\Sigma}). \tag{B.37}$$

B.5 Generalised Gaussian Distribution

The generalised inverse Gaussian distribution (GIG) is a three-parameter family of continuous probability distributions with probability density function

$$\mathcal{GIG}(x|\delta,\chi,\psi) = \frac{(\psi/\chi)^{\delta/2}}{2K_\delta(\sqrt{\psi\chi})} x^{\delta-1} \exp\Big\{ -\frac{1}{2}\left(\psi x + \frac{\chi}{x}\right)\Big\} \mathbb{1}_{(0,\infty)}(x), \tag{B.38}$$

where $K_\delta(\cdot)$ denotes the modified Bessel function of the second kind with index $\delta \in \mathbb{R}$, and $\chi, \psi > 0$. It is used extensively in geostatistics, statistical linguistics, finance, etc. Its statistical properties, and in particular the following lemma, can be found in [Jørgensen, 1982].

Lemma B.5.1. *Let x be distributed according to $\mathcal{GIG}(x|\delta,\chi,\psi)$, with $\psi > 0$ and $\chi > 0$. Then*

$$\mathbb{E}(x^p) = \left(\frac{\chi}{\psi}\right)^{p/2} \frac{K_{\delta+p}(\sqrt{\psi\chi})}{K_\delta(\sqrt{\psi\chi})}. \tag{B.39}$$

An immediate consequence of this lemma that turns out to be useful for our purposes is given below.

Corollary B.5.1. *If x is distributed according to $\mathcal{GIG}(x|1/2,\chi,\psi)$, where $\psi > 0$ and $\chi > 0$, then*

$$\mathbb{E}(x) = \frac{1+\sqrt{\psi\chi}}{\psi}, \qquad \mathbb{E}(x^{-1}) = \sqrt{\frac{\psi}{\chi}}. \tag{B.40}$$

It is straightforward to prove this corollary, by combining Lemma B.5.1 with the following one from [Abramowitz and Stegun, 1972].

Lemma B.5.2. *The modified Bessel function of the second kind, $K_\delta(\cdot)$, satisfies the following properties:*

1. $K_\delta(x) = K_{-\delta}(x)$;
2. $K_{\delta+1}(x) = 2\frac{\delta}{x}K_\delta(x) + K_{\delta-1}(x)$;
3. $K_{1/2}(x) = K_{-1/2}(x) = \sqrt{\frac{\pi}{2x}}\exp\{-x\}$.

B.5.1 Entropy

Using Lemmata B.5.1 and B.5.2, the entropy of the generalised inverse Gaussian distribution is given as

$$\begin{aligned}\mathrm{H}(x) &= \frac{\delta}{2}\log\frac{\chi}{\psi} + \log\Big[2K_\delta\Big(\sqrt{\psi\chi}\Big)\Big] - (\delta-1)\langle\log x\rangle \\ &\quad + \frac{\sqrt{\psi\chi}}{2K_\delta(\sqrt{\psi\chi})}\Big[K_{\delta+1}\Big(\sqrt{\psi\chi}\Big) + K_{\delta-1}\Big(\sqrt{\psi\chi}\Big)\Big] \\ &= \frac{\delta}{2}\log\frac{\chi}{\psi} + \log\Big[2K_\delta\Big(\sqrt{\psi\chi}\Big)\Big] - (\delta-1)\frac{\frac{\mathrm{d}}{\mathrm{d}\nu}K_\nu(\sqrt{\psi\chi})\Big|_{\nu=\delta}}{K_\delta(\sqrt{\psi\chi})} \\ &\quad + \frac{\sqrt{\psi\chi}}{2K_\delta(\sqrt{\psi\chi})}\Big[K_{\delta+1}\Big(\sqrt{\psi\chi}\Big) + K_{\delta-1}\Big(\sqrt{\psi\chi}\Big)\Big],\end{aligned} \tag{B.41}$$

where $\frac{\mathrm{d}}{\mathrm{d}\nu}K_\nu(\sqrt{\psi\chi})\Big|_{\nu=\delta}$ is the derivate of the modified Bessel function of the second kind with respect to its order ν, evaluated at $\nu=\delta$. In particular, when $\delta = 1/2$, this boils down to

$$\mathrm{H}(x) = \frac{1}{2}\left(1 - \log\frac{\psi}{2\pi}\right) - (\delta-1)\langle\log x\rangle, \tag{B.42}$$

which is easily proved by making use of Lemma B.5.2.

B.5.2 Special cases

It is well know that the special cases of the GIG distribution include the Gamma distribution $\mathcal{G}(x|\delta,\psi/2)$ when $\chi = 0$ and $\delta > 0$, the inverse Gamma distribution $\mathcal{IG}(x|-\delta,\chi/2)$ when $\psi = 0$ and $\delta < 0$, the inverse Gaussian distribution when $\delta = -1/2$, and the hyperbolic distribution when $\delta = 0$. We refer the reader to [Jørgensen, 1982] for details.

Note in particular that the PDF of the inverse Gaussian $\mathcal{GIG}(x|1/2,\chi,\psi)$ is

$$p(x) = \sqrt{\frac{\psi}{2\pi}}\exp\Big\{\sqrt{\psi\chi}\Big\}x^{-1/2}\exp\Big\{-\frac{1}{2}\left(\psi x + \frac{\chi}{x}\right)\Big\}\mathbb{1}_{(0,\infty)}(x). \tag{B.43}$$

Bibliography

M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, Inc., New York, 10th edition, 1972.

A. Agarwal, E. Hazan, S. Kale, and R. E. Schapire. Algorithms for Portfolio Management based on the Newton Method. In *Proceedings of the 23rd International Conference on Machine Learning (ICML 06)*, 2006.

A. Agarwal, O. Dekel, and L. Xiao. Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback. In *Proceedings of the 23rd Annual Conference on Learning Theory*, 2010.

K. Akcoglu, P. Drineas, and M. Kao. Fast universalization of investment strategies with provably good relative returns. In *Proceedings of International Colloquium on Automata, Languages and Programming*, pages 888–900, 2002.

K. Akcoglu, P. Drineas, and M. Kao. 2004. Fast Universalization of Investment Strategies. *SIAM J. Comput.*, 34:1–22, 2004.

M. Akian, A. Sulem, and M. I. Taksar. Dynamic Optimization of Long-Term Growth Rate for a Portfolio with Transaction Costs and Logarithmic Utility. *Mathematical Finance*, 11(2):153–188, 2001.

Ö. D. Akyildiz, V. Elvira, and J. Miguez. The Incremental Proximal Method: A Probabilistic Perspective. *CoRR*, abs/1807.04594, 2018. URL `https://arxiv.org/pdf/1807.04594.pdf`.

L. J. Bain. *Statistical Analysis of Reliability and Life-Testing Models*. Dekker, New York, 1978.

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

D. Barber and C. M. Bishop. *Ensemble Learning in Bayesian Neural Networks*, volume 168 of *NATO ASI Subseries F: Computer and System Sciences*, pages 215–237. Springer, 1998.

J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer-Verlag New York, 2nd edition edition, 1985.

O. Besbes, Y. Gur, and A. Zeevi. Non-Stationary Stochastic Optimization. *Operations Research*, 63(5):1227–1244, 2015.

C. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag New York, 2006.

M. Blondel, Y. Kubo, and U. Naonori. Online Passive-Aggressive Algorithms for Non-Negative Matrix Factorization and Completion. In S. Kaski and J. Corander, editors, *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 96–104, 2014.

A. Blum and A. Kalai. Universal Portfolios With and Without Transaction Costs. *Machine Learning*, 35(3):193–205, 1999.

A. Blum and Y. Mansour. From External to Internal Regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

W. F. M. Bondt and R. Thaler. Does the Stock Market Overreact? *The Journal of Finance*, 40(3):793–805, 1985.

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.

J. Borwein and A. Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, 2006.

G. E. P. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. Wiley-Interscience, 1992.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

L. Breiman. Optimal Gambling Systems for Favorable Games. In *Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 65–78. University of California Press, 1961.

G. L. Bretthorst. *Bayesian Spectrum Analysis and Parameter Estimation*, volume 48 of *Lecture Notes in Statistics*. Springer-Verlag New York, 1988.

S. Bubeck. Convex Optimization: Algorithms and Complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning and Games*. Cambridge University Press, 2006.

N. Cesa-Bianchi, P. M. Long, and M. K. Warmuth. Worst-Case Quadratic Loss Bounds for Prediction Using Linear Functions and Gradient Descent. *IEEE Transactions on Neural Networks*, 7(3):604–619, 1996.

E. Challis and D. Barber. Concave Gaussian Variational Approximations for Inference in Large-Scale Bayesian Linear Models. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15, pages 199–207. PMLR, 2011.

E. P. Chan. *Algorithmic Trading: Winning Strategies and their Rationale*. Wiley Trading Series. John Wiley and Sons, Inc., 2013.

R. Cont. Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues. *Quantitative Finance*, 1(2):223–236, 2001.

T. M. Cover. Universal Portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

T. M. Cover and D. H. Gluss. Empirical Bayes Stock Market Portfolios. *Advances in Applied Mathematics*, 7(2):170–181, 1986.

T. M. Cover and E. Ordentlich. Universal Portfolios with Side Information. *IEEE Transactions on Information Theory*, 42(2):348–363, 1996.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2nd edition edition, 2006.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

K. Crammer, A. Kulesza, and M. Dredze. Adaptive Regularization of Weight Vectors. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 414–422. Curran Associates, Inc., 2009.

P. Das and A. Banerjee. Meta Optimization and Its Application to Portfolio Selection. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 2011.

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 (AAAI 2013)*, pages 202–208, 2013.

M. H. A. Davis and A. R. Norman. Portfolio Selection with Transaction Costs. *Mathematics of Operations Research*, 15(4):676–713, 1990.

S. Deng, K. Gao, C. Du, W. Ma, G. Long, and Y. Li. Online Variational Bayesian Support Vector Regression. In *Proceedings of the 2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3950–3957, 2016.

P. Domingos and G. Hulten. A General Framework for Mining Massive Data Streams. *Journal of Computational and Graphical Statistics*, 12(4):945–949, 2003.

M. Dredze, K. Crammer, and F. Pereira. Confidence-Weighted Linear Classification. In *Proceedings of the 25th International Conference on Machine Learning*, pages 264–271. ACM, 2008.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient Projections onto the ℓ_1-Ball for Learning in High Dimensions. In A. McCallum and S. Roweis, editors, *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 272–279. Omnipress, 2008.

J. Duchi, E. Hazan, and Y. Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

A. Flaxman, A. Kalai, and B. McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.

R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition edition, 2000.

Y. Freund and R. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

A. A. Gaivoronski and F. Stella. Stochastic Nonstationary Optimization for Finding Universal Portfolios. *Annals of Operations Research*, 100:165–188, 2000.

X. Gao, X. Li, and S. Zhang. Online Learning with Non-Convex Losses and Non-Stationary Regret . In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*, 2018.

R. Garnett, M. A. Osborne, S. Reece, A. Rogers, and S. J. Roberts. equential Bayesian Prediction in the Presence of Changepoints and Faults. *The Computer Journal*, 53(9): 1430–1446, 2010.

L. Gyorfi and D. Schäfer. Nonparametric Prediction. In J. A. K. Suykens, G. Horvath, S. Basu, C. Micchelli, and J. Vandevalle, editors, *Advances in Learning Theory: Methods, Models and Applications*, pages 339–354. IOS Press, 2003.

L. Györfi and I. Vajda. Growth Optimal Investment with Transaction Costs. In Y. Freund, L. Györfi, G. Turán, and Zeugmann T., editors, *Proceedings of the 19th International Conference on Algorithmic Learning Theory*, volume 5254 of *Lecture Notes in Computer Science*, pages 108–122, Berlin, Heidelberg, 2008. Springer.

L. Györfi, G. Lugosi, and F. Udina. Nonparametric Kernel-Based Sequential Investment Strategies. *Mathematical Finance*, 16(2):337–357, 2006.

L. Györfi, A. Urbán, and I. Vajda. Kernel-Based Semi-Log-Optimal Empirical Portfolio Selection Strategies. *International Journal of Theoretical and Applied Finance*, 10(3): 505–516, 2007.

L. Györfi, F. Udina, and H. Walk. Nonparametric nearest neighbor based empirical portfolio selection strategies. *Statistics & Decisions*, 26:145–157, 2008.

L. Györfi, G. Ottucsák, and H. Walk. *Machine Learning for Financial Engineering*. Imperial College Press, 2012.

N. H. Hakansson and W. T. Ziemba. Capital growth theory. In *Handbooks in OR & MS*. Elsevier Science, 1995.

E. C. Hall and R. M. Willett. Dynamical Models and Tracking Regret in Online Convex Programming. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 579–587. PMLR, 2013. URL `http://proceedings.mlr.press/v28/hall13.html`.

E. C. Hall and R. M. Willett. Online Convex Optimization in Dynamic Environments. *IEEE Journal of Selected Topics in Signal Processing*, 9(4):647–662, 2015.

J. D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

J. D. Hamilton. *New Palgrave Dictionary of Economics*. Palgrave McMillan Ltd., 2008.

M. R. Hardy. A Regime-Switching Model of Long-Term Stock Returns. *North American Actuarial Journal Society of Acutaries*, 5(2):41–53, 2001.

S. Haykin. *Adaptive Filter Theory*. Pearson, 5 edition, 2013.

E. Hazan. *Efficient Algorithms for Online Convex Optimization and Their Applications*. PhD thesis, Princeton University, 2006.

E. Hazan. Introduction to Online Convex Optimization. *Foundations and Trends in Optimization*, 2(3–4):157–325, 2016.

E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th International Conference on Machine Learning (ICML 09)*, 2009.

E. Hazan, A. Agarwal, and S. Kale. Logarithmic Regret Algorithms for Online Convex Optimization. *Machine Learning*, 69(2-3):169–192, 2007.

D. P. Helmbold, R. E. Schapire, Y. Singer, and M. K. Warmuth. A Comparison of New and Old Algorithms for a Mixture Estimation Problem. *Machine Learning*, 27(1): 97–119, 1997.

D. P. Helmbold, R. E. Schapire, Y. Singer, and M. K. Warmuth. On-Line Portfolio Selection Using Multiplicative Updates. *Mathematical Finance*, 8(4):325–347, 1998.

R. Henao, X. Yuan, and L. Carin. Bayesian Nonlinear Support Vector Machines and Discriminative Factor Modeling. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1754–1762. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5507-bayesian-nonlinear-support-vector-machines-and-discriminative-factor-modeling.pdf`.

M. Herbster and M. K. Warmuth. Tracking the Best Expert. *Machine Learning*, 32(2): 151–178, 1998.

S. C. H. Hoi. Online Learning for Big Data Mining: Methods and Applications. In *SIAM International Conference on Data Mining (SDM13) Tutorials*, 2013.

S. C. H. Hoi, D. Sahoo, J. Lu, and P. Zhao. Online Learning: A Comprehensive Survey. *CoRR abs/1802.02871*, 2018.

D. Huang, J. Zhao, B. Li, S. C. H. Hoi, and S. Zhou. Robust Median Reversion Strategy for On-Line Portfolio Selection. In *Proceedings of the 23rd International Conference on Artificial Intelligence*, 2011.

J. C. Hull. *Options, Futures, and Other Derivatives*. Prentice Hall, 7th edition, 2008.

G. N. Iyengar and T. M. Cover. Growth Optimal Investment in Horse Race Markets with Costs. *IEEE Transactions on Information Theory*, 46(7):2675–2683, 2000.

A. Jadbabaie, A. Rakhlin, S.Shahrampour, and K. Sridharan. Online Optimization: Competing with Dynamic Comparators. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015.

A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

B. Jørgensen. *Statistical Properties of the Generalized Inverse Gaussian Distribution*. Lecture Notes in Statistics. Springer, New York, 1982.

A. Kalai and S. Vempala. Efficient Algorithms for Universal Portfolios. *Journal of Machine Learning Research*, 3:423–440, 2002.

A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82:35–45, 1960.

J. L. Kelly. A new interpretation of information rate. *The Bell System Technical Journal*, 35(4):917–926, 1956.

S. Kessler, A. Salas, V. W. C. Tan, S. Zohren, and S. J. Roberts. Practical Bayesian Neural Networks via Adaptive Optimization Methods. *ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning*, 2020. URL `https://arxiv.org/pdf/1811.03679.pdf`.

D. P. Kingma and J. L. Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

J. Kivinen and M. K. Warmuth. Exponentiated Gradient versus Gradient Descent for Linear Predictors. *Information and Computation*, 132:1–63, 1997.

J. F. Lawless. *Statistical Models and Methods for Lifetime Data*. Wiley, New York, 1982.

B. Li and S. C. H. Hoi. Online Portfolio Selection: A Survey. *ACM Computing Surveys*, 46(3), 2014.

B. Li and S. C. H. Hoi. *Online Portfolio Selection: Principles and Algorithms*. CRC Press, 2015.

B. Li, S. C. H. Hoi, and V. Gopalkrishnan. CORN : Correlation-driven Nonparametric Learning Approach for Portfolio Selection. *ACM Transactions on Intelligent Systems and Technology*, 1(1):1–30, 2010.

B. Li, S. C. H. Hoi, P. Zhao, and V. Gopalkrishnan. Confidence Weighted Mean Reversion Strategy for On-Line Portfolio Selection. In G. Gordon, D. Dunson, and M. Dudík, editors, *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 434–442. PMLR, 2011. URL `http://proceedings.mlr.press/v15/li11b.html`.

B. Li, P. Zhao, S. C. H. Hoi, and V. Gopalkrishnan. PAMR: Passive aggressive mean reversion strategy for portfolio selection . *Machine Learning*, 87:221–258, 2012.

B. Li, S. C. H. Hoi, P. Zhao, and V. Gopalkrishnan. Confidence Weighted Mean Reversion Strategy for Online Portfolio Selection. *ACM Transactions on Knowledge Discovery from Data*, 7(1):4:1–4:38, 2013.

B. Li, S. C. H. Hoi, D. Sahoo, and Z.-Y. Liu. Moving average reversion strategy for on-line portfolio selection. *Artificial Intelligence*, 222:104–123, 2015.

A. W. Lo and A. C. MacKinlay. When Are Contrarian Profits Due to Stock Market Overreaction? *Review of Financial Studies*, 3(2):175–205, 1990.

D. G. Luenberger. *Investment Science*. Oxford University Press, 1998.

D. J. C. MacKay. Bayesian Interpolation. *Neural Computation*, 4(3):415–447, 1992.

D. J. C. MacKay. Hyperparameters: Optimize, or integrate out? In *Maximum Entropy and Bayesian Methods*, pages 43–59. Kluwer Academic Publishers, 1996.

H. M. Markowitz. Portfolio Selection. *Journal of Finance*, 7(1):77–91, 1952.

H. M. Markowitz. *Portfolio Selection: Efficient Diversification of Investments*. John Wiley & Sons, 1959.

H. M. Markowitz, G. P. Todd, and W. F. Sharpe. *Mean-Variance Analysis in Portfolio Choice and Capital Markets*. Wiley, 2000.

T. Minka. Expectation Propagation for Approximate Bayesian Inference. In J. Breese and D. Koller, editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann, 2001a.

T. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001b.

A. Mokhtari, S. Shahrampour, A. Jadbabaie, and A. Ribeiro. Online Optimization in Dynamic Environments: Improved Regret Rates for Strongly Convex Problems. *ArXiv e-prints*, 2016.

G. Montana and F. Parrella. *Learning to Trade with Incremental Support Vector Regression Experts*, volume 5271 of *Lecture Notes in Computer Science*, pages 591–598. Springer Berlin Heidelberg, 2008.

G. Montana, K. Triantafyllopoulos, and T. Tsagaris. Flexible least squares for temporal data mining and statistical arbitrage. *Expert Systems with Applications*, 36(2):2819–2830, 2009.

A. S. Nemirovski and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley (Chichester and New York), 1983.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, New York, 2nd edition, 2006.

M. Opper. A Bayesian Approach to Online Learning. In D. Saad, editor, *Online Learning in Neural Networks*, pages 363–378. Cambridge University Press, 1998.

E. Ordentlich. *Universal Investment and Universal Data Compression*. PhD thesis, Stanford University, 1996.

M. Ormos and A. Urbán. Performance Analysis of Log-Optimal Portfolio Strategies with Transaction Costs. *Quantitative Finance*, 13(10):1587–1597, 2013.

Michael Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature*. PhD thesis, University of Oxford, 2010.

G. Ottucsák and I. Vajda. An Asymptotic Analysis of the Mean-Variance Portfolio Selection. *Statistics and Decisions*, 25:63–88, 2007.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1 (3):123–231, 2013.

G. Parisi. *Statistical Field Theory*. Addison-Wesley, 1988.

N. G. Polson and J. G. Scott. Data augmentation for non-Gaussian regression models using variance-mean mixtures. *Biometrika*, 100(2):459–471, 2013.

N. G. Polson and S. L. Scott. Data Augmentation for Support Vector Machines. *Bayesian Analysis*, 6(1):1–24, 2011.

J. M. Poterba and L. H. Summers. Mean Reversion in Stock Prices: Evidence and Implications. *Journal of Financial Economics*, 22(1):27–59, 1988.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts, 2006.

B. K. Ray and R. S. Tsay. Bayesian methods for change-point detection in long-range dependent processes. *Journal of Time Series Analysis*, 23(6):687–705, 2002.

R. Rockafellar. *Convex Analysis*. Princeton University Press, 1997.

S. Särkkä. *Bayesian Filtering and Smoothing*, volume 3 of *Institute of Mathematical Statistics Textbooks*. Cambridge University Press, 2013.

D. Schäfer. *Nonparametric Estimation for Financial Investment under Log-Utility*. PhD thesis, Universität Stuttgart, 2002.

S. Shahrampour and A. Jadbabaie. Distributed Online Optimization in Dynamic Environments Using Mirror Descent. *IEEE Transactions on Automatic Control*, 63(3), 2018.

S. Shalev-Shwartz. Online Learning and Online Convex Optimization. *Foundations and Trends in Machine Learning*, 4(2), 2011.

W. F. Sharpe. Mutual Fund Performance. *The Journal of Business*, 39(1):119–138, 1966.

T. Shi and J. Zhu. Online Bayesian Passive-Aggressive Learning. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 378–386, Bejing, China, 2014. PMLR.

T. Shi and J. Zhu. Online Bayesian Passive-Learning. *Journal of Machine Learning Research*, 18:1–39, 2017.

R. H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications: With R Examples*. Springer-Verlag New York, 3rd edition, 2011.

Y. Singer. Switching Portfolios. *International Journal of Neural Systems*, 8(4):488–495, 1997.

J. C. Spall. A One-measurement Form of Simultaneous Perturbation Stochastic Approximation. *Automatica*, 33(1):109–112, 1997.

G. Stoltz and G. Lugosi. Internal Regret in On-Line Portfolio Selection. *Machine Learning*, 59(1–2):125–159, 2005.

D. F. Swensen. *Unconventional Success: A Fundamental Approach to Personal Investment*. Free Press, 2005.

M. A. Tanner and W. H. Wong. The Calculation of Posterior Distributions by Data Augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

E. O. Thorp. Optimal Gambling Systems for Favorable Games. *Review of the International Statistical Institute*, 37(3):273–293, 1969.

E. O. Thorp. Portfolio Choice and the Kelly Criterion. In *Business and Economics Section of the American Statistical Association*, pages 215–224, 1971.

R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society (Series B)*, 58(1):267–288, 1996.

Tijman Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

K. Triantafyllopoulos and G. Montana. Dynamic modeling of mean-reverting spreads for statistical arbitrage. *Computational Management Science*, 8(1–2):23–49, 2011.

M. Turchi, A. Anastasopoulos, J. G. C. de Souza, and M. Negri. Adaptive Quality Estimation for Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 710–720, 2014.

V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

V. N. Vapnik. An Overview of Statistical Learning Theory. *IEEE Transactions on Neural Networks*, 10(5):988–999, 1999.

V. Vovk. Aggregating Strategies. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory (COLT 90)*, pages 371–386, 1990.

V. Vovk and C. Watkins. Universal portfolio selection. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT 98)*, pages 12–23, 1998.

J. Wang, P. Zhao, and S. C. H. Hoi. Exact Soft Confidence-Weighted Learning. In *Proceedings of the 29th International Conference on Machine Learning*, pages 107–114. Omnipress, 2012.

J. Wang, P. Zhao, and S. C. H. Hoi. Soft Confidence-Weighted Learning. *ACM Transactions on Intelligent Systems and Technology*, 8(1):15, 2016.

E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal*, 43:355–386, 1937.

B. Whitcher, S. D. Byers, P. Guttorp, and D. B. Percival. Testing for homogeneity of variance in time series: Long memory, wavelets, and the Nile River. *Water Resources Research*, 38(5):12–1–12–16, 2002.

T. Yang, L. Zhang, R. Jin, and J. Yi. Tracking Slowly Moving Clairvoyant: Optimal Dynamic Regret of Online Learning with True and Noisy Gradient . In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, Proceedings of Machine Learning Research, pages 449–457. PMLR, 2016.

M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. URL `http://arxiv.org/abs/1212.5701`.

L. Zhang, T. Yang, J. Yi, J. Rong, and Z.-H. Zhou. Improved Dynamic Regret for Non-degenerate Functions. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 732–741. Curran Associates, Inc., 2017.

M. Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003a.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936, 2003b.