PORTFOLIO MANAGEMENT AND OPTIMAL EXECUTION VIA CONVEX OPTIMIZATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF MANAGEMENT
SCIENCE AND ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Enzo Busseti

May 2018

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Stephen Boyd, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Yinyu Ye

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Lisa Borland,

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

For my mother
and my father.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This dissertation covers three related applications, in the field of finance, and in particular of multi-period investment management, of convex optimization [21] and model predictive control [7].

First, in Chapter 2, we look at the classical multi-period trading problem, consisting in trading (buying or selling) assets within a certain universe, for a sequence of periods in time. We formulate the problem, *i.e.*, its objective and constraints, as a convex optimization problem. Indeed, many practical constraints can be handled this way. We then develop a framework for single-period optimization: the trades in each period are found by solving a convex optimization problem that trades off expected return, risk, transaction cost and holding cost such as the borrowing cost for shorting assets. We further extend this model to a multi-period setting: optimization is used to plan a sequence of trades, with only the first one executed, using estimates of future quantities that are unknown when the trades are chosen. The single-period method traces back to Markowitz [83]; the multi-period methods trace back to model predictive control [7, 85]. Our contribution is to describe the single-period and multi-period methods in one simple framework, giving a clear description of the development and the approximations made. We do not address a critical component in a trading algorithm, the predictions or forecasts of future quantities. The methods we describe in this chapter can be thought of as good ways to exploit predictions, no matter how they are made. We have also developed a companion open-source software library

[25] that implements many of the ideas and methods described here. This chapter has been published in [17].

Then, in Chapter 3, we look at another classical model for the multi-period optimal investment problem, *Kelly gambling*, or *betting* [69]. We consider a sequence of identical bets: At each period, the investor allocates wealth among these bets, so as to maximize the expected growth rate of wealth. After the random returns are revealed the wealth is updated. Wealth in time follows a geometric random walk. It is well known (*e.g.*, [81]) that Kelly gambling can lead to substantial *drawdown*, *i.e.*, the loss of a large fraction of the initial wealth. We develop a technique to control the risk of such drawdown, using modern convex optimization. Unlike the Markowitz framework, the Kelly gambling model is well suited to handle *extreme* distributions of returns (*i.e.*, ones with fat tails), multi-modal distributions, or, in general, distributions that are not well approximated by multi-variate Gaussians. We complement the mathematical derivation with extensive numerical tests, on both finite and continuous distributions of returns, developed with open-source convex optimization software [41]. This chapter has been published in [26].

Finally, in Chapter 4, we look at an *optimal execution* problem, consisting in buying, or selling, a given quantity of some asset on a limit-order book market over some period of time (typically, a trading day). In particular, we study the case when the execution is *benchmarked* to the market volume weighted average price (VWAP), which is the most commonly used execution price benchmark [82]. The problem consists in minimizing the mean-variance of the execution price minus the benchmark. Again, we use multi-period optimization to choose the trades, and the resulting optimization problem is convex. This chapter is extracted from the unpublished manuscript [24].

In all three cases we use convex optimazion to solve the core problem, or a very close approximation of it. In practice these are solved to high accuracy in little time on commodity hardware (see, *e.g.*, § 2.7.5), thanks to strong theoretical guarantees (from modern convex optimization), and a rich and growing ecosystem of open source software.

Acknowledgements I would like to thank my advisor Stephen Boyd for his superb teaching and guidance, my co-authors on this dissertation Ernest Ryu, Steven Diamond, Ron Kahn, Kwangmoo Koh, Peter Nystrup, and Jan Speth, for their help, and Lisa Borland for many useful discussions. I would also like to thank the Department of Management Science and Engineering, Integral Development Co., and DARPA, for their financial support.

None of the work I have done for this dissertation would have been possible without the continued support of my family, friends, and many colleagues at Stanford and elsewhere. To them I'm grateful.

Chapter 2

Multi-Period Trading via Convex Optimization

In this chapter, we consider a basic model of multi-period trading, which can be used to evaluate the performance of a trading strategy. We describe a framework for single-period optimization, where the trades in each period are found by solving a convex optimization problem that trades off expected return, risk, transaction cost and holding cost such as the borrowing cost for shorting assets. We then describe a multi-period version of the trading method, where optimization is used to plan a sequence of trades, with only the first one executed, using estimates of future quantities that are unknown when the trades are chosen. The single-period method traces back to Markowitz; the multi-period methods trace back to model predictive control. Our contribution is to describe the single-period and multi-period methods in one simple framework, giving a clear description of the development and the approximations made. In this chapter we do not address a critical component in a trading algorithm, the predictions or forecasts of future quantities. The methods we describe in this chapter can be thought of as good ways to exploit predictions, no matter how they are made. We have also developed a companion open-source software library that implements many of the ideas and methods described in the chapter.

2.1 Introduction

Single and multi-period portfolio selection. Markowitz [83] was the first to formulate the choice of an investment portfolio as an optimization problem trading off risk and return. Traditionally, this was done independently of the cost associated with trading, which can be significant when trades are made over multiple periods [72]. Goldsmith [56] was among the first to consider the effect of transaction cost on portfolio selection in a single-period setting. It is possible to include many other costs and constraints in a single-period optimization formulation for portfolio selection [79, 94].

In multi-period portfolio selection, the portfolio selection problem is to choose a sequence of trades to carry out over a set of periods. There has been much research on this topic since the work of Samuelson [111] and Merton [88, 89]. Constantinides [34] extended Samuelson's discrete-time formulation to problems with proportional transaction costs. Davis and Norman [38] and Dumas and Lucian [44] derived similar results for the continuous-time formulation. Transaction costs, constraints, and time-varying forecasts are more naturally dealt with in a multi-period setting. Following Samuelson and Merton, the literature on multi-period portfolio selection is predominantly based on dynamic programming [6, 11], which properly takes into account the idea of recourse and updated information available as the sequence of trades are chosen (see [55] and references therein). Unfortunately, actually carrying out dynamic programming for trade selection is impractical, except for some very special or small cases, due to the 'curse of dimensionality' [107, 19]. As a consequence, most studies include only a very limited number of assets and simple objectives and constraints. A large literature studies multi-period portfolio selection in the absence of transaction cost (see, *e.g.*, [28] and references therein); in this special case, dynamic programming is tractable.

For practical implementation, various approximations of the dynamic programming approach are often used, such as approximate dynamic programming, or even simpler formulations that generalize the single-period formulations to multi-period optimization problems [19]. We will focus on these simple multi-period methods in this chapter. While these simplified approaches can be criticized for only approximating the full dynamic programming trading policy, the performance loss is likely very small in practical problems, for several reasons. In [19], the authors developed a numerical bounding method that quantifies the loss of optimality when using a simplified approach, and found it to be very small in numerical examples. But in fact, the dynamic programming formulation is itself an approximation, based on assumptions (like independent or identically distributed returns) that need not hold well in practice, so the idea of an 'optimal strategy' itself should be regarded with some suspicion.

Why now? What is different now, compared to 10, 20, or 30 years ago, is vastly more powerful computers, better algorithms, specification languages for optimization, and access to much more data. These developments have changed how we can use optimization in multi-period investing. In particular, we can now quickly run full-blown optimization, run multi-period optimization, and search over hyper-parameters in back-tests. We can run end-to-end analyses, indeed many at a time in parallel. Earlier generations of investment researchers, relying on computers much less powerful than today, relied more on separate models and analyses to estimate parameter values, and tested signals using simplified (usually unconstrained) optimization.

Goal. In this chapter we consider multi-period investment and trading. Our goal is to describe a simple model that takes into account the main practical issues that arise, and several simple and practical frameworks based on solving convex optimization problems [21] that determine the trades to make. We describe the approximations made, and briefly discuss how the methods can be used in practice. Our methods do not give a complete trading system, since we leave a critical part unspecified: Forecasting future returns, volumes, volatilities, and other important quantities (see, *e.g.*, [61]). This chapter describes good practical methods that can be used to trade, given forecasts.

The optimization-based trading methods we describe are practical and reliable when the problems to be solved are convex. Real-world single-period convex problems with thousands of assets can be solved using generic algorithms in well under a second, which is critical for evaluating a proposed algorithm with historical or simulated data, for many values of the parameters in the method.

Outline. We start in § 2.2 by describing a simple model of multi-period trading, taking into account returns, trading costs, holding costs, and (some) corporate actions. This model allows us to carry out simulation, used for what-if analyses, to see what would have happened under different conditions, or with a different trading strategy. The data in simulation can be realized past data (in a *back-test*) or simulated data that did not occur, but could have occurred (in a *what-if simulation*), or data chosen to be particularly challenging (in a *stress-test*). In § 2.3 we review several common metrics used to evaluate (realized or simulated) trading performance, such as active return and risk with respect to a benchmark.

We then turn to optimization-based trading strategies. In § 2.4 we describe *single-period optimization* (SPO), a simple but effective framework for trading based on optimizing the portfolio performance over a single period. In § 2.5 we consider *multi-period optimization* (MPO), where the trades are chosen by solving an optimization problem that covers multiple periods in the future.

Contribution. Most of the material that appears in this chapter has appeared before, in other papers, books, or EE364A, the Stanford course on convex optimization. Our contribution is to collect in one place the basic definitions, a careful description of the model, and discussion of how convex optimization can be used in multi-period trading, all in a common notation and framework. Our goal is not to survey all the work done in this and related areas, but rather to give a unified, self-contained treatment. Our focus is not on theoretical issues, but on practical ones that arise in multi-period trading. To further this goal, we have developed an accompanying open-source software library implemented in Python, and available at

`https://github.com/cvxgrp/cvxportfolio`.

Target audience. We assume that the reader has a background in the basic ideas of quantitative portfolio selection, trading, and finance, as described for example in the books by Grinold & Kahn [61], Meucci [91], or Narang [96]. We also assume that the reader has seen some basic mathematical optimization, specifically convex optimization [21]. The reader certainly does not need to know more than the very basic ideas of convex optimization, for example the overview material covered in § 1 of [21]. In a nutshell, our target reader is a quantitative trader, or someone who works with or for, or employs, one.

2.2 The Model

In this section we set the notation and give some detail of our simplified model of multi-period trading. We develop our basic dynamic model of trading, which tells us how a portfolio and associated cash account change over time, due to trading, investment gains, and various costs associated with trading and holding portfolios. The model developed in this chapter is independent of any method for choosing or evaluating the trades or portfolio strategy, and independent of any method used to evaluate the performance of the trading.

2.2.1 Portfolio asset and cash holdings

Portfolio. We consider a portfolio of holdings in n assets, plus a cash account, over a finite time horizon, which is divided into discrete time periods labeled $t = 1, \dots, T$. These time periods need not be uniformly spaced in real time or be of equal length; for example when they represent trading days, the periods are one (calendar) day during the week and three (calendar) days over a weekend. We use the label t to refer to both a point in time, the beginning of time period t, as well as the time interval from time t to $t + 1$. The time period in our model is arbitrary, and could be daily, weekly, or one hour intervals, for example. We will occasionally give examples where the time indexes trading days, but the same notation and model apply to any other time period.

Our investments will be in a *universe* of n assets, along with an associated cash account. We let $h_t \in \mathbf{R}^{n+1}$ denote the portfolio (or vector of positions or holdings) at the beginning of time period t, where $(h_t)_i$ is the *dollar value* of asset i at the beginning of time period t, with $(h_t)_i < 0$ meaning a short position in asset i, for $i = 1, \ldots, n$. The portfolio is *long-only* when the asset holdings are all nonnegative, *i.e.*, $(h_t)_i \geq 0$ for $i = 1, \ldots, n$.

The value of $(h_t)_{n+1}$ is the *cash balance*, with $(h_t)_{n+1} < 0$ meaning that money is owed (or borrowed). The dollar value for the assets is determined using the reference prices $p_t \in \mathbf{R}^n_+$, defined as the average of the bid and ask prices at the beginning of time period t. When $(h_t)_{n+1} = 0$, the portfolio is *fully invested*, meaning that we hold (or owe) zero cash, and all our holdings (long and short) are in assets.

Total value, exposure, and leverage. The total value (or *net asset value*, NAV) v_t of the portfolio, in dollars, at time t is $v_t = \mathbf{1}^T h_t$, where $\mathbf{1}$ is the vector with all entries one. (This is not quite the amount of cash the portfolio would yield on liquidation, due to transaction costs, discussed below.) Throughout this chapter we will assume that $v_t > 0$, *i.e.*, the total portfolio value is positive.

The vector

$$(h_t)_{1:n} = ((h_t)_1, \ldots, (h_t)_n)$$

gives the asset holdings. The *gross exposure* can be expressed as

$$\|(h_t)_{1:n}\|_1 = |(h_t)_1| + \cdots + |(h_t)_n|,$$

the sum of the absolute values of the asset positions. The *leverage* of the portfolio is the gross exposure divided by the value, $\|(h_t)_{1:n}\|_1 / v_t$. (Several other definitions of leverage are also used, such as half the quantity above.) The leverage of a fully invested long-only portfolio is one.

Weights. We will also describe the portfolio using weights or fractions of total value. The *weights* (or weight vector) $w_t \in \mathbf{R}^{n+1}$ associated with the portfolio h_t are defined as $w_t = h_t / v_t$. (Recall our assumption that $v_t > 0$.) By definition the weights sum to one, $\mathbf{1}^T w_t = 1$, and are unitless. The weight $(w_t)_{n+1}$ is the fraction of the total portfolio value held in cash. The weights are all nonnegative when the asset positions are long and the cash balance is nonnegative. The dollar value holdings vector can be expressed in terms of the weights as $h_t = v_t w_t$. The leverage of the portfolio can be expressed in terms of the weights as $\|w_{1:n}\|_1$, the ℓ_1-norm of the asset weights.

2.2.2 Trades

Trade vector. In our simplified model we assume that all trading, *i.e.*, buying and selling of assets, occurs at the beginning of each time period. (In reality the trades would likely be spread over at least some part of the period.) We let $u_t \in \mathbf{R}^n$ denote the dollar values of the trades, at the current price: $(u_t)_i > 0$ means we buy asset i and $(u_t)_i < 0$ means we sell asset i, at the beginning of time period t, for $i = 1, \ldots, n$. The number $(u_t)_{n+1}$ is the amount we put into the cash account (or take out, if it is negative). The vector $z_t = u_t / v_t$ gives the trades normalized by the total value. Like the weight vector w_t, it is unitless.

Post-trade portfolio. The post-trade portfolio is denoted

$$h_t^+ = h_t + u_t, \quad t = 1, \ldots, T.$$

This is the portfolio in time period t immediately after trading. The post-trade portfolio value is $v_t^+ = \mathbf{1}^T h_t^+$. The change in total portfolio value from the trades is given by

$$v_t^+ - v_t = \mathbf{1}^T h_t^+ - \mathbf{1}^T h_t = \mathbf{1}^T u_t.$$

The vector $(u_t)_{1:n} \in \mathbf{R}^n$ is the set of (non-cash) asset trades. Half its ℓ_1-norm $\|(u_t)_{1:n}\|_1/2$ is the *turnover* (in dollars) in period t. This is often expressed as a percentage of total value, as $\|(u_t)_{1:n}\|_1/(2v_t) = \|z_{1:n}\|_1/2$.

We can express the post-trade portfolio, normalized by the portfolio value, in terms of the weights $w_t = h_t / v_t$ and normalized trades as

$$h_t^+ / v_t = w_t + z_t. \tag{2.1}$$

Note that this normalized quantity does not necessarily add up to one.

2.2.3 Transaction cost

The trading incurs a trading or transaction cost (in dollars), which we denote as $\phi_t^{\text{trade}}(u_t)$, where $\phi_t^{\text{trade}} : \mathbf{R}^{n+1} \to \mathbf{R}$ is the (dollar) transaction cost function. We will assume that ϕ_t^{trade} does not depend on $(u_t)_{n+1}$, *i.e.*, there is no transaction cost associated with the cash account. To emphasize this we will sometimes write the transaction cost as $\phi_t^{\text{trade}}((u_t)_{1:n})$. We assume that $\phi_t^{\text{trade}}(0) = 0$, *i.e.*, there is no transaction cost when we do not trade. While $\phi_t^{\text{trade}}(u_t)$ is typically nonnegative, it can be negative in some cases, discussed below. We assume that the transaction cost function ϕ_t^{trade} is *separable*, which means it has the form

$$\phi_t^{\text{trade}}(x) = \sum_{i=1}^{n} (\phi_t^{\text{trade}})_i(x_i),$$

i.e., the transaction cost breaks into a sum of transaction costs associated with the individual assets. We refer to $(\phi_t^{\text{trade}})_i$, which is a function from $\mathbf{R}$ into $\mathbf{R}$, as the transaction cost function for asset i, period t. We note that some authors have used models of transaction cost which are not separable, for example Grinold's quadratic dynamic model [60].

A generic transaction cost model. A reasonable model for the scalar transaction cost functions $(\phi_t^{\text{trade}})_i$ is

$$x \mapsto a|x| + b\sigma\frac{|x|^{3/2}}{V^{1/2}} + cx, \tag{2.2}$$

where a, b, σ, V, and c are real numbers described below, and x is a dollar trade amount [61]. The number a is one half the bid-ask spread for the asset at the beginning of the time period, expressed as a fraction of the asset price (and so is unitless). We can also include in this term broker commissions or fees which are a linear function of the number of shares (or dollar value) bought or sold. The number b is a positive constant with unit inverse dollars. The number V is the total market volume traded for the asset in the time period, expressed in dollar value, so $|x|^{3/2}/V^{1/2}$ has units of dollars. The number σ the corresponding price volatility (standard deviation) over recent time periods, in dollars. According to a standard rule of thumb, trading one day's volume moves the price by about one day's volatility, which suggests that the value of the number b is around one. (In practice, however, the value of b is determined by fitting the model above to data on realized transaction costs.) The number c is used to create asymmetry in the transaction cost function. When $c = 0$, the transaction cost is the same for buying and selling; it is a function of $|x|$. When $c > 0$, it is cheaper to sell than to buy the asset, which generally occurs in a market where the buyers are providing more liquidity than the sellers (*e.g.*, if the book is not balanced in a limit order exchange). The asymmetry in transaction cost can also be used to model price movement during trade execution. Negative transaction cost can occur when $|c| > |a|$. The constants in the transaction cost model (2.2) vary with asset, and with trading period, *i.e.*, they are indexed by i and t. This 3/2 power transaction cost model is widely known and employed by practitioners.

We are not aware of empirical tests of the specific transaction cost model (2.2), but several references describe and validate similar models [78, 95, 9, 57]. In particular, these empirical works suggest that transaction cost grows (approximately) with the 3/2 power of transaction size.

Normalized transaction cost. The transaction cost model (2.2) is in dollars. We can normalize it by v_t, the total portfolio value, and express it in terms of z_i, the normalized trade of asset i, resulting in the function (with t suppressed for simplicity)

$$a_i|z_i| + b_i\sigma_i\frac{|z_i|^{3/2}}{(V_i/v)^{1/2}} + c_iz_i. \tag{2.3}$$

The only difference with (2.2) is that we use the normalized asset volume V_i/v instead of the dollar volume V_i. This shows that the same transaction cost formula can be used to express the dollar transaction cost as a function of the dollar trade, with the volume denoted in dollars, or the normalized transaction cost as a function of the normalized trade, with the volume normalized by the portfolio value.

With some abuse of notation, we will write the normalized transaction cost in period t as $\phi_t^{\mathrm{trade}}(z_t)$. When the argument to the transaction cost function is normalized, we use the version where asset volume is also normalized. The normalized transaction cost $\phi_t^{\mathrm{trade}}(z_t)$ depends on the portfolio value v_t, as well as the current values of the other parameters, but we suppress this dependence to lighten the notation.

Other transaction cost models. Other transaction cost models can be used. Common variants include a piecewise linear model, or adding a term that is quadratic in the trade value z_i [3, 60, 55]. Almost all of these are convex functions. (We discuss this later.) An example of a transaction cost term that is not convex is a fixed fee for any nonzero trading in an asset. For simulation, however, the transaction cost function can be arbitrary.

2.2.4 Holding cost

We will hold the post-trade portfolio h_t^+ over the tth period. This will incur a holding-based cost (in dollars) $\phi_t^{\mathrm{hold}}(h_t^+)$, where $\phi_t^{\mathrm{hold}} : \mathbf{R}^{n+1} \to \mathbf{R}$ is the holding cost function. Like transaction cost, it is typically nonnegative, but it can also be negative in certain cases, discussed below. The holding cost can include a factor related to the length of the period; for example if our periods are trading days, but holding costs are assessed on all days (including weekend and holidays), the Friday holding cost might be multiplied by three. For simplicity, we will assume that the holding cost function does not depend on the post-trade cash balance $(h_t^+)_{n+1}$.

A basic holding cost model includes a charge for borrowing assets when going short, which has the form

$$\phi_t^{\mathrm{hold}}(h_t^+) = s_t^T (h_t^+)_-, \tag{2.4}$$

where $(s_t)_i \geq 0$ is the borrowing fee, in period t, for shorting asset i, and $(z)_- = \max\{-z, 0\}$ denotes the negative part of a number z. This is the fee for shorting the post-trade assets, over the investment period, and here we are paying this fee in advance, at the beginning of the period. Our assumption that the holding cost does not depend on the cash account requires $(s_t)_{n+1} = 0$. But we can include a cash borrow cost if needed, in which case $(s_t)_{n+1} > 0$. This is the premium for borrowing, and not the interest rate, which is included in another part of our model, discussed below.

The holding cost (2.4), normalized by portfolio value, can be expressed in terms of weights and normalized trades as

$$\phi^{\text{hold}}(h_t^+)/v_t = s_t^T(w_t + z_t)_-. \tag{2.5}$$

As with the transaction cost, with some abuse of notation we use the same function symbol to denote the normalized holding cost, writing the quantity above as $\phi_t^{\text{hold}}(w_t + z_t)$. (For the particular form of holding cost described above, there is no abuse of notation since ϕ_t^{hold} is the same when expressed in dollars or normalized form.)

More complex holding cost functions arise, for example when the assets include ETFs (exchange traded funds). A long position incurs a fee proportional to h_i; when we hold a short position, we *earn* the same fee. This is readily modeled as a linear term in the holding cost. (We can in addition have a standard fee for shorting.) This leads to a holding cost of the form

$$\phi_t^{\text{hold}}(w_t + z_t) = s_t^T(w_t + z_t)_- + f_t^T(w_t + z_t),$$

where f_t is a vector with $(f_t)_i$ representing the per-period management fee for asset i, when asset i is an ETF.

Even more complex holding cost models can be used. One example is a piecewise linear model for the borrowing cost, which increases the marginal borrow charge rate when the short position exceeds some threshold. These more general holding cost functions are almost always convex. For simulation, however, the holding cost function can be arbitrary.

2.2.5 Self-financing condition

We assume that no external cash is put into or taken out of the portfolio, and that the trading and holding costs are paid from the cash account at the beginning of the period. This *self-financing condition* can be expressed as

$$\mathbf{1}^T u_t + \phi_t^{\text{trade}}(u_t) + \phi_t^{\text{hold}}(h_t^+) = 0. \tag{2.6}$$

Here $-\mathbf{1}^T u_t$ is the total cash out of the portfolio from the trades; (2.6) says that this cash out must balance the cash cost incurred, *i.e.*, the transaction cost plus the holding cost. The self-financing condition implies $v_t^+ = v_t - \phi_t^{\text{trade}}(u_t) - \phi_t^{\text{hold}}(h_t^+)$, *i.e.*, the post-trade value is the pre-trade value minus the transaction and holding costs.

The self-financing condition (2.6) connects the cash trade amount $(u_t)_{n+1}$ to the asset trades, $(u_t)_{1:n}$, by

$$(u_t)_{n+1} = -\left(\mathbf{1}^T (u_t)_{1:n} + \phi_t^{\text{trade}}((h_t + u_t)_{1:n}) + \phi_t^{\text{hold}}((u_t)_{1:n})\right). \tag{2.7}$$

Here we use the assumption that the transaction and holding costs do not depend on the $n+1$ (cash) component by explicitly writing the argument as the first n components, *i.e.*, those associated with the (non-cash) assets. The formula (2.7) shows that if we are given the trade values for the non-cash assets, *i.e.*, $(u_t)_{1:n}$, we can find the cash trade value $(u_t)_{n+1}$ that satisfies the self-financing condition (2.6).

We mention here a subtlety that will come up later. A trading algorithm chooses the asset trades $(u_t)_{1:n}$ before the transaction cost function ϕ_t^{trade} and (possibly) the holding cost function ϕ_t^{hold} are known. The trading algorithm must use *estimates* of these functions to make its choice of trades. The formula (2.7) gives the cash trade amount that is *realized.*

Normalized self-financing. By dividing the dollar self-financing condition (2.6) by the portfolio value v_t, we can express the self-financing condition in terms of weights and normalized trades as

$$\mathbf{1}^T z_t + \phi_t^{\text{trade}}(v_t z_t)/v_t + \phi_t^{\text{hold}}(v_t(w_t + z_t))/v_t = 0,$$

where we use $u_t = v_t z_t$ and $h_t^+ = v_t(w_t + z_t)$, and the cost functions above are the dollar value versions. Expressing the costs in terms of normalized values we get

$$\mathbf{1}^T z_t + \phi_t^{\text{trade}}(z_t) + \phi_t^{\text{hold}}(w_t + z_t) = 0, \tag{2.8}$$

where here the costs are the normalized versions.

As in the dollar version, and assuming that the costs do not depend on the cash values, we can express the cash trade value $(z_t)_{n+1}$ in terms of the non-cash asset trade values $(z_t)_{1:n}$ as

$$(z_t)_{n+1} = -\left(\mathbf{1}^T (z_t)_{1:n} + \phi_t^{\text{trade}}((w_t + z_t)_{1:n}) + \phi_t^{\text{hold}}((z_t)_{1:n})\right). \tag{2.9}$$

2.2.6 Investment

The post-trade portfolio and cash are invested for one period, until the beginning of the next time period. The portfolio at the next time period is given by

$$h_{t+1} = h_t^+ + r_t \circ h_t^+ = (\mathbf{1} + r_t) \circ h_t^+, \quad t = 1, \ldots, T-1,$$

where $r_t \in \mathbf{R}^{n+1}$ is the vector of asset and cash returns from period t to period $t+1$ and $\circ$ denotes Hadamard (elementwise) multiplication of vectors. The return of asset i over period t is defined as

$$(r_t)_i = \frac{(p_{t+1})_i - (p_t)_i}{(p_t)_i}, \quad i = 1, \ldots, n,$$

the fractional increase in the asset price over the investment period. We assume here that the prices and returns are adjusted to include the effects of stock splits and dividends. We will assume that the prices are nonnegative, so $\mathbf{1} + r_t \geq 0$ (where the inequality means elementwise). We mention an alternative to our definition above, the *log-return*,

$$\log \frac{(p_{t+1})_i}{(p_t)_i} = \log(1 + (r_t)_i), \quad i = 1, \ldots, n.$$

For returns that are small compared to one, the log-return is very close to the return defined above.

The number $(r_t)_{n+1}$ is the return to cash, *i.e.*, the risk-free interest rate. In the simple model, the cash interest rate is the same for cash deposits and loans. We can also include a premium for borrowing cash (say) in the holding cost function, by taking $(s_t)_{n+1} > 0$ in (2.4). When the asset trades $(u_t)_{1:n}$ are chosen, the asset returns $(r_t)_{1:n}$ are not known. It is reasonable to assume that the cash interest rate $(r_t)_{n+1}$ is known.

Next period portfolio value. For future reference we work out some useful formulas for the next period portfolio value. We have

$$\begin{aligned} v_{t+1} &= \mathbf{1}^T h_{t+1} \\ &= (\mathbf{1} + r_t)^T h_t^+ \\ &= v_t + r_t^T h_t + (\mathbf{1} + r_t)^T u_t \\ &= v_t + r_t^T h_t + r_t^T u_t - \phi_t^{\text{trade}}(u_t) - \phi_t^{\text{hold}}(h_t^+). \end{aligned}$$

Portfolio return. The *portfolio realized return* in period t is defined as

$$R_t^{\text{p}} = \frac{v_{t+1} - v_t}{v_t},$$

the fractional increase in portfolio value over the period. It can be expressed as

$$R_t^{\text{p}} = r_t^T w_t + r_t^T z_t - \phi_t^{\text{trade}}(z_t) - \phi_t^{\text{hold}}(w_t + z_t). \tag{2.10}$$

This is easily interpreted. The portfolio return over period t consists of four parts:

- $r_t^T w_t$ is the portfolio return without trades or holding cost,
- $r_t^T z_t$ is the return on the trades,
- $-\phi_t^{\text{trade}}(z_t)$ is the transaction cost, and
- $-\phi_t^{\text{hold}}(w_t + z_t)$ is the holding cost.

Next period weights. We can derive a formula for the next period weights w_{t+1} in terms of the current weights w_t and the normalized trades z_t, and the return r_t, using the equations above. Simple algebra gives

$$w_{t+1} = \frac{1}{1+R_t^{\mathrm{p}}}(\mathbf{1}+r_t) \circ (w_t + z_t). \tag{2.11}$$

By definition, we have $\mathbf{1}^T w_{t+1} = 1$. This complicated formula reduces to $w_{t+1} = w_t + z_t$ when $r_t = 0$. We note for future use that when the per-period returns are small compared to one, we have $w_{t+1} \approx w_t + z_t$.

2.2.7 Aspects not modeled

We list here some aspects of real trading that our model ignores, and discuss some approaches to handle them if needed.

External cash. Our self-financing condition (2.6) assumes that no external cash enters or leaves the portfolio. We can easily include external deposits and withdrawals of cash by replacing the right-hand side of (2.6) with the external cash put into the account (which is positive for cash deposited into the account and negative for cash withdrawn).

Dividends. Dividends are usually included in the asset return, which implicitly means they are re-invested. Alternatively we can include cash dividends from assets in the holding cost, by adding the term $-d_t^T h_t$, where d_t is the vector of dividend rates (in dollars per dollar of the asset held) in period t. In other words, we can treat cash dividends as negative holding costs.

Non-instant trading. Our model assumes all trades are carried out instantly at the beginning of each investment period, but the trades are really executed over some fraction of the period. This can be modeled using the linear term in the transaction cost, which can account for the movement of the price during the execution. We can also change the dynamics equation

$$h_{t+1} = (\mathbf{1} + r_t) \circ (h_t + u_t)$$

to

$$h_{t+1} = (\mathbf{1} + r_t) \circ h_t + (1 - \theta_t/2)(\mathbf{1} + r_t) \circ u_t,$$

where θ_t is the fraction of the period over which the trades occur. In this modification, we do not get the full period return on the trades when $\theta_t > 0$, since we are moving into the position as the price moves.

The simplest method to handle non-instant trading is to use a shorter period. For example if we are interested in daily trading, but the trades are carried out over the whole trading day and we wish to model this effect, we can move to an hourly model.

Imperfect execution. Here we distinguish between $u_t^{\rm req}$, the requested trade, and u_t, the actual realized trade [105]. In a back-test simulation we might assume that some (very small) fraction of the requested trades are only partially completed.

Multi-period price impact. This is the effect of a large order in one period affecting the asset price in future periods [3, 102]. In our model the transaction cost is only a function of the current period trade vector, not previous ones.

Trade settlement. In trade settlement we keep track of cash from trades one day and two days ago (in daily simulation), as well as the usual (unencumbered) cash account which includes all cash from trades that occurred three or more days ago, which have already settled. Shorting expenses come from the unencumbered cash, and trade-related cash moves immediately into the one day ago category (for daily trading).

Merger/acquisition. In a certain period one company buys another, converting the shares of the acquired company into shares of the acquiring company at some rate. This modifies the asset holdings update. In a cash buyout, positions in the acquired company are converted to cash.

Bankruptcy or dissolution. The holdings in an asset are reduced to zero, possibly with a cash payout.

Trading freeze. A similar action is a trading freeze, where in some time periods an asset cannot be bought, or sold, or both.

2.2.8 Simulation

Our model can be used to simulate the evolution of a portfolio over the periods $t = 1, \ldots, T$. This requires the following data, when the standard model described above is used. (If more general transaction or holding cost functions are used, any data required for them is also needed.)

- Starting portfolio and cash account values, $h_1 \in \mathbf{R}^{n+1}$.
- Asset trade vectors $(u_t)_{1:n}$. The cash trade value $(u_t)_{n+1}$ is determined from the self-financing condition by (2.7).
- Transaction cost model parameters $a_t \in \mathbf{R}^n$, $b_t \in \mathbf{R}^n$, $c_t \in \mathbf{R}^n$, $\sigma_t \in \mathbf{R}^n$, and $V_t \in \mathbf{R}^n$.
- Shorting rates $s_t \in \mathbf{R}^n$.
- Returns $r_t \in \mathbf{R}^{n+1}$.
- Cash dividend rates $d_t \in \mathbf{R}^n$, if they are not included in the returns.

Back-test. In a back-test the values would be past realized values, with $(u_t)_{1:n}$ the trades proposed by the trading algorithm being tested. Such a test estimates what the evolution of the portfolio would have been with different trades or a different trading algorithm. The simulation determines the portfolio and cash account values over the simulation period, from which other metrics, described in § 2.3 below, can be computed. As a simple example, we can compare the performance of re-balancing to a given target portfolio daily, weekly, or quarterly.

A simple but informative back-test is to simulate the portfolio evolution using the actual trades that were executed in a portfolio. We can then compare the actual and simulated or predicted portfolio holdings and total value over some time period. The true and simulated portfolio values will not be identical, since our model relies on estimates of transaction and holding costs, assumes instantaneous trade execution, and so on.

What-if simulations. In a what-if simulation, we change the data used to carry out the simulation, *i.e.*, returns, volumes, and so on. The values used are ones that (presumably) could have occurred. This can be used to stress-test a trading algorithm, by using data that did not occur, but would have been very challenging.

Adding uncertainty in simulations. Any simulation of portfolio evolution relies on models of transaction and holding costs, which in turn depend on parameters. These parameters are not known exactly, and in any case, the models are not exactly correct. So the question arises, to what extent should we trust our simulations? One simple way to check this is to carry out multiple simulations, where we randomly perturb the model parameters by reasonable amounts. For example, we might vary the daily volumes from their true (realized) values by 10% each day. If simulation with parameters that are perturbed by reasonable amounts yields divergent results, we know that (unfortunately) we cannot trust the simulations.

2.3 Metrics

Several generic performance metrics can be used to evaluate the portfolio performance.

2.3.1 Absolute metrics

We first consider metrics that measure the growth of portfolio value in absolute terms, not in comparison to a benchmark portfolio or the risk-free rate.

Return and growth rate. The average realized return over periods $t = 1, \ldots, T$ is

$$\overline{R^{\mathrm{p}}} = \frac{1}{T}\sum_{t=1}^{T} R_t^{\mathrm{p}}.$$

An alternative measure of return is the *growth rate* (or log-return) of the portfolio in period t, defined as

$$G_t^{\mathrm{p}} = \log(v_{t+1}/v_t) = \log(1 + R_t^{\mathrm{p}}).$$

The average growth rate of the portfolio is the average value of G_t^{p} over the periods $t = 1, \ldots, T$. For per-period returns that are small compared to one (which is almost always the case in practice) G_t^{p} is very close to R_t^{p}.

The return and growth rates given above are per-period. For interpretability they are typically *annualized* [4]: Return and growth rates are multiplied by P, where P is the number of periods in one year. (For periods that are trading days, we have $P \approx 250$.)

Volatility and risk. The realized *volatility* [15] is the standard deviation of the portfolio return time series,

$$\sigma^{\mathrm{p}} = \left(\frac{1}{T}\sum_{t=1}^{T}(R_t^{\mathrm{p}} - \overline{R^{\mathrm{p}}})^2\right)^{1/2}.$$

(This is the maximum-likelihood estimate; for an unbiased estimate we replace $1/T$ with $1/(T-1)$). The square of the volatility is the *quadratic risk*. When R_t^{p} are small (in comparison to 1), a good approximation of the quadratic risk is the second moment of the return,

$$(\sigma^{\mathrm{p}})^2 \approx \frac{1}{T}\sum_{t=1}^{T}(R_t^{\mathrm{p}})^2.$$

The volatility and quadratic risk given above are per-period. For interpretability they are typically annualized. To get the annualized values we multiply volatility by $\sqrt{P}$, and quadratic risk by P. (This scaling is based on the idea that the returns in different periods are independent random variables.)

2.3.2 Metrics relative to a benchmark

Benchmark weights. It is common to measure the portfolio performance against a *benchmark*, given as a set of weights $w_t^{\mathrm{b}} \in \mathbf{R}^{n+1}$, which are fractions of the assets (including cash), and satisfy $\mathbf{1}^T w_t^{\mathrm{b}} = 1$. We will assume the benchmark weights are nonnegative, *i.e.*, the entries in w_t^{b} are nonnegative. The benchmark weight $w_t^{\mathrm{b}} = e_{n+1}$ (the unit vector with value 0 for all entries except the last, which has value 1) represents the cash, or risk-free, benchmark. More commonly the benchmark consists of a particular set of assets with weights proportional to their capitalization. The benchmark return in period t is $R_t^{\mathrm{b}} = r_t^T w_t^{\mathrm{b}}$. (When the benchmark is cash, this is the risk-free interest rate $(r_t)_{n+1}$.)

Active and excess return. The *active* return [114, 61] (of the portfolio, with respect to a benchmark) is given by

$$R_t^{\mathrm{a}} = R_t^{\mathrm{p}} - R_t^{\mathrm{b}}.$$

In the special case when the benchmark consists of cash (so that the benchmark return is the risk-free rate) this is known as *excess return*, denoted

$$R_t^{\mathrm{e}} = R_t^{\mathrm{p}} - (r_t)_{n+1}.$$

We define the average *active return* $\overline{R^{\mathrm{a}}}$, relative to the benchmark, as the average of R_t^{a}. We have

$$\begin{aligned} R_t^{\mathrm{a}} &= R_t^{\mathrm{p}} - R_t^{\mathrm{b}} \\ &= r_t^T \left(w_t - w_t^{\mathrm{b}}\right) + r_t^T z_t - \phi_t^{\mathrm{trade}}(z_t) - \phi_t^{\mathrm{hold}}(w_t + z_t). \end{aligned}$$

Note that if $z_t = 0$ and $w_t = w_t^{\mathrm{b}}$, *i.e.*, we hold the benchmark weights and do not trade, the active return is zero. (This relies on the assumption that the benchmark weights are nonnegative, so $\phi_t^{\mathrm{hold}}(w_t^{\mathrm{b}}) = 0$.)

Active risk. The standard deviation of R_t^{a}, denoted σ^{a}, is the risk relative to the benchmark, or *active risk*. When the benchmark is cash, this is the excess risk σ^{e}. When the risk-free interest rate is constant, this is the same as the risk σ^{p}.

Information and Sharpe ratio. The (realized) *information ratio* (IR) of the portfolio relative to a benchmark is the average of the active returns $\overline{R^{\mathrm{a}}}$ over the standard deviation of the active returns σ^{a} [61],

$$\mathrm{IR} = \overline{R^{\mathrm{a}}}/\sigma^{\mathrm{a}}.$$

In the special case of a cash benchmark this is known as *Sharpe ratio* (SR) [113, 115]

$$\mathrm{SR} = \overline{R^{\mathrm{e}}}/\sigma^{\mathrm{e}}.$$

Both IR and SR are typically given using the annualized values of the return and risk [4].

2.4 Single-Period Optimization

In this section we consider optimization-based trading strategies where at the beginning of period t, using all the data available, we determine the asset portion of the current trade vector $(u_t)_{1:n}$ (or the normalized asset trades $(z_t)_{1:n}$). The cash component of the trade vector $(z_t)_{n+1}$ is then determined by the self-financing equation (2.9), once we know the realized costs. We formulate this as a convex optimization problem, which takes into account the portfolio performance over one period, the constraints on the portfolio, and investment risk (described below). The idea goes back to Markowitz [83], who was the first to formulate the choice of a portfolio as an optimization problem. (We will consider multi-period optimization in the next subsection.)

When we choose $(z_t)_{1:n}$, we do not know r_t and the other market parameters (and therefore the transaction cost function ϕ_t^{trade}), so instead we must rely on estimates of these quantities and functions. We will denote an estimate of the quantity or function Z, made at the beginning of period t (*i.e.*, when we choose $(z_t)_{1:n}$), as $\hat{Z}$. For example $\hat{\phi}_t^{\mathrm{trade}}$ is our estimate of the current period transaction cost function (which depends on the market volume and other parameters, which are predicted or estimated). The most important quantity that we estimate is the return over the current period r_t, which we denote as $\hat{r}_t$. (Return forecasts are sometimes called *signals.*) If we adopt a stochastic model of returns and other quantities, $\hat{Z}$ could be the conditional expectation of Z, given all data that is available at the beginning of period t, when the asset trades are chosen.

Before proceeding we note that most of the effort in developing a good trading algorithm goes into forming the estimates or forecasts, especially of the return r_t [27, 61]. In this chapter, however, we consider the estimates as given. Thus we focus on the question, given a set of estimates, what is a good way to trade based on them? Even though we do not focus on how the estimates should be constructed, the ideas in this chapter are useful in the development of estimates, since the value of a set of estimates can depend considerably on how they are exploited, *i.e.*, how the estimates are turned into trades. To properly assess the value of a proposed set of estimates or forecasts, we must evaluate them using a realistic simulation with a good trading algorithm.

We write our *estimated* portfolio return as

$$\hat{R}_t^{\mathrm{p}} = \hat{r}_t^T w_t + \hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t),$$

which is (2.10), with the unknown return r_t replaced with the estimate $\hat{r}_t$. The estimated active return is

$$\hat{R}_t^{\mathrm{a}} = \hat{r}_t^T (w_t - w_t^{\mathrm{b}}) + \hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t).$$

Each of these consists of a term that does not depend on the trades, plus

$$\hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t), \tag{2.12}$$

the return on the trades minus the transaction and holding costs.

2.4.1 Risk-return optimization

In a basic optimization-based trading strategy, we determine the normalized asset trades z_t by solving the optimization problem

$$\begin{array}{ll} \text{maximize} & \hat{R}_t^{\mathrm{p}} - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t \\ & \mathbf{1}^T z_t + \hat{\phi}_t^{\mathrm{trade}}(z_t) + \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) = 0, \end{array} \tag{2.13}$$

with variable z_t. Here $\psi_t : \mathbf{R}^{n+1} \to \mathbf{R}$ is a risk function, described below, and $\gamma_t > 0$ is the *risk aversion parameter*. The objective in (2.13) is called the *risk-adjusted estimated return*. The sets $\mathcal{Z}_t$ and $\mathcal{W}_t$ are the trading and holdings constraint sets, respectively, also described in more detail below. The current portfolio weight w_t is known, *i.e.*, a parameter, in the problem (2.13). The risk function, constraint sets, and estimated transaction and holding costs can all depend on the portfolio value v_t, but we suppress this dependence to keep the notation light.

To optimize performance against the risk-free interest rate or a benchmark portfolio, we replace $\hat{R}_t^{\mathrm{p}}$ in (2.13) with $\hat{R}_t^{\mathrm{e}}$ or $\hat{R}_t^{\mathrm{a}}$. By (2.12), these all have the form of a constant that does not depend on z_t, plus

$$\hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t).$$

So in all three cases we get the same trades by solving the problem

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t \\ & \mathbf{1}^T z_t + \hat{\phi}_t^{\mathrm{trade}}(z_t) + \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) = 0, \end{array} \tag{2.14}$$

with variable z_t. (We will see later that the risk functions are not the same for absolute, excess, and active return.) The objective has four terms: The first is the estimated return for the trades, the second is the estimated transaction cost, the third term is the holding cost of the post-trade portfolio, and the last is the risk of the post-trade portfolio. Note that the first two depend on the trades z_t and the last two depend on the post-trade portfolio $w_t + z_t$. (Similarly, the first constraint depends on the trades, and the second on the post-trade portfolio.)

Estimated versus realized transaction and holding costs. The asset trades we choose are given by $(z_t)_{1:n} = (z_t^\star)_{1:n}$, where $z_t^\star$ is optimal for (2.14). In dollar terms, the asset trades are $(u_t)_{1:n} = v_t(z_t^\star)_{1:n}$.

The true normalized cash trade value $(z_t)_{n+1}$ is found by the self-financing condition (2.9) from the non-cash asset trades $(z_t^\star)_{1:n}$ and the realized costs. This is not (in general) the same as $(z_t^\star)_{n+1}$, the normalized cash trade value found by solving the optimization problem (2.14). The quantity $(z_t)_{n+1}$ is the normalized cash trade value with the *realized* costs, while $(z_t^\star)_{n+1}$ is the normalized cash trade value with the *estimated* costs.

The (small) discrepancy between the realized cash trade value $(z_t)_{n+1}$ and the planned or estimated cash trade value $(z_t^\star)_{n+1}$ has an implication for the post-trade holding constraint $w_t + z_t^\star \in \mathcal{W}_t$. When we solve (2.14) we require that the post-trade portfolio with the *estimated* cash balance satisfies the constraints, which is not quite the same as requiring that the post-trade portfolio with the *realized* cash balance satisfies the constraints. The discrepancy is typically very small, since our estimation errors for the transaction cost are typically small compared to the true transactions costs, which in turn are small compared to the total portfolio value. But it should be remembered that the realized post-trade portfolio $w_t + z_t$ can (slightly) violate the constraints since we only constrain the estimated post-trade portfolio $w_t + z_t^\star$ to satisfy the constraints. (Assuming perfect trade execution, constraints relating to the asset portion of the post-trade portfolio $(w_t + z_t^\star)_{1:n}$ will hold exactly.)

Simplifying the self-financing constraint. We can simplify problem (2.14) by replacing the self-financing constraint

$$\mathbf{1}^T z_t + \hat{\phi}_t^{\mathrm{trade}}(z_t) + \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) = 0$$

with the constraint $\mathbf{1}^T z_t = 0$. In all practical cases, the cost terms are small compared to the total portfolio value, so the approximation is good. At first glance it appears that by using the simplified constraint $\mathbf{1}^T z = 0$ in the optimization problem, we are essentially ignoring the transaction and holding costs, which would not produce good results. But we still take the transaction and holding costs into account in the objective.

With this approximation we obtain the simplified problem

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T z_t - \hat{\phi}_t^{\text{trade}}(z_t) - \hat{\phi}_t^{\text{hold}}(w_t + z_t) - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & \mathbf{1}^T z_t = 0, \quad z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t. \end{array} \tag{2.15}$$

The solution $z_t^\star$ to the simplified problem slightly over-estimates the realized cash trade $(z_t)_{n+1}$, and therefore the post-trade cash balance $(w_t + z_t)_{n+1}$. The cost functions used in optimization are only estimates of what the realized values will be; in most practical cases this estimation error is much larger than the approximation introduced with the simplification $\mathbf{1}^T z_t = 0$. One small advantage (that will be useful in the multi-period trading case) is that in the optimization problem (2.15), $w_t + z_t$ is a bona fide set of weights, *i.e.*, $\mathbf{1}^T(w_t + z_t) = 1$; whereas in (2.14), $\mathbf{1}^T(w_t + z_t)$ is (typically) slightly less than one.

We can re-write the problem (2.15) in terms of the variable $w_{t+1} = w_t + z_t$, which we interpret as the post-trade portfolio weights:

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T w_{t+1} - \hat{\phi}_t^{\text{trade}}(w_{t+1} - w_t) - \hat{\phi}_t^{\text{hold}}(w_{t+1}) - \gamma_t \psi_t(w_{t+1}) \\ \text{subject to} & \mathbf{1}^T w_{t+1} = 1, \quad w_{t+1} - w_t \in \mathcal{Z}_t, \quad w_{t+1} \in \mathcal{W}_t, \end{array} \tag{2.16}$$

with variable w_{t+1}.

2.4.2 Risk measures

The risk measure ψ_t in (2.14) or (2.15) is traditionally an estimate of the variance of the return, using a stochastic model of the returns [83, 72]. But it can be any function that measures our perceived risk of holding a portfolio [53]. We first describe the traditional risk measures.

Absolute risk. Under the assumption that the returns r_t are stochastic, with covariance matrix $\Sigma_t \in \mathbf{R}^{(n+1)\times(n+1)}$, the variance of R_t^{p} is given by

$$\mathbf{var}(R_t^{\mathrm{p}}) = (w_t + z_t)^T \Sigma_t (w_t + z_t).$$

This gives the traditional quadratic risk measure for period t,

$$\psi_t(x) = x^T \Sigma_t x.$$

It must be emphasized that Σ_t is an *estimate* of the return covariance under the assumption that the returns are stochastic. It is usually assumed that the cash return (risk-free interest rate) $(r_t)_{n+1}$ is known, in which case the last row and column of Σ_t are zero.

Active risk. With the assumption that r_t is stochastic with covariance Σ_t, the variance of the active return R_t^{a} is

$$\mathbf{var}(R_t^{\mathrm{a}}) = (w_t + z_t - w_t^{\mathrm{b}})^T \Sigma_t (w_t + z_t - w_t^{\mathrm{b}}).$$

This gives the traditional quadratic active risk measure

$$\psi_t(x) = (x - w_t^{\mathrm{b}})^T \Sigma_t (x - w_t^{\mathrm{b}}).$$

When the benchmark is cash, this reduces to $x^T \Sigma_t x$, the absolute risk, since the last row and column of Σ_t are zero. In the sequel we will work with the active risk, which reduces to the absolute or excess risk when the benchmark is cash.

Risk aversion parameter. The risk aversion parameter γ_t in (2.14) or (2.15) is used to scale the relative importance of the estimated return and the estimated risk. Here we describe how the particular value $\gamma_t = 1/2$ arises in an approximation of maximizing expected growth rate, neglecting costs. Assuming that the returns r_t are independent samples from a distribution, and w is fixed, the portfolio return $R_t^{\mathrm{p}} = w^T r_t$ is a (scalar) random variable. The weight vector that maximizes the expected portfolio growth rate $\mathbf{E}\log(1+R_t^{\mathrm{p}})$ (subject to $\mathbf{1}^T w = 1$, $w \geq 0$) is called the *Kelly optimal portfolio* or *log-optimal portfolio* [69, 26]. Using the quadratic approximation of the logarithm $\log(1+a) \approx a - (1/2)a^2$ we obtain

$$\begin{array}{rcl} \mathbf{E}\log(1+R_t^{\mathrm{p}}) & \approx & \mathbf{E}\left(R_t^{\mathrm{p}} - (1/2)(R_t^{\mathrm{p}})^2\right) \\ & = & \mu^T w - (1/2) w^T(\Sigma + \mu\mu^T)w, \end{array}$$

where $\mu = \mathbf{E}\, r_t$ and $\Sigma = \mathbf{E}(r_t-\mu)(r_t-\mu)^T$ are the mean and covariance of the return r_t. Assuming that the term $\mu\mu^T$ is small compared to Σ (which is the case for realistic daily returns and covariance), the expected growth rate can be well approximated as $\mu^T w - (1/2)w^T\Sigma w$. So the choice of risk aversion parameter $\gamma_t = 1/2$ in the single-period optimization problems (2.14) or (2.15) corresponds to approximately maximizing growth rate, *i.e.*, Kelly optimal trading. In practice it is found that Kelly optimal portfolios tend to have too much risk [26], so we expect that useful values of the risk aversion parameter γ_t are bigger than 1/2.

Factor model. When the number of assets n is large, the covariance estimate Σ_t is typically specified as a low rank ('factor') component, plus a diagonal matrix,

$$\Sigma_t = F_t \Sigma_t^{\mathrm{f}} F_t^T + D_t,$$

which is called a *factor model* (for quadratic risk). Here $F_t \in \mathbf{R}^{(n+1)\times k}$ is the *factor loading matrix*, $\Sigma_t^{\mathrm{f}} \in \mathbf{R}^{k\times k}$ is an estimate of the covariance of $F^T r_t$ (the vector of *factor returns*), and $D_t \in \mathbf{R}^{(n+1)\times(n+1)}$ is a nonnegative diagonal matrix.

The number of factors k is much less than n [31] (typically, tens versus thousands). Each entry $(F_t)_{ij}$ is the loading (or *exposure*) of asset i to factor j. Factors can represent economic concepts such as industrial sectors, exposure to specific countries, accounting measures, and so on. For example, a technology factor would have loadings of 1 for technology assets and 0 for assets in other industries. But the factor loading matrices can be found using many other methods, for example by a purely data-driven analysis. The matrix D_t accounts for the additional variance in individual asset returns beyond that predicted by the factor model, known as the *idiosyncratic* *risk.*

When a factor model is used in the problems (2.14) or (2.15), it can offer a very substantial increase in the speed of solution [104, 21]. Provided the problem is formulated in such a way that the solver can exploit the factor model, the computational complexity drops from $O(n^3)$ to $O(nk^2)$ flops, for a savings of $O((n/k)^2)$. The speedup can be substantial when (as is typical) n is on the order of thousands and k on the order of tens. (Computational issues are discussed in more detail in §2.4.7.)

We now mention some less traditional risk functions that can be very useful in practice.

Transformed risk. We can apply a nonlinear transformation to the usual quadratic risk,

$$\psi_t(x) = \varphi((x - w_t^{\mathrm{b}})^T \Sigma_t (x - w_t^{\mathrm{b}})),$$

where $\varphi : \mathbf{R} \to \mathbf{R}$ is a nondecreasing function. (It should also be convex, to keep the optimization problem tractable, as we will discuss below.) This allows us to shape our aversion to different levels of quadratic risk. For example, we can take $\varphi(x) = (x-a)_+$. In this case the transformed risk assesses no cost for quadratic risk levels up to a. This can be useful to hit a target risk level, or to be maximally aggressive in seeking returns, up to the risk threshold a. Another option is $\varphi(x) = \exp(x/\eta)$, where $\eta > 0$ is a parameter. This assesses a strong cost to risks substantially larger than η, and is closely related to risk aversion used in stochastic optimization.

The solution of the optimization problem (2.14) with transformed risk is the same as the solution with the traditional risk function, but with a different value of the risk aversion parameter. So we can think of transformed risk aversion as a method to automatically tune the risk aversion parameter, increasing it as the risk increases.

Worst-case quadratic risk. We now move beyond the traditional quadratic risk to create a risk function that is more robust to unpredicted changes in market conditions. We define the *worst-case* risk for portfolio x as

$$\psi_t(x) = \max_{i=1,\ldots,M} (x - w_t^{\mathrm{b}})^T \Sigma_t^{(i)} (x - w_t^{\mathrm{b}}).$$

Here $\Sigma^{(i)}$, $i = 1, \dots, M$, are M given covariance matrices; we refer to i as the *scenario*. We can motivate the worst-case risk by imagining that the returns are generated from one of M distributions, with covariances $\Sigma^{(i)}$ depending on which scenario occurs. In each period, we do not know, and do not attempt to predict, which scenario will occur. The worst-case risk is the largest risk under the M scenarios.

If we estimate the probabilities of occurrence of the scenarios, and weight the scenario covariance matrices by these probabilities, we end up back with a single quadratic risk measure, the weighted sum of the scenario covariances. It is critical that we combine them using the maximum, and not a weighted sum. (Although other nonlinear combining functions would also work.) We should think of the scenarios as describing situations that could arise, but that we cannot or do not attempt to predict.

The scenario covariances $\Sigma^{(i)}$ can be found by many reasonable methods. They can be empirical covariances estimated from realized (past) returns conditioned on the scenario, for example, high or low market volatility, high or low interest rates, high or low oil prices, and so on [92]. They could be an analyst's best guess for what the asset covariance would be in a situation that could occur.

2.4.3 Forecast error risk

The risk measures considered above attempt to model the period to period variation in asset returns, and the associated period to period variation in the portfolio return they induce. In this subsection we consider terms that take into account errors in our prediction of return and covariance. (The same ideas can be applied to other parameters that we estimate, like volume.) Estimation errors can significantly impact the resulting portfolio weights, resulting in poor out-of-sample performance [67, 93, 33, 68, 40, 46, 72, 5].

Return forecast error risk. We assume our forecasts of the return vector $\hat{r}$ are uncertain: Any forecast $\hat{r} + \delta$ with $|\delta| \leq \rho$ and $\rho \in \mathbf{R}^n$ is possible and consistent with what we know. In other words, ρ is a vector of uncertainties on our return prediction $\hat{r}$. If we are confident in our (nominal) forecast of the return of asset i, we take ρ_i

small; conversely large ρ_i means that we are not very confident in our forecast. The uncertainty in return forecast is readily interpreted when annualized; for example, our uncertain return forecast for an asset might be described as $6\% \pm 2\%$, meaning any forecast return between 4% and 8% is possible.

The post-trade estimated return is then $(\hat{r}_t+\delta_t)^T(w_t+z_t)$; we define the minimum of this over $|\delta| \leq \rho$ as the *worst-case return forecast*. It is easy to see what the worst-case value of δ is: If we hold a long position, the return (for that asset) should take its minimum value $\hat{r}_i + \rho_i$; if we hold a short position, it should take its maximum allowed value $\hat{r}_i - \rho_i$. The worst-case return forecast has the value

$$\hat{R}_t^{\text{wc}} = \hat{r}_t^T(w_t + z_t - w_t^{\text{b}}) - \rho^T|w_t + z_t - w_t^{\text{b}}|.$$

The first term here is our original estimate (including the constant terms we neglect in (2.14) and (2.15)); the second term (which is always nonpositive) is the worst possible value of our estimated active return over the allowed values of δ. It is a risk associated with forecast uncertainty. This gives

$$\psi_t(x) = \rho^T|x - w_t^{\text{b}}|. \qquad (2.17)$$

(This would typically be added to a traditional quadratic risk measure.) This term is a weighted ℓ_1-norm of the deviation from the weights, and encourages weights that deviate sparsely from the benchmark, *i.e.*, weights with some or many entries equal to those of the benchmark [117, 48, 64, 76].

Covariance forecast error risk. In a similar way we can add a term that corresponds to risk of errors in forecasting the covariance matrix in a traditional quadratic risk model. As an example, suppose that we are given a nominal covariance matrix Σ, and consider the *perturbed covariance matrix*

$$\Sigma^{\text{pert}} = \Sigma + \Delta,$$

where Δ is a symmetric perturbation matrix with

$$|\Delta_{ij}| \leq \kappa \left(\Sigma_{ii}\Sigma_{jj}\right)^{1/2}, \tag{2.18}$$

where $\kappa \in [0, 1)$ is a parameter. This perturbation model means that the diagonal entries of covariance can change by the fraction κ; ignoring the change in the diagonal entries, the asset correlations can change by up to (roughly) κ. The value of κ depends on our confidence in the covariance matrix; reasonable values are $\kappa = 0.02$, 0.05, or more.

With $v = x - w_t^{\mathrm{b}}$, the maximum (worst-case) value of the quadratic risk over this set of perturbations is given by

$$\begin{array}{rcl}
\displaystyle\max_{|\Delta_{ij}|\leq\kappa(\Sigma_{ii}\Sigma_{jj})^{1/2}} v^T(\Sigma^{\mathrm{pert}})v & = & \displaystyle\max_{|\Delta_{ij}|\leq\kappa(\Sigma_{ii}\Sigma_{jj})^{1/2}} v^T(\Sigma+\Delta)v \\
& = & \displaystyle v^T\Sigma v + \max_{|\Delta_{ij}|\leq\kappa(\Sigma_{ii}\Sigma_{jj})^{1/2}} \sum_{ij} v_i v_j \Delta_{ij} \\
& = & \displaystyle v^T\Sigma v + \kappa \sum_{ij} |v_i v_j| (\Sigma_{ii}\Sigma_{jj})^{1/2} \\
& = & \displaystyle v^T\Sigma v + \kappa \left(\sum_i \Sigma_{ii}^{1/2}|v_i|\right)^2 .
\end{array}$$

This shows that the worst-case covariance, over all perturbed covariance matrices consistent with our risk forecast error assumption (2.18), is given by

$$\psi_t(x) = (x - w_t^{\mathrm{b}})^T\Sigma(x - w_t^{\mathrm{b}}) + \kappa \left(\sigma^T|x - w_t^{\mathrm{b}}|\right)^2, \tag{2.19}$$

where $\sigma = (\Sigma_{11}^{1/2}, \ldots, \Sigma_{nn}^{1/2})$ is the vector of asset volatilities. The first term is the usual quadratic risk with the nominal covariance matrix; the second term can be interpreted as risk associated with covariance forecasting error [64, 76]. It is the square of a weighted ℓ_1-norm of the deviation of the weights from the benchmark. (With cash benchmark, this directly penalizes large leverage.)

2.4.4 Holding constraints

Holding constraints restrict our choice of normalized post-trade portfolio $w_t + z_t$. Holding constraints may be surrogates for constraints on w_{t+1}, which we cannot constrain directly since it depends on the unknown returns. Usually returns are small and w_{t+1} is close to $w_t + z_t$, so constraints on $w_t + z_t$ are good approximations for constraints on w_{t+1}. Some types of constraints always hold exactly for w_{t+1} when they hold for $w_t + z_t$.

Holding constraints may be mandatory, imposed by law or the investor, or discretionary, included to avoid certain undesirable portfolios. We discuss common holding constraints below. Depending on the specific situation, each of these constraints could be imposed on the *active* holdings $w_t + z_t - w_t^{\mathrm{b}}$ instead of the absolute holdings $w_t + z_t$, which we use here for notational simplicity.

Long only. This constraint requires that only long asset positions are held,

$$w_t + z_t \geq 0.$$

If only the assets must be long, this becomes $(w_t + z_t)_{1:n} \geq 0$. When a long only constraint is imposed on the post-trade weight $w_t + z_t$, it automatically holds on the next period value $(\mathbf{1} + r_t) \circ (h_t + z_t)$, since $\mathbf{1} + r_t \geq 0$.

Leverage constraint. The leverage can be limited with the constraint

$$\|(w_t + z_t)_{1:n}\|_1 \leq L^{\max},$$

which requires the post-trade portfolio leverage to not exceed $L^{\max}$. (Note that the leverage of the next period portfolio can be slightly larger than $L^{\max}$, due to the returns over the period.)

Limits relative to asset capitalization. Holdings are commonly limited so that the investor does not own too large a portion of the company total value. Let C_t

denote the vector of asset capitalization, in dollars. The constraint

$$(w_t + z_t)_i \leq \delta \circ C_t / v_t,$$

where $\delta \geq 0$ is a vector of fraction limits, and / is interpreted elementwise, limits the long post-trade position in asset i to be no more than the fraction δ_i of the capitalization. We can impose a similar limit on short positions, relative to asset capitalization, total outstanding short value, or some combination.

Limits relative to portfolio. We can limit our holdings in each asset to lie between a minimum and a maximum fraction of the portfolio value,

$$-w^{\min} \leq w_t + z_t \leq w^{\max},$$

where $w^{\min}$ and $w^{\max}$ are nonnegative vectors of the maximum short and long allowed fractions, respectively. For example with $w^{\max} = w^{\min} = (0.05)\mathbf{1}$, we are not allowed to hold more than 5% of the portfolio value in any one asset, long or short.

Minimum cash balance. Often the cash balance must stay above a minimum dollar threshold $c_{\min}$ (which can be negative). We express a minimum cash balance as the constraint

$$(w_t + z_t)_{n+1} \geq c_{\min} / v_t.$$

This constraint can be slightly violated by the realized values, due to our error in estimation of the costs.

No-hold constraints. A no-hold constraint on asset i forbids holding a position in asset i, *i.e.*,

$$(w_t + z_t)_i = 0.$$

β-neutrality. A β-neutral portfolio is one whose return R^{p} is uncorrelated with the benchmark return R^{b}, according to our estimate Σ_t of $\mathbf{cov}(r_t)$. The constraint that

$w_t + z_t$ be β neutral takes the form

$$(w_t^{\mathrm{b}})^T \Sigma_t (w_t + z_t) = 0.$$

Factor neutrality. In the factor covariance model, the estimated portfolio risk σ_i^{F} due to factor i is given by

$$\left(\sigma_i^{\mathrm{F}}\right)^2 = (w_t + z_t)^T (F_t)_i (\Sigma_t^{\mathrm{f}})_{ii} (F_t)_i^T (w_t + z_t).$$

The constraint that the portfolio be neutral to factor i means that $\sigma_i^{\mathrm{F}} = 0$, which occurs when

$$(F_t)_i^T (w_t + z_t) = 0.$$

Stress constraints. Stress constraints protect the portfolio against unexpected changes in market conditions. Consider scenarios $1, \ldots, K$, each representing a market shock event such as a sudden change in oil prices, a general reversal in momentum, or a collapse in real estate prices. Each scenario i has an associated (estimated) return c_i. The c_i could be based on past occurrences of scenario i or predicted by analysts if scenario i has never occurred before.

Stress constraints take the form

$$c_i^T (w_t + z_t) \geq R^{\mathrm{min}},$$

i.e., the portfolio return in scenario i is above R^{min}. (Typically R^{min} is negative; here we are limiting the decrease in portfolio value should scenario i actually occur.) Stress constraints are related to chance constraints such as value at risk in the sense that they restrict the probability of large losses due to shocks.

Liquidation loss constraint. We can bound the loss of value incurred by liquidating the portfolio over T^{liq} periods. A constraint on liquidation loss will deter the optimizer from investing in illiquid assets. We model liquidation as the transaction cost to trade h^+ over T^{liq} periods. If we use the transaction cost estimate $\hat{\phi}$ for all periods, the optimal schedule is to trade $(w_t + z_t)/T^{\mathrm{liq}}$ each period. The constraint that the liquidation loss is no more than the fraction δ of the portfolio value is given by

$$T^{\mathrm{liq}} \hat{\phi}_t^{\mathrm{trade}}((w_t + z_t)/T^{\mathrm{liq}}) \leq \delta.$$

(For optimization against a benchmark, we replace this with the cost to trade the portfolio to the benchmark over T^{liq} periods.)

Concentration limit. As an example of a non-traditional constraint, we consider a *concentration limit*, which requires that no more than a given fraction ω of the portfolio value can be held in some given fraction (or just a specific number K) of assets. This can be written as

$$\sum_{i=1}^{K} (w_t + z_t)_{[i]} \leq \omega,$$

where the notation $a_{[i]}$ refers to the ith largest element of the vector a. The left-hand side is the sum of the K largest post-trade positions. For example with $K = 20$ and $\omega = 0.4$, this constraint prohibits holding more than 40% of the total value in any 20 assets. (It is not well known that this constraint is convex, and indeed, easily handled; see [21, §3.2.3]. It is easily extended to the case where K is not an integer.)

2.4.5 Trading constraints

Trading constraints restrict the choice of normalized trades z_t. Constraints on the non-cash trades $(z_t)_{1:n}$ are exact (since we assume that our trades are executed in full), while constraints on the cash trade $(z_t)_{n+1}$ are approximate, due to our estimation of the costs. As with holding constraints, trading constraints may be mandatory or discretionary.

Turnover limit. The turnover of a portfolio in period t is given by $\|(z_t)_{1:n}\|_1/2$. It is common to limit the turnover to a fraction δ (of portfolio value), *i.e.*,

$$\|(z_t)_{1:n}\|_1/2 \leq \delta.$$

Limits relative to trading volume. Trades in non-cash assets may be restricted to a certain fraction δ of the current period market volume V_t (estimate),

$$|(z_t)_{1:n}| \leq \delta(V_t/v_t),$$

where the division on the right-hand side means elementwise.

No-buy, sell, or trade restriction. A no-buy restriction on asset i imposes the constraint

$$(z_t)_i \leq 0,$$

while a no-sell restriction imposes the constraint

$$(z_t)_i \geq 0.$$

A no-trade restriction imposes both a no-buy and no-sell restriction.

2.4.6 Soft constraints

Any of the constraints on holdings or transactions can be made *soft*, which means that are not strictly enforced. We explain this in a general setting. For a vector equality constraint $h(x) = 0$ on the variable or expression x, we replace it with a term subtracted the objective of the form $\gamma\|h(x)\|_1$, where $\gamma > 0$ is the *priority* of the constraint. (We can generalize this to $\gamma^T|h(x)|$, with γ a vector, to give different priorities to the different components of $h(x)$.) In a similar way we can replace an inequality constraint $h(x) \leq 0$ with a term, subtracted from the objective, of the form $\gamma^T(h(x))_+$, where $\gamma > 0$ is a vector of priorities. Replacing the hard constraints with these *penalty* terms results in *soft constraints*. For large enough values of the priorities, the constraints hold exactly; for smaller values, the constraints are (roughly speaking) violated only when they need to be.

As an example, we can convert a set of factor neutrality constraints $F_t^T(w_t+z_t) = 0$ to soft constraints, by subtracting a term $\gamma \|F_t^T(w_t + z_t)\|_1$ from the objective, where $\gamma > 0$ is the priority. For larger values of γ factor neutrality $F_t^T(w_t + z_t) = 0$ will hold (exactly, when possible); for smaller values some factor exposures can become nonzero, depending on other objective terms and constraints.

2.4.7 Convexity

The portfolio optimization problem (2.14) can be solved quickly and reliably using readily available software so long as the problem is convex. This requires that the risk and estimated transaction and holding cost functions are convex, and the trade and holding constraint sets are convex. All the functions and constraints discussed above are convex, except for the self-financing constraint

$$\mathbf{1}^T z_t + \hat{\phi}_t^{\text{trade}}(z_t) + \hat{\phi}_t^{\text{hold}}(w_t + z_t) = 0,$$

which must be relaxed to the inequality

$$\mathbf{1}^T z_t + \hat{\phi}_t^{\text{trade}}(z_t) + \hat{\phi}_t^{\text{hold}}(w_t + z_t) \leq 0.$$

The inequality will be tight at the optimum of (2.14). Alternatively, the self-financing constraint can be replaced with the simplified version $\mathbf{1}^T z_t = 0$ as in problem (2.15).

Solution times. The SPO problems described above, except for the multi-covariance risk model, can be solved using standard interior-point methods [100] with a complexity $O(nk^2)$ flops, where n is the number of assets and k is the number of factors. (Without the factor model, we replace k with n.) The coefficient in front is on the order of 100, which includes the interior-point iteration count and other computation. This should be the case even for complex leverage constraints, the 3/2 power transaction costs, limits on trading and holding, and so on.

This means that a typical current single core (or thread) of a processor can solve an SPO problem with 1500 assets and 50 factors in under one half second (based conservatively on a computation speed of 1G flop/sec). This is more than fast enough to use the methods to carry out trading with periods on the order of a second. But the speed is still very important even when the trading is daily, in order to carry out back-testing. For daily trading, one year of back-testing, around 250 trading days, can be carried out in a few minutes or less. A generic 32 core computer, running 64 threads, can carry out a back-test on five years of data, with 64 different choices of parameters (see below), in under 10 minutes. This involves solving 80000 convex optimization problems. All of these times scale linearly with the number of assets, and quadratically with the number of factors. For a problem with, say, 4500 assets and 100 factors, the computation times would be around 12× longer. Our estimates are conservatively based on a computation speed of 1G flop/sec; for these or larger problems multi-threaded optimized linear algebra routines can achieve 100G flop/sec, making the back-testing 100× faster.

We mention one special case that can be solved much faster. If the objective is quadratic, which means that the risk and costs are quadratic functions, and the only constraints are linear equality constraints (*e.g.*, factor neutrality), the problem can be solved with the same $O(nk^2)$ complexity, but the coefficient in front is closer to 2, around 50 times faster than using an interior-point method.

Custom solvers, or solvers targeted to specific platforms like GPUs, can solve SPO problems much faster [103]. For example the first order operator-splitting method implemented in POGS [50] running on a GPU, can solve extremely large SPO problems. POGS can solve a problem with 100000 assets and 1000 factors (which is much larger than any practical problem) in a few seconds or less. At the other extreme, code generation systems like CVXGEN [84] can solve smaller SPO problems with stunning speed; for example a problem with 30 assets in well under one millisecond.

Problem specification. New frameworks for convex optimization such as CVX [50], CVXPY [41], and Convex.jl [118], based on the idea of disciplined convex programming (DCP) [59], make it very easy to specify and modify the SPO problem in just a handful of lines of easy to understand code. These frameworks make it easy to experiment with non-standard trading and holding constraints, or risk and cost functions.

Nonconvexity. The presence of nonconvex constraints or terms in the optimization problem greatly complicates its solution, making its solution time much longer, and sometimes very much longer. This may not be a problem in the production trading engine that determines one trade per day, or per hour. But nonconvexity makes back-testing much slower at the least, and in many cases simply impractical. This greatly reduces the effectiveness of the whole optimization-based approach. For this reason, nonconvex constraints or terms should be strenuously avoided.

Nonconvex constraints generally arise only when someone who does not understand this adds a reasonable sounding constraint, unaware of the trouble he or she is making. As an example, consider imposing a minimum trade condition, which states that if $(z_t)_i$ is nonzero, it must satisfy $|(z_t)_i| \geq \epsilon$, where $\epsilon > 0$. This constraint seems reasonable enough, but makes the problem nonconvex. If the intention was to achieve sparse trading, or to avoid many very small trades, this can be accomplished (in a far better way) using convex constraints or cost terms.

Other examples of nonconvex constraints (that should be avoided) include limits on the number of assets held, minimum values of nonzero holdings, or restricting trades to be integer numbers of share lots, or restricting the total number of assets we can trade. The requirement that we must trade integer numbers of shares is also nonconvex, but irrelevant for any practical portfolio. The error induced by rounding our trade lists (which contain real numbers) to an integer number of shares is negligible for reasonably sized portfolios.

While nonconvex constraints and objective terms should be avoided, and are generally not needed, it is possible to handle many of them using simple powerful heuristics, such as solving a relaxation, fixing the nonconvex terms, and then solving the convex problem again [42]. As a simple example of this approach, consider the minimum nonzero trade requirement $|(z_t)_i| \geq \epsilon$ for $(z_t)_i \neq 0$. We first solve the SPO problem without this constraint, finding a solution $\tilde{z}$. We use this tentative trade vector to determine which entries of z will be zero, negative, or positive (*i.e.*, which assets we hold, sell, or buy). We now impose these sign constraints on the trade vector: We require $(z_t)_i = 0$ if $(\tilde{z}_t)_i = 0$, $(z_t)_i \geq 0$ if $(\tilde{z}_t)_i > 0$, and $(z_t)_i \leq 0$ if $(\tilde{z}_t)_i < 0$. We solve the SPO again, with these sign constraints, and the minimum trade constraints as well, which are now linear, and therefore convex. This simple method will work very well in practice.

As another example, suppose that we are limited to make at most K nonzero trades in any given period. A very simple scheme, based on convex optimization, will work extremely well. First we solve the problem ignoring the limit, and possibly with an additional ℓ_1 transaction cost added in, to discourage trading. We take this trade list and find the K largest trades (buy or sell). We then add the constraint to our problem that we will only trade these assets, and we solve the portfolio optimization problem again, using only these trades. As in the example described above, this approach will yield extremely good, if not optimal, trades. This approximation will have no effect on the real metrics of interest, *i.e.*, the portfolio performance.

There is generally no need to solve the nonconvex problem globally, since this greatly increases the solve time and delivers no practical benefit in terms of trading performance. The best method for handling nonconvex problems in portfolio optimization is to avoid them.

2.4.8 Using single-period optimization

The idea. In this subsection we briefly explain, at a high level, how the SPO trading algorithm is used in practice. We do not discuss what is perhaps the most critical part, the return (and other parameter) estimates and forecasts. Instead, we assume the forecasts are given, and focus on how to use SPO to exploit them.

In SPO, the parameters that appear in the transaction and holding costs can be inspired or motivated by our estimates of what their true values will be, but it is better to think of them as ‘knobs’ that we turn to achieve trading behavior that we like (see, *e.g.*, [35, § 8], [66, 39, 76]), as verified by back-testing, what-if simulation, and stress-testing.

As a crude but important example, we can scale the entire transaction cost function ϕ_t^{trade} by a *trading aversion factor* γ^{trade}. (The name emphasizes the analogy with the risk aversion parameter, which scales the risk term in the objective.) Increasing the trading aversion parameter will deter trading or reduce turnover; decreasing it will increase trading and turnover. We can even think of $1/\gamma^{\text{trade}}$ as the number of periods over which we will amortize the transaction cost we incur [60]. As a more sophisticated example, the transaction cost parameters a_t, meant to model bid-ask spread, can be scaled up or down. If we increase them, the trades become more sparse, *i.e.*, there are many periods in which we do not trade each asset. If we scale the 3/2-power term, we encourage or discourage large trades. Indeed, we could add a quadratic transaction term to the SPO problem, not because we think it is a good model of transaction costs, but to discourage large trades even more than the 3/2-power term does. Any SPO variation, such as scaling certain terms, or adding new ones, is assessed by back-testing and stress-testing.

The same ideas apply to the holding cost. We can scale the holding cost rates by a positive *holdings aversion parameter* γ^{hold} to encourage, or discourage, holding positions that incur holding costs, such as short positions. If the holding cost reflects the cost of holding short positions, the parameter γ^{hold} scales our aversion to holding short positions. We can modify the holding cost by adding a quadratic term of the short positions $\kappa^T(w_t + z_t)_-^2$, (with the square interpreted elementwise and $\kappa \geq 0$), not because our actual borrow cost rates increase with large short positions, but to send the message to the SPO algorithm that we wish to avoid holding large short positions.

As another example, we can add a liquidation loss term to the holding cost, with a scale factor to control its effect. We add this term not because we intend to liquidate the portfolio, but to avoid building up large positions in illiquid assets. By increasing the scale factor for the liquidation loss term, we discourage the SPO algorithm from holding illiquid positions.

Trade, hold, and risk aversion parameters. The discussion above suggests that we modify the objective in (2.15) with scaling parameters for transaction and holding costs, in addition to the traditional risk aversion parameter, which yields the SPO problem

$$\begin{array}{ll}\text{maximize} & \Big(\hat{r}_t^T z_t - \gamma_t^{\text{trade}} \hat{\phi}_t^{\text{trade}}(z_t) \\ & \quad - \gamma_t^{\text{hold}} \hat{\phi}_t^{\text{hold}}(w_t + z_t) - \gamma_t^{\text{risk}} \psi_t(w_t + z_t)\Big) \\ \text{subject to} & \mathbf{1}^T z_t = 0, \quad z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t. \end{array} \tag{2.20}$$

where γ_t^{trade}, γ_t^{hold}, and γ_t^{risk} are positive parameters used to scale the respective costs. These parameters are sometimes called *hyper-parameters*, which emphasizes the analogy to the hyper-parameters used when fitting statistical models to data. The hyper-parameters are ‘knobs’ that we ‘turn’ (*i.e.*, choose or change) to obtain good performance, which we evaluate by back-testing. We can have even more than three hyper-parameters, which scale individual terms in the holding and transaction costs. The choice of hyper-parameters can greatly affect the performance of the SPO method. They should be chosen using back-testing, what-if testing, and stress-testing.

This style for using SPO is similar to how optimization is used in many other applied areas, for example control systems or machine learning. In machine learning, for example, the goal is to find a model that makes good predictions on new data. Most methods for constructing a model use optimization to minimize a so-called loss function, which penalizes not fitting the observed data, plus a regularizer, which penalizes model sensitivity or complexity. Each of these functions is inspired by a (simplistic) theoretical model of how the data were generated. But the final choice of these functions, and the (hyper-parameter) scale factor between them, is done by out-of-sample validation or cross validation, *i.e.*, testing the model on data it has not seen [52]. For general discussion of how convex optimization is used in this spirit, in applications such as control or estimation, see [21].

Judging value of forecasts. In this chapter we do not consider forecasts, which of course are critically important in trading. The most basic test of a new proposed return estimate or forecast is that it does well predicting returns. This is typically judged using a simple model that evaluates Sharpe ratio or information ratio, implicitly ignoring all portfolio constraints and costs. If a forecast fails these simple SR or IR tests, it is unlikely to be useful in a trading algorithm.

But the true value of a proposed estimate or forecast in the context of multi-period trading can be very different from what is suggested by the simple SR or IR prediction tests, due to costs, portfolio constraints, and other issues. A new proposed forecast should be judged in the context of the portfolio constraints, other forecasts (say, of volume), transaction costs, holding costs, trading constraints, and choice of parameters such as risk aversion. This can be done using simulation, carrying out back-tests, what-if simulations, and stress-tests, in each case varying the parameters to achieve the best performance. The result of this testing is that the forecast might be less valuable (the usual case) or more valuable (the less usual case) than it appeared from the simple SR and IR tests. One consequence of this is that the true value of a forecast can depend considerably on the type and size of the portfolio being traded; for example, a forecast could be very valuable for a small long-short portfolio with modest leverage, and much less valuable for a large long-only portfolio.

2.5 Multi-Period Optimization

2.5.1 Motivation

In this section we discuss optimization-based strategies that consider information about multiple periods when choosing trades for the current period. Before delving into the details, we should consider what we hope to gain over the single-period approach. Predicting the returns for the current period is difficult enough. Why attempt to forecast returns in future periods?

One reason is to better account for transaction costs. In the absence of transaction cost (and other limitations on trading), a greedy strategy that only considers one period at a time is optimal, since performance for the current period does not depend on previous holdings. However in any realistic model current holdings strongly affect whether a return prediction can be profitably acted on. We should therefore consider whether the trades we make in the current period put us in a good or bad position to trade in future periods. While this idea can be incorporated into single-period optimization, it is more naturally handled in multi-period optimization.

For example, suppose our single period optimization-based strategy tells us to go very long in a rarely traded asset. We may not want to make the trade because we know that unwinding the position will incur large transaction costs. The single-period problem models the cost of moving into the position, but not the cost of moving out of it. To model the fact that we will over time revert positions towards the benchmark, and thus must eventually sell the positions we buy, we need to model time beyond the current period. (One standard trick in single-period optimization is to double the transaction cost, which is then called the *round-trip cost*.)

Another advantage of multi-period optimization is that it naturally handles multiple, possibly conflicting return estimates on different time scales (see, *e.g.*, [55, 101]). As an example, suppose we predict that a return will be positive over a short period, but over a longer period it will be negative. The first prediction might be relevant for only a day, while the second for a month or longer. In a single-period optimization framework, it is not clear how to account for the different time scales when blending the return predictions. Combining the two predictions would likely cancel them, or have us move according to whichever prediction is larger. But the resulting behavior could be quite non-optimal. If the trading cost is high, taking no action is likely the right choice, since we will have to reverse any trade based on the fast prediction as we follow the slow prediction in future periods. If the trading cost is low, however, the right choice is to follow the fast prediction, since unwinding the position is cheap. This behavior falls naturally out of a multi-period optimization, but is difficult to capture in a single-period problem.

There are many other situations where predictions over multiple periods, as opposed to just the current period, can be taken advantage of in multi-period optimization. We describe a few of them here.

- *Signal decay and time-varying returns predictions.* Generalizing the discussion above on fast versus slow signals, we may assign an exponential *decay-rate* to every return prediction signal. (This can be estimated historically, for example, by fitting an auto-regressive model to the signal values.) Then it is easy to compute return estimates at any time scale. The decay in prediction accuracy is also called mean-reversion or alpha decay (see, *e.g.*, [27, 60, 55]).

- *Known future changes in volatility or risk.* If we know that a future event will increase the risk, we may want to exit some of the risky positions in advance. In MPO, trading towards a lower risk position starts well before the increase in risk, trading it off with the transaction costs. In SPO, (larger) trading to a lower risk position occurs only once the risk has increased, leading to larger transaction costs. Conversely, known periods of low risk can be exploited as well.

- *Changing constraints over multiple periods.* As an example, assume we want to de-leverage the portfolio over multiple periods, *i.e.*, reduce the leverage constraint $L^{\max}$ over some number of periods to a lower value. If we use a multi-period optimization framework we will likely incur lower trading cost than by some ad-hoc approach, while still exploiting our returns predictions.

- *Known future changes in liquidity or volume.* Future volume or volatility predictions can be exploited for transaction cost optimization, for example by delaying some trades until they will be cheaper. Market volumes V_t have much better predictability than market returns.

- *Setting up, shutting down, or transferring a portfolio.* These transitions can all be handled naturally by MPO, with a combination of constraints and objective terms changing over time.

2.5.2 Multi-period optimization

In multi-period optimization, we choose the current trade vector z_t by solving an optimization problem over a *planning horizon* that extends H periods into the future,

$$t,\ t+1,\ \ldots,\ t+H-1.$$

(Single-period optimization corresponds to the case $H = 1$.)

Many quantities at times $t, t+1, \ldots, t+H-1$ are unknown at time t, when the optimization problem is solved and the asset trades are chosen, so as in the single-period case, we will estimate them. For any quantity or function Z, we let $\hat{Z}_{\tau|t}$ denote our estimate of Z_τ given all information available to us at the beginning of period t. (Presumably $\tau \geq t$; otherwise we can take $\hat{Z}_{\tau|t} = Z_\tau$, the realized value of Z at time τ.) For example $\hat{r}_{t|t}$ is the estimate made at time t of the return at time t (which we denoted $\hat{r}_t$ in the subsection on single-period optimization); $\hat{r}_{t+2|t}$ is the estimate made at time t of the return at time $t+2$.

We can develop a multi-period optimization problem starting from (2.14). Let

$$z_t, \; z_{t+1}, \; \ldots, \; z_{t+H-1}$$

denote our sequence of planned trades over the horizon. A natural objective is the total risk-adjusted return over the horizon,

$$\sum_{\tau=t}^{t+H-1} \left(\hat{r}_{\tau|t}^T (w_\tau + z_\tau) - \gamma_\tau \psi_\tau (w_\tau + z_\tau) - \hat{\phi}_\tau^{\text{hold}} (w_\tau + z_\tau) - \hat{\phi}_\tau^{\text{trade}} (z_\tau) \right).$$

(This expression drops a constant that does not depend on the trades, and handles absolute or active return.) In this expression, w_t is known, but $w_{t+1}, \ldots, w_{t+H}$ are not, since they depend on the trades $z_t, \ldots, z_{t+H-1}$ (which we will choose) and the unknown returns, via the dynamics equation (2.11),

$$w_{t+1} = \frac{1}{1 + R_t^{\text{p}}} (\mathbf{1} + r_t) \circ (w_t + z_t),$$

which propagates the current weight vector to the next one, given the trading and return vectors. (This true dynamics equation ensures that if $\mathbf{1}^T w_t = 1$, we have $\mathbf{1}^T w_{t+1} = 1$.)

In adding the risk terms $\gamma_\tau \psi_t (w_\tau + z_\tau)$ in this objective, we are implicitly relying on the idea that the returns are independent random variables, so the variance of the sum is the sum of the variances. We can also interpret $\gamma_\tau \psi_\tau (w_\tau + z_\tau)$ as cost terms that discourage us from holding certain portfolios.

Simplifying the dynamics. We now make a simplifying approximation: For the purpose of propagating w_t and z_t to w_{t+1} in our planning exercise, we will assume

$R_t^{\mathrm{p}} = 0$ and $r_t = 0$ (*i.e.*, that the one period returns are small compared to one). This results in the much simpler dynamics equation $w_{t+1} = w_t + z_t$. With this approximation, we must add the constraints $\mathbf{1}^T z_t = 0$ to ensure that the weights in our planning exercise add to one, *i.e.*, $\mathbf{1}^T w_\tau = 1$, $\tau = t+1, \ldots, t+H$. So we will impose the constraints

$$\mathbf{1}^T z_\tau = 0, \quad \tau = t+1, \ldots, t+H-1.$$

The current portfolio weights w_t are given, and satisfy $\mathbf{1}^T w_t = 1$; we get that $\mathbf{1}^T w_\tau = 1$ for $\tau = t+1, \ldots, t+H$ due to the constraints. (Implications of the dynamics simplification are discussed below.)

Multi-period optimization problem. With the dynamics simplification we arrive at the MPO problem

$$\begin{array}{ll} \text{maximize} & \sum_{\tau=t}^{t+H-1} \Big(\hat{r}_{\tau|t}^T (w_\tau + z_\tau) - \gamma_\tau \psi_\tau (w_\tau + z_\tau) \\ & \qquad - \hat{\phi}_\tau^{\mathrm{hold}}(w_\tau + z_\tau) - \hat{\phi}_\tau^{\mathrm{trade}}(z_\tau) \Big) \\ \text{subject to} & \mathbf{1}^T z_\tau = 0, \quad z_\tau \in \mathcal{Z}_\tau, \quad w_\tau + z_\tau \in \mathcal{W}_\tau, \\ & w_{\tau+1} = w_\tau + z_\tau, \quad \tau = t, \ldots, t+H-1, \end{array} \tag{2.21}$$

with variables $z_t, z_{t+1}, \ldots, z_{t+H-1}$ and $w_{t+1}, \ldots, w_{t+H}$. Note that w_t is not a variable, but the (known) current portfolio weights. When $H = 1$, the multi-period problem reduces to the simplified single-period problem (2.15). (We can ignore the constant $\hat{r}_{t|t}^T w_t$, which does not depend on the variables, that appears in (2.21) but not (2.15).)

Using $w_{\tau+1} = w_\tau + z_\tau$ we can eliminate the trading variables z_τ to obtain the equivalent problem

$$\begin{array}{ll} \text{maximize} & \sum_{\tau=t+1}^{t+H} \Big(\hat{r}_{\tau|t}^T w_\tau - \gamma_\tau \psi_\tau (w_\tau) \\ & \qquad - \hat{\phi}_\tau^{\mathrm{hold}}(w_\tau) - \hat{\phi}_\tau^{\mathrm{trade}}(w_\tau - w_{\tau-1}) \Big) \\ \text{subject to} & \mathbf{1}^T w_\tau = 1, \quad w_\tau - w_{\tau-1} \in \mathcal{Z}_\tau, \quad w_\tau \in \mathcal{W}_\tau, \\ & \qquad \tau = t+1, \ldots, t+H, \end{array} \tag{2.22}$$

with variables $w_{t+1}, \ldots, w_{t+H}$, the planned weights over the next H periods. This is the multi-period analog of (2.16).

Both MPO formulations (2.21) and (2.22) are convex optimization problems, provided the transaction cost, holding cost, risk functions, and trading and holding constraints are all convex.

Interpretation of MPO. The MPO problems (2.21) or (2.22) can be interpreted as follows. The variables constitute a *trading plan*, *i.e.*, a set of trades to be executed over the next H periods. Solving (2.21) or (2.22) is forming a trading plan, based on forecasts of critical quantities over the planning horizon, and some simplifying assumptions. We do not intend to execute this sequence of trades, except for the first one z_t. It is reasonable to ask then why we optimize over the future trades $z_{t+1}, \ldots, z_{t+H-1}$, since we do not intend to execute them. The answer is simple: We optimize over them as part of a *planning exercise*, just to be sure we don't carry out any trades now (*i.e.*, z_t) that will put us in a bad position in the future. The idea of carrying out a planning exercise, but only executing the current action, occurs and is used in many fields, such as automatic control (where it is called model predictive control, MPC, or receding horizon control) [75, 85], supply chain optimization [32], and others. Applications of MPC in finance include [63, 19, 8, 24, 101].

About the dynamics simplification. Before proceeding let us discuss the simplification of the dynamics equation, where we replace the exact weight update

$$w_{t+1} = \frac{1}{1 + R_t^{\mathrm{p}}} (\mathbf{1} + r_t) \circ (w_t + z_t)$$

with the simplified version $w_{t+1} = w_t + z_t$, by assuming that $r_t = 0$. At first glance it appears to be a gross simplification, but this assumption is only made for the purpose of propagating the portfolio forward in our planning process; we do take the returns into account in the first term of our objective. We are thus neglecting *second-order* terms, and we cannot be too far off if the per period returns are small compared to one.

In a similar way, adding the constraints $\mathbf{1}^T z_\tau = 0$ for $\tau = t+1, \ldots, t+H-1$ suggests that we are ignoring the transaction and holding costs, since if z_τ were a realized trade we would have $\mathbf{1}^T z_\tau = -\phi_\tau^{\mathrm{trade}}(z_\tau) - \phi_\tau^{\mathrm{hold}}(w_\tau + z_\tau)$. As above, this assumption is only made for the purpose of propagating our portfolio forward in our planning exercise; we do take the costs into account in the objective.

Terminal constraints. In MPO, with a reasonably long horizon, we can add a terminal (equality) constraint, which requires the final planned weight to take some specific value, $w_{t+H} = w^{\mathrm{term}}$. A reasonable choice for the terminal portfolio weight is (our estimate of) the benchmark weight w^{b} at period $t + H$.

For optimization of absolute or excess return, the terminal weight would be cash, *i.e.*, $w^{\mathrm{term}} = e_{n+1}$. This means that our planning exercise should finish with the portfolio all cash. This does not mean we intend to liquidate the portfolio in H periods; rather, it means we should carry out our planning as if this were the case. This will keep us from making the mistake of moving into what appears, in terms of our returns predictions, to be an attractive position that it is, however, expensive to unwind. For optimization relative to a benchmark, the natural terminal constraint is to be in the (predicted) benchmark.

Note that adding a terminal constraint reduces the number of variables. We solve the problem (2.21), but with w_{t+H} a given constant, not a variable. The initial weight w_t is also a given constant; the intermediate weights $w_{t+1}, \ldots, w_{t+H-1}$ are variables.

2.5.3 Computation

The MPO problem (2.22) has Hn variables. In general the complexity of a convex optimization increases as the cube of the number of variables, but in this case the special structure of the problem can be exploited so that the computational effort grows linearly in H, the horizon. Thus, solving the MPO problem (2.22) should be a factor H slower than solving the SPO problem (2.16). For modest H (say, a few tens), this is not a problem. But for $H = 100$ (say) solving the MPO problem can be very challenging. Distributed methods based on ADMM [20, 19] can be used to solve the MPO problem using multiple processors. In most cases we can solve the MPO problem in production. The issue is back-testing, since we must solve the problem many times, and with many variations of the parameters.

2.5.4 How MPO is used

All of the general ideas about how SPO is used apply to MPO as well; for example, we consider the parameters in the MPO problem as knobs that we adjust to achieve good performance under back-test and stress-test. In MPO, we must provide forecasts of each quantity for each period over the next H periods. This can be done using sophisticated forecasts, with possibly different forecasts for each period, or in a very simple way, with predictions that are constant.

2.5.5 Multi-scale optimization

MPO trading requires estimates of all relevant quantities, like returns, transaction costs, and risks, over H trading periods into the future. In this subsection we describe a simplification of MPO that requires fewer predictions, as well as less computation to carry out the optimization required in each period. We still create a plan for trades and weights over the next H periods, but we assume that trades take place only a few times over the horizon; in other time periods the planned portfolio is maintained with no trading. This preserves the idea that we have recourse; but it greatly simplifies the problem (2.21). We describe the idea for three trades, taken in the short term, medium term, and long term, and an additional trade at the end to satisfy a terminal constraint $w_{t+H} = w^{\mathrm{b}}$.

Specifically we add the constraint that in (2.21), trading (*i.e.*, $z_\tau \neq 0$) only occurs at specific periods in the future, for

$$\tau = t, \quad \tau = t + T^{\mathrm{med}}, \quad \tau = t + T^{\mathrm{long}}, \quad \tau = t + H - 1,$$

where

$$1 < T^{\mathrm{med}} < T^{\mathrm{long}} < H - 1.$$

We interpret $z^{\mathrm{short}} = z_t$ as our short term trade, $z^{\mathrm{med}} = z_{t+T^{\mathrm{med}}}$ as our medium term trade, and $z^{\text{long}} = z_{t+T^{\text{long}}}$ as our long term trade, in our trading plan. The final nonzero trade z_{t+H-1} is determined by the terminal constraint.

For example we might take $T^{\text{med}} = 5$ and $T^{\text{long}} = 21$, with $H = 100$. If the periods represent days, we plan to trade now (short term), in a week (medium term) and in month (longer term); in 99 days, we trade to the benchmark. The only variables we have are the short, medium, and long term trades, and the associated weights, given by

$$w^{\text{short}} = w_t + z^{\text{short}}, \quad w^{\text{med}} = w^{\text{short}} + z^{\text{med}}, \quad w^{\text{long}} = w^{\text{med}} + z^{\text{long}}.$$

To determine the trades to make, we solve (2.21) with all other z_τ set to zero, and using the weights given above. This results in an optimization problem with the same form as (2.21), but with only three variables each for trading and weights, and three terms in the objective, plus an additional term that represents the transaction cost associated with the final trade to the benchmark at time $t + H - 1$.

2.6 Implementation

We have developed an open-source Python package `CVXPortfolio` [25] that implements the portfolio simulation and optimization concepts discussed in the chapter. The package relies on Pandas [87] for managing data. Pandas implements structured data types as in-memory databases (similar to R dataframes) and provides a rich API for accessing and manipulating them. Through Pandas, it is easy to couple our package with database backends. The package uses the convex optimization modeling framework CVXPY [41] to construct and solve portfolio optimization problems.

The package provides an object-oriented framework with classes representing return, risk measures, transaction costs, holding constraints, trading constraints, etc. Single-period and multi-period optimization models are constructed from instances of these classes. Each instance generates CVXPY expressions and constraints for any given period t, making it easy to combine the instances into a single convex model. In § 2.7 we give some simple numerical examples that use `CVXPortfolio`.

2.6.1 Components

We briefly review the major classes in the software package. Implementing additional classes, such as novel policies or risk measures, is straightforward.

Return estimates. Instances of the `ReturnsForecast` class generate a return estimate $\hat{r}_t$ for period t using only information available at that period. The simplest `ReturnsForecast` instance wraps a Pandas dataframe with return estimates for each period:

```
r_hat = ReturnsForecast(return_estimates_dataframe)
```

Multiple `ReturnsForecast` instances can be blended into a linear combination.

Risk measures. Instances of a risk measure class, contained in the `risks` submodule, generate a convex cost representing a risk measure at a given period t. For example, the `FullSigma` class generates the cost $(w_t + z_t)^T \Sigma_t (w_t + z_t)$ where $\Sigma_t \in \mathbf{R}^{(n+1)\times(n+1)}$ is an explicit matrix, whereas the `FactorModel` class generates the cost with a factor model of Σ_t. Any risk measure can be switched to absolute or active risk and weighted with a risk aversion parameter. The package provides all the risk measures discussed in §2.4.2.

Costs. Instances of the `TcostModel` and `HcostModel` classes generate transaction and holding cost estimates, respectively. The same classes work both for modeling costs in a portfolio optimization problem and calculating realized costs in a trading simulation. Cost objects can also be used to express other objective terms like soft constraints.

Constraints. The package provides classes representing each of the constraints discussed in §2.4.4 and §2.4.5. For example, instances of the `LeverageLimit` class generate a leverage limit constraint that can vary by period. Constraint objects can be converted into soft constraints, which are cost objects.

Policies. Instances of a policy class take holdings w_t and value v_t and output trades z_t using information available in period t. Single-period optimization policies are constructed using the `SinglePeriodOpt` class. The constructor takes a `ReturnsForecast`, a list of costs, including risk models (multiplied by their coefficients), and constraints. For example, the following code snippet constructs a SPO policy:

```
spo_policy = SinglePeriodOpt(r_hat,
                             [gamma_risk*factor_risk,
                             gamma_trade*tcost_model,
                             gamma_hold*hcost_model],
                             [leverage_limit])
```

Multi-period optimization policies are constructed similarly. The package also provides classes for simple policies such as periodic re-balancing.

Simulator. The `MarketSimulator` class is used to run trading simulations, or back-tests. Instances are constructed with historical returns and other market data, as well as transaction and holding cost models. Given a `MarketSimulator` instance `market_sim`, a back-test is run by calling the `run_backtest` method with an initial portfolio, policy, and start and end periods:

```
backtest_results = market_sim.run_backtest(init_portfolio,
                                           policy,
                                           start_t, end_t)
```

Multiple back-tests can be run in parallel with different conditions. The back-test results include all the metrics discussed in § 2.3.

2.7 Examples

In this section we present simple numerical examples illustrating the ideas developed above, all carried out using `CVXPortfolio` and open-source market data (and some approximations where no open source data is available). The code for these is available at `http://github.com/cvxgrp/cvxportfolio/tree/master/examples`. Given our approximations, and other short-comings of our simulations that we will mention below, the particular numerical results we show should not be taken too seriously. But the simulations are good enough for us to illustrate real phenomena, such as the critical role transaction costs can play, or how important hyper-parameter search can be.

2.7.1 Data for simulation

We work with a period of 5 years, from January 2012 through December 2016, on the components of the S&P 500 index as of December 2016. We select the ones continuously traded in the period. (By doing this we introduce survivorship bias [45].) We collect open-source market data from Quandl [108]. The data consists of realized daily market returns r_t (computed using closing prices) and volumes V_t. We use the federal reserve overnight rate for the cash return. Following [2] we approximate the daily volatility with a simple estimator, $(\sigma_t)_i = |\log(p_t^{\mathrm{open}})_i - \log(p_t^{\mathrm{close}})_i|$, where $(p_t^{\mathrm{open}})_i$ and $(p_t^{\mathrm{close}})_i$ are the open and close prices for asset i in period t. We could not find open-source data for the bid-ask spread, so we used the value $a_t = 0.05\%$ (5 basis points) for all assets and periods. As holding costs we use $s_t = 0.01\%$ (1 basis point) for all assets and periods. We chose standard values for the other parameters of the transaction and holding cost models: $b_t = 1$, $c_t = 0$, $d_t = 0$ for all assets and periods.

2.7.2 Portfolio simulation

To illustrate back-test portfolio simulation, we consider a portfolio that is meant to track the *uniform portfolio* benchmark, which has weight $w^{\mathrm{b}} = (\mathbf{1}/n, 0)$, *i.e.*, equal fraction of value in all non-cash assets. This is not a particularly interesting or good benchmark portfolio; we use it only as a simple example to illustrate the effects of transaction costs. The portfolio starts with $w_1 = w^{\mathrm{b}}$, and due to asset returns drifts from this weight vector. We periodically re-balance, which means using trade vector

Initial total val.	Rebalancing frequency	Active return	Active risk	Trans. cost	Turnover
\$100M	Daily	-0.07%	0.00%	0.07%	220.53%
	Weekly	-0.07%	0.09%	0.04%	105.67%
	Monthly	-0.12%	0.21%	0.02%	52.71%
	Quarterly	-0.11%	0.35%	0.01%	29.98%
	Annually	-0.10%	0.63%	0.01%	12.54%
	Hold	-0.36%	1.53%	0.00%	0.00%
\$10B	Daily	-0.25%	0.01%	0.25%	220.53%
	Weekly	-0.19%	0.09%	0.16%	105.67%
	Monthly	-0.20%	0.21%	0.10%	52.71%
	Quarterly	-0.17%	0.35%	0.07%	29.99%
	Annually	-0.13%	0.63%	0.04%	12.54%
	Hold	-0.36%	1.53%	0.00%	0.00%

Table 2.1: Portfolio simulation results with different initial value and different re-balancing frequencies. All values are annualized.

$z_t = w^{\mathrm{b}} - w_t$. For other values of t (*i.e.*, the periods in which we do not re-balance) we have $z_t = 0$.

We carry out six back-test simulations for each of two initial portfolio values, \$100M and \$10B. The six simulations vary in re-balancing frequency: Daily, weekly, monthly, quarterly, annually, or never (also called ‘hold’ or ‘buy-and-hold’). For each simulation we give the portfolio active return $\bar{R}^{\mathrm{a}}$ and active risk σ^{a} (defined in §2.3.2), the annualized average transaction cost $\frac{250}{T}\sum_{t=1}^{T}\phi_t^{\mathrm{trade}}(z_t)$, and the annualized average turnover $\frac{250}{T}\sum_{t=1}^{T}\|(z_t)_{1:n}\|_1/2$.

Table 2.1 shows the results. (The active return is also included for completeness.) We observe that transaction cost depends on the total value of the portfolio, as expected, and that the choice of re-balancing frequency trades off transaction cost and active risk. (The active risk is not exactly zero when re-balancing daily because of the variability of the transaction cost, which is included in the portfolio return.) Figure 2.1 shows, separately for the two portfolio sizes, the active risk versus the transaction cost.

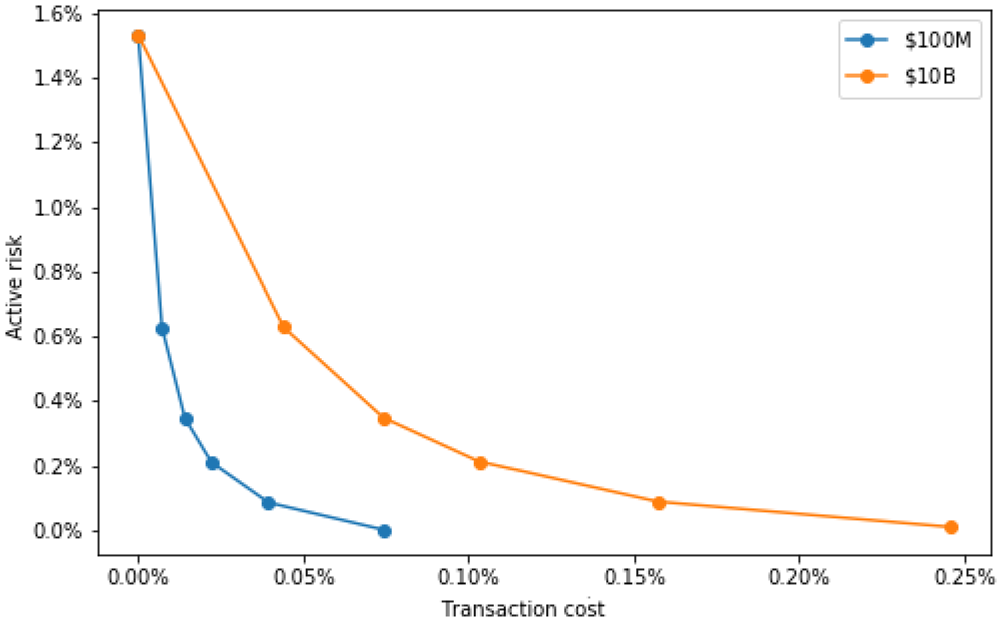

Figure 2.1: Active risk versus transaction cost, for the two initial portfolio sizes. The points on the lines correspond to re-balancing frequencies.

2.7.3 Single-period optimization

In this subsection we show a simple example of the single-period optimization model developed in § 2.4. The portfolio starts with total value $v_1 = \$100\text{M}$ and allocation equal to the uniform portfolio $w_1 = (\mathbf{1}/n, 0)$. We impose a leverage constraint of $L^{\max} = 3$. This simulation uses the market data defined in §2.7.1. The forecasts and risk model used in the SPO are described below.

Risk model. Proprietary risk models, *e.g.*, from MSCI (formerly Barra), are widely used. Here we use a simple factor risk model estimated from past realized returns, using a similar procedure to [2]. We estimate it on the first day of each month, and use it for the rest of the month. Let t be an estimation time period, and $t - M^{\text{risk}}$ the time period two years before. Consider the second moment of the window of realized returns $\Sigma^{\text{exp}} = \frac{1}{M^{\text{risk}}} \sum_{\tau=t-M^{\text{risk}}}^{t-1} r_\tau r_\tau^T$, and its eigenvalue decomposition $\Sigma^{\text{exp}} = \sum_{i=1}^{n} \lambda_i q_i q_i^T$, where the eigenvalues λ_i are in descending order. Our factor risk model is

$$F = [q_1 \cdots q_k], \quad \Sigma^{\text{f}} = \mathbf{diag}(\lambda_1, \ldots, \lambda_k), \quad D = \sum_{i=k+1}^{n} \lambda_i \, \mathbf{diag}(q_i) \, \mathbf{diag}(q_i),$$

with $k = 15$. (The diagonal matrix D is chosen so the factor model $F\Sigma^{\text{f}}F^T + D$ and the empirical second moment Σ^{exp} have the same diagonal elements.)

Return forecasts. The risk-free interest rates are known exactly, $(\hat{r}_t)_{n+1} = (r_t)_{n+1}$ for all t. Return forecasts for the non-cash assets are always proprietary. They are generated using many methods, ranging from analyst predictions to sophisticated machine learning techniques, based on a variety of data feeds and sources. For these examples we generate simulated return forecasts by adding zero-mean noise to the realized returns and then rescaling, to obtain return estimates that would (approximately) minimize mean squared error. Of course this is not a real return forecast, since it uses the actual realized return; but our purpose here is only to illustrate the ideas and methods.

For all t the return estimates for non-cash assets are

$$(\hat{r}_t)_{1:n} = \alpha\left((r_t)_{1:n} + \epsilon_t\right), \tag{2.23}$$

where $\epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I)$ are independent. We use noise variance $\sigma_\epsilon^2 = 0.02$, so the noise components have standard deviation around 14%, around a factor of 10 larger than the standard deviation of the realized returns. The scale factor α is chosen to minimize the mean squared error $\mathbf{E}((\hat{r}_t)_{1:n} - (r_t)_{1:n})^2$, if we think of r_t as a random variable with variance σ_r, *i.e.*, $\alpha = \sigma_r^2/(\sigma_r^2 + \sigma_\epsilon^2)$. We use the typical value $\sigma_r^2 = 0.0005$, *i.e.*, a realized return standard deviation of around 2%, so $\alpha = 0.024$. Our typical return forecast is on the order of ±0.3%. This corresponds to an information ratio $\sqrt{\alpha} \approx 0.15$, which is on the high end of what might be expected in practice [61].

With this level of noise and scaling, our return forecasts have an accuracy on the order of what we might expect from a proprietary forecast. For example, across all the assets and all days, the sign of predicted return agrees with the sign of the real return around 54% of the times.

Volume and volatility forecasts. We use simple estimates of total market volumes and daily volatilities (used in the transaction cost model), as moving averages of the realized values with a window of length 10. For example, the volume forecast at time period t and asset i is $(\hat{V_t})_i = \frac{1}{10}\sum_{\tau=1}^{10}(V_{t-\tau})_i$.

SPO back-tests. We carry out multiple back-test simulations over the whole period, varying the risk aversion parameter γ^{risk}, the trading aversion parameter γ^{trade}, and the holding cost multiplier γ^{hold} (all defined and discussed in §2.4.8). We first perform a coarse grid search in the hyper-parameter space, testing all combinations of

$$\begin{aligned}
\gamma^{\text{risk}} &= 0.1,\ 0.3,\ 1,\ 3,\ 10,\ 30,\ 100,\ 300,\ 1000,\\
\gamma^{\text{trade}} &= 1,\ 2,\ 5,\ 10,\ 20,\\
\gamma^{\text{hold}} &= 1,
\end{aligned}$$

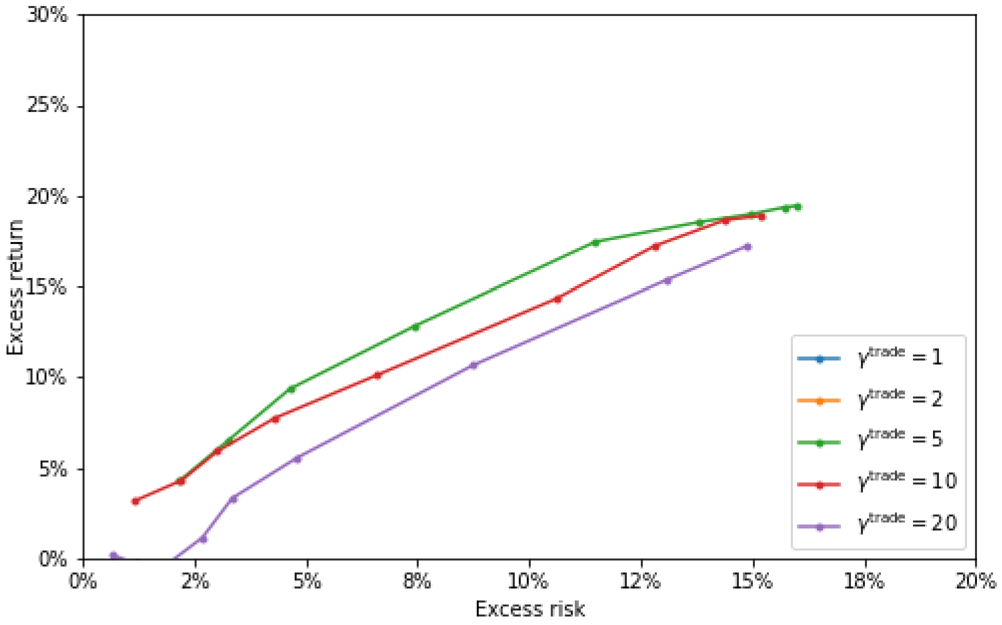

Figure 2.2: SPO example, coarse hyper-parameter grid search. (Some curves do not fit in the plot.)

a total of 45 back-test simulations. (Logarithmic spacing is common in hyper-parameter searches.)

Figure 2.2 shows mean excess portfolio return $\overline{R^{\text{e}}}$ versus excess volatility σ^{e} (defined in §2.3.2), for these combinations of parameters. For each value of γ^{trade}, we connect with a line the points corresponding to the different values of γ^{risk}, obtaining a risk-return trade-off curve for that choice of γ^{trade} and γ^{hold}. These show the expected trade-off between mean return and risk. We see that the choice of trading aversion parameter is critical: for some values of γ^{trade} the results are so poor that the resulting curve does not even fit in the plotting area. Values of γ^{trade} around 5 seem to give the best results.

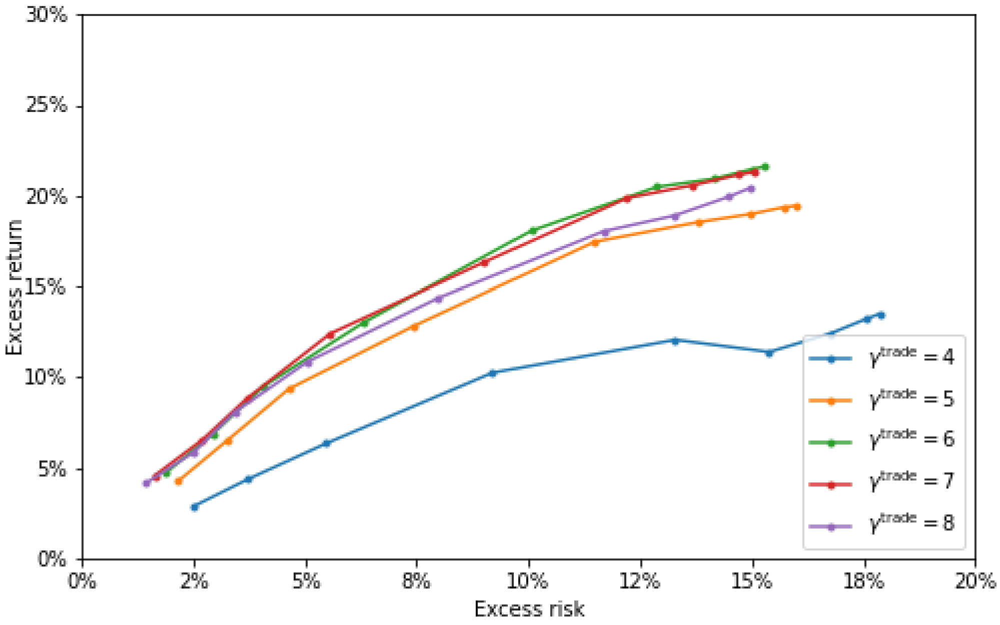

Figure 2.3: SPO example, fine hyper-parameter grid search.

We then perform a fine hyper-parameter search, focusing on trade aversion parameters values around $\gamma^{\text{trade}} = 5$,

$$\gamma^{\text{trade}} = 4,\ 5,\ 6,\ 7,\ 8,$$

and the same values of γ^{risk} and γ^{hold}. Figure 2.3 shows the resulting curves of excess return versus excess risk. A value around $\gamma^{\text{trade}} = 6$ seems to be best.

For our last set of simulations we use a finer range of risk aversion parameters, focus on an even narrower range of the trading aversion parameter, and also vary the

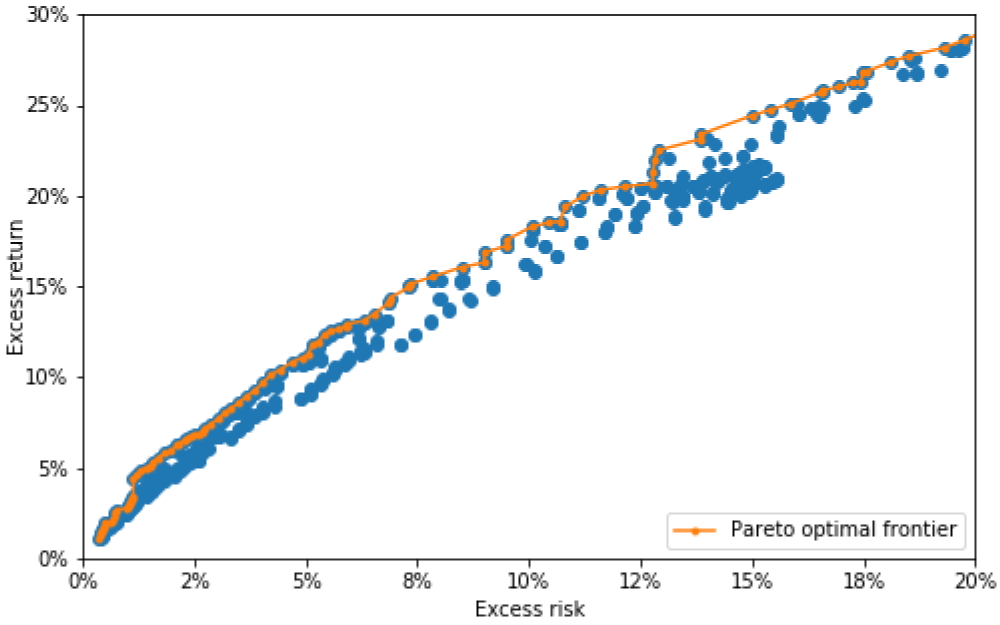

Figure 2.4: SPO example, grid search over 410 hyper-parameter combinations. The line connects the Pareto optimal points.

hold aversion parameter. We test all combinations of

$$\begin{aligned}\gamma^{\text{risk}} &= 0.1,\ 0.178,\ 0.316,\ 0.562,\ 1,\ 2,\ 3,\ 6,\ 10,\ 18,\ 32,\ 56,\\ &\qquad 100,\ 178,\ 316,\ 562,\ 1000,\\ \gamma^{\text{trade}} &= 5.5,\ 6,\ 6.5,\ 7,\ 7.5,\ 8,\\ \gamma^{\text{hold}} &= 0.1,\ 1,\ 10,\ 100,\ 1000,\end{aligned}$$

a total of 410 back-test simulations. The results are plotted in figure 2.4 as points in the risk-return plane. The Pareto optimal points, *i.e.*, those with the lowest risk for a given level of return, are connected by a line. Table 2.2 lists a selection of the Pareto optimal points, giving the associated hyper-parameter values.

From this curve and table we can make some interesting observations. The first

γ^{risk}	γ^{trade}	γ^{hold}	Excess return	Excess risk
1000.00	8.0	100	1.33%	0.39%
562.00	6.0	100	2.49%	0.74%
316.00	7.0	100	2.98%	1.02%
1000.00	7.5	10	4.64%	1.22%
562.00	8.0	10	5.31%	1.56%
316.00	7.5	10	6.53%	2.27%
316.00	6.5	10	6.88%	2.61%
178.00	6.5	10	8.04%	3.20%
100.00	8.0	10	8.26%	3.32%
32.00	7.0	10	12.35%	5.43%
18.00	6.5	0.1	14.96%	7.32%
6.00	7.5	10	18.51%	10.44%
2.00	6.5	10	23.40%	13.87%
0.32	6.5	10	26.79%	17.50%
0.18	7.0	10	28.16%	19.30%

Table 2.2: SPO example, selection of Pareto optimal points (ordered by increasing risk and return).

is that we do substantially better with large values of the holding cost multiplier parameter compared to $\gamma^{\text{hold}} = 1$, even though the actual holding cost (used by the simulator to update the portfolio each day) is very small, one basis point. This is a good example of regularization in SPO; our large holding cost multiplier parameter tells the SPO algorithm to avoid short positions, and the result is that the overall portfolio performance is better.

It is hardly surprising that the risk aversion parameter varies over this selection of Pareto optimal points; after all, this is the parameter most directly related to the risk-return trade-off. One surprise is that the value of the hold aversion hyper-parameter varies considerably as well.

In practice, we would back-test many more combinations of these three hyper-parameters. Indeed we would also carry out back-tests varying combinations of other parameters in the SPO algorithm, for example the leverage, or the individual terms in transaction cost functions. In addition, we would carry out stress-tests and other what-if simulations, to get an idea of how our SPO algorithm might perform in other, or more stressful, market conditions. (This would be especially appropriate given our choice of back-test date range, which was entirely a bull market.) Since these back-tests can be carried out in parallel, there is no reason to not carry out a large number of them.

2.7.4 Multi-period optimization

In this subsection we show the simplest possible example of the multi-period optimization model developed in § 2.5, using planning horizon $H = 2$. This means that in each time period the MPO algorithm plans both current day and next day trades, and then executes only the current day trades. As a practical matter, we would not expect a great performance improvement over SPO using a planning horizon of $H = 2$ days compared to SPO, which uses $H = 1$ day. Our point here is to demonstrate that it is different.

The simulations are carried out using the market data described in §2.7.1. The portfolio starts with total value $v_1 = \$100\text{M}$ and uniform allocation $w_1 = (\mathbf{1}/n, 0)$.

We impose a leverage constraint of $L^{\max} = 3$. The risk model is the same one used in the SPO example. The volume and volatility estimates (for both the current and next period) are also the same as those used in the SPO example.

Return forecasts. We use the same return forecast we generated for the previous example, but at every time period we provide both the forecast for the current time period and the one for the next:

$$\hat{r}_{t|t} = \hat{r}_t, \quad \hat{r}_{t+1|t} = \hat{r}_{t+1},$$

where $\hat{r}_t$ and $\hat{r}_{t+1}$ are the same ones used in the SPO example, given in (2.23). The MPO trading algorithm thus sees each return forecast twice, $\hat{r}_{t+1} = \hat{r}_{t+1|t} = \hat{r}_{t+1|t+1}$, *i.e.*, today's forecast of tomorrow's return is the same as tomorrow's forecast of tomorrow's return.

As in the SPO case, this is clearly not a practical forecast, since it uses the realized return. In addition, in a real setting the return forecast would be updated at every time period, so that $\hat{r}_{t+1|t} \neq \hat{r}_{t+1|t+1}$. Our goal in choosing these simulated return forecasts is to have ones that are similar to the ones used in the SPO example, in order to compare the results of the two optimization procedures.

Back-tests. We carry out multiple back-test simulations varying the parameters γ^{risk}, γ^{trade}, and γ^{hold}. We first perform a coarse grid search in the hyper-parameter space, with the same parameters as in the SPO example. We test all combinations of

$$\begin{aligned} \gamma^{\text{risk}} &= 0.1,\ 0.3,\ 1,\ 3,\ 10,\ 30,\ 100,\ 300,\ 1000, \\ \gamma^{\text{trade}} &= 1,\ 2,\ 5,\ 10,\ 20, \\ \gamma^{\text{hold}} &= 1, \end{aligned}$$

a total of 45 back-test simulations.

The results are shown in figure 2.5, where we plot mean excess portfolio return $\overline{R^{\text{e}}}$ versus excess risk σ^{e}. For some trading aversion parameter values the results were so bad that they did not fit in the plotting area.

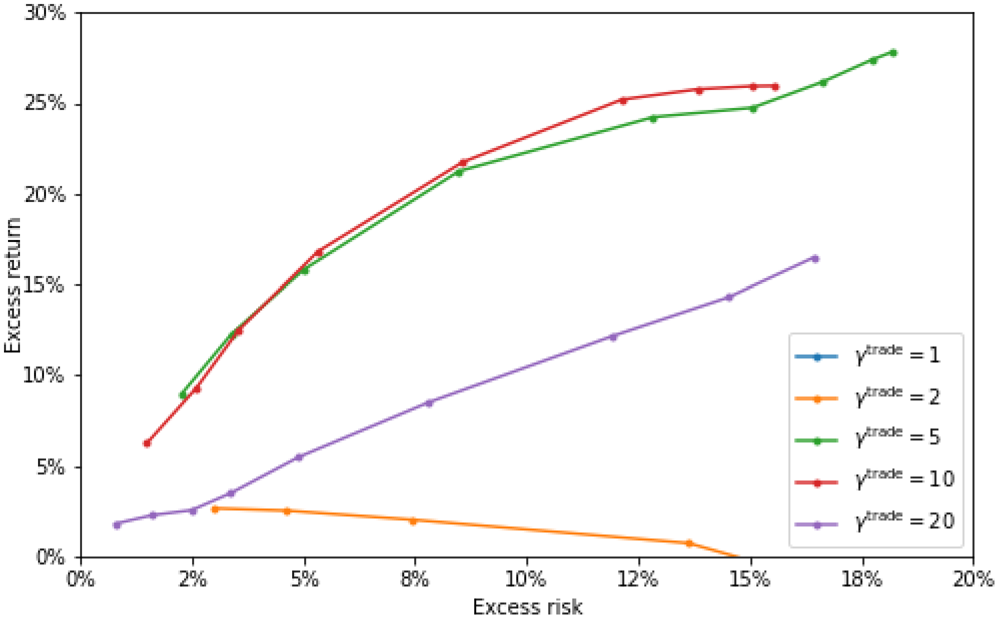

Figure 2.5: MPO example, coarse hyper-parameter grid search.

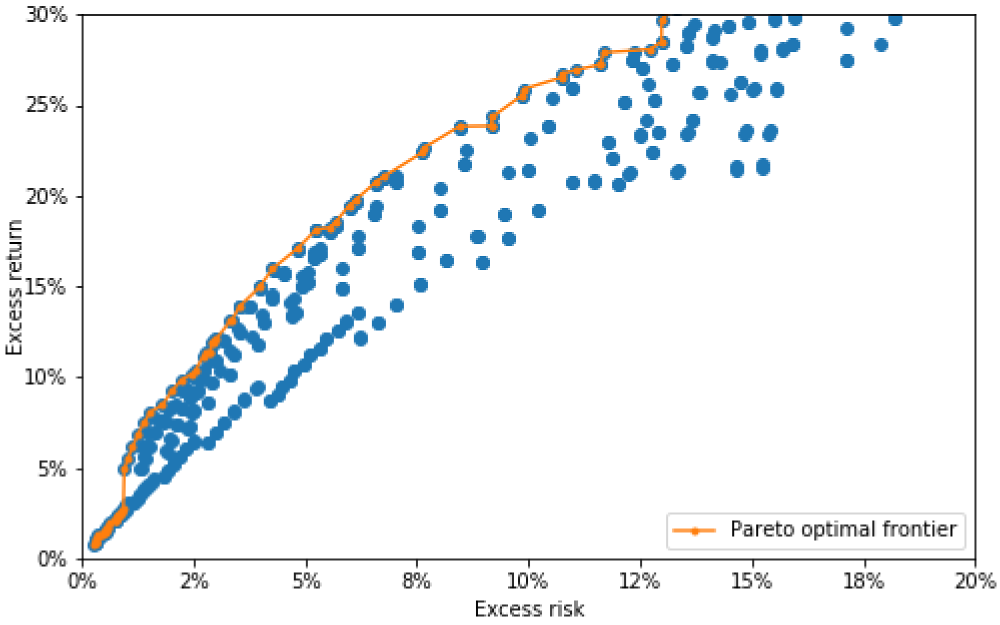

Figure 2.6: MPO example, grid search over 390 hyper-parameter combinations. The line connects the Pareto optimal points.

We then perform a more accurate hyper-parameter search using a finer range for γ^{risk}, focusing on the values around $\gamma^{\text{trade}} = 10$, and also varying the hold aversion parameter. We test all combinations of

$$\begin{aligned} \gamma^{\text{risk}} &= 1,\ 2,\ 3,\ 6,\ 10,\ 18,\ 32,\ 56,\ 100,\ 178,\ 316,\ 562,\ 1000, \\ \gamma^{\text{trade}} &= 7,\ 8,\ 9,\ 10,\ 11,\ 12, \\ \gamma^{\text{hold}} &= 0.1,\ 1,\ 10,\ 100,\ 1000, \end{aligned}$$

for a total of 390 back-test simulations. The results are plotted in figure 2.6 as points in the risk-return plane. The Pareto optimal points are connected by a line.

Finally we compare the results obtained with the SPO and MPO examples. Figure 2.7 shows the Pareto optimal frontiers for both cases. We see that the MPO method

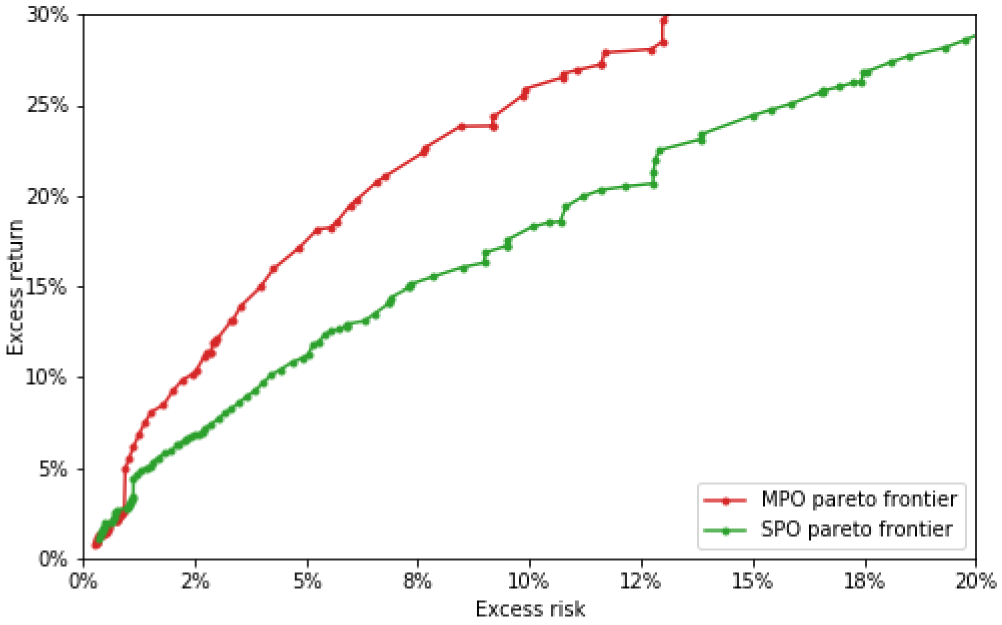

Figure 2.7: Pareto optimal frontiers for SPO and MPO.

has a substantial advantage over the SPO method, mostly explained by the advantage of a forecast for tomorrow's, as well as today's, return.

2.7.5 Simulation time

Here we give some rough idea of the computation time required to carry out the simulation examples shown above, focusing on the SPO case. The back-test simulation is single-threaded, so multiple back-tests can be carried out on separate threads.

Figure 2.8 gives the breakdown of execution time for a back-test, showing the time taken for each step of simulation, broken down into the simulator, the numerical solver, and the rest of the policy (data management and CVXPY manipulations). We can see that simulating one day takes around 0.25 seconds, so a back test over 5 years takes around 5 minutes. The bulk of this (around 0.15 seconds) is the optimization carried out each day. The simulator time is, as expected, negligible.

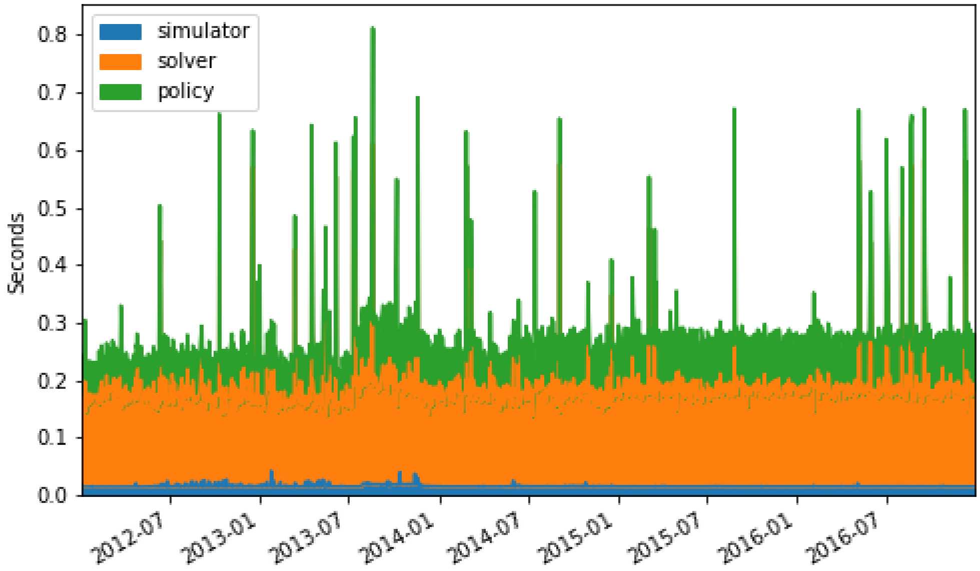

Figure 2.8: Execution time for each day for one SPO back-test.

We carried out the multiple back-tests using a 32 core machine that can execute 64 threads simultaneously. Carrying out 410 back-tests, which entails solving around a half million convex optimization problems, thus takes around thirty minutes. (In fact, it takes a bit longer, due to system overhead.)

We close by making a few comments about these optimization times. First, they can be considerably reduced by avoiding the 3/2-power transaction cost terms, which slow the optimizer. By replacing these terms with square transaction cost terms, we can obtain a speedup of more than a factor of two. Replacing the default generic solver ECOS [43] used in CVXPY with a custom solver, such as one based on operator-splitting methods [20], would result in an even more substantial speedup.

Chapter 3

Risk-Constrained Kelly Gambling

In this chapter, we consider the classic Kelly gambling problem with general distribution of outcomes, and an additional risk constraint that limits the probability of a drawdown of wealth to a given undesirable level. We develop a bound on the drawdown probability; using this bound instead of the original risk constraint yields a convex optimization problem that guarantees the drawdown risk constraint holds. Numerical experiments show that our bound on drawdown probability is reasonably close to the actual drawdown risk, as computed by Monte Carlo simulation. Our method is parametrized by a single parameter that has a natural interpretation as a risk-aversion parameter, allowing us to systematically trade off asymptotic growth rate and drawdown risk. Simulations show that this method yields bets that out perform fractional-Kelly bets for the same drawdown risk level or growth rate. Finally, we show that a natural quadratic approximation of our convex problem is closely connected to the classical mean-variance Markowitz portfolio selection problem.

3.1 Introduction

In 1956 John Kelly proposed a systematic way to allocate a total wealth across a number of bets so as to maximize the long term growth rate when the gamble is repeated [69, 81]. Similar results were later derived in the finance literature, under the name of *growth-optimum portfolio*; see, *e.g.*, [90, Ch. 6]). It is well known that with Kelly optimal bets there is a risk of the wealth dropping substantially from its original value before increasing, *i.e.*, a *drawdown*. Several ad hoc methods can be used to limit this drawdown risk, at the cost of decreased growth rate. The best known method is fractional Kelly betting, in which only a fraction of the Kelly optimal bets are made [37]. The same method has been proposed in the finance literature [22]. Another ad hoc method is Markowitz's mean-variance portfolio optimization [83], which trades off two objectives that are related to, but not the same as, long term growth rate and drawdown risk.

In this chapter we directly address drawdown risk and show how to find bets that trade off drawdown risk and growth rates. We introduce the risk-constrained Kelly gambling problem, in which the long term wealth growth rate is maximized with an additional constraint that limits the probability of a drawdown to a specified level. This idealized problem captures what we want, but seems very difficult to solve. We then introduce a convex optimization problem that is a *restriction* of this problem; that is, its feasible set is smaller than that of the risk-constrained problem. This problem is tractable, using modern convex optimization methods.

Our method can be used in two ways. First, it can be used to find a conservative set of bets that are guaranteed to satisfy a given drawdown risk constraint. Alternatively, its single parameter can be interpreted as a risk-aversion parameter that controls the trade-off between growth rate and drawdown risk, analogous to Markowitz mean-variance portfolio optimization [83], which trades off mean return and (variance) risk. Indeed, we show that a natural quadratic approximation of our convex problem can be closely connected to Markowitz mean-variance portfolio optimization.

In § 3.2 we review the Kelly gambling problem, and describe methods for computing the optimal bets using convex optimization. In simple cases, such as when there are two possible outcomes, the Kelly optimal bets are well known. In others, for example when the returns come from infinite distributions, the methods do not seem to be well known. In § 3.3, we define the drawdown risk, and in § 3.4, we derive a bound on the drawdown risk. In § 3.5, we use this bound to form the risk-constrained Kelly gambling problem, which is a tractable convex optimization problem. In § 3.6, we derive a quadratic approximation of the risk-constrained Kelly gambling problem, and relate it to classical Markowitz portfolio optimization. Finally, in § 3.7 we give some numerical examples to illustrate the methods.

3.2 Kelly gambling

In Kelly gambling, we place a fixed fraction of our total wealth (assumed positive) on n bets. We denote the fractions as $b \in \mathbf{R}^n$, so $b \geq 0$ and $\mathbf{1}^T b = 1$, where $\mathbf{1}$ is the vector with all components 1. The n bets have a random nonnegative payoff or return, denoted $r \in \mathbf{R}^n_+$, so the wealth after the bet changes by the (random) factor $r^T b$. We will assume that all bets do not have infinite return in expectation, *i.e.*, that $\mathbf{E}\, r_i < \infty$ for $i = 1, \ldots, n$. We will also assume that the bet n has a certain return of one, *i.e.*, $r_n = 1$ almost surely. This means that b_n represents the fraction of our wealth that we do not wager, or hold as cash. The bet vector $b = e_n$ corresponds to not betting at all. We refer to the bets $1, \ldots, n-1$ as the risky bets.

We mention some special cases of this general Kelly gambling setup.

- *Two outcomes.* We have $n = 2$, and r takes on only two values: $(P, 1)$, with probability π, and $(0, 1)$, with probability $1 - \pi$. The first outcome corresponds to winning the bet (π is the probability of winning) and $P > 1$ is the payoff.

- *Mutually exclusive outcomes.* There are $n - 1$ mutually exclusive outcomes, with return vectors: $r = (P_k e_k, 1)$ with probability $\pi_k > 0$, for $k = 1, \ldots, n-1$, where $P_k > 1$ is the payoff for outcome k, and e_k is the unit vector with kth entry one and all others 0. Here we bet on which of outcomes $1, \ldots, n-1$ will be the winner (*e.g.*, the winner of a horse race).

- *General finite outcomes.* The return vector takes on K values $r_1, \ldots, r_K$, with probabilities $\pi_1, \ldots, \pi_K$. This case allows for more complex bets, for example in horse racing show, place, exacta, perfecta, and so on.

- *General returns.* The return r comes from an arbitrary infinite distribution (with $r_n = 1$ almost surely). If the returns are log-normal, the gamble is a simple model of investing (long only) in $n - 1$ assets with log-normal returns;

the nth asset is risk free (cash). More generally, we can have $n-1$ arbitrary derivatives (*e.g.*, options) with payoffs that depend on an underlying random variable.

3.2.1 Wealth growth

The gamble is repeated at times (epochs) $t = 1, 2, \ldots$, with IID (independent and identically distributed) returns. Starting with initial wealth $w_1 = 1$, the wealth at time t is given by

$$w_t = (r_1^T b) \cdots (r_{t-1}^T b),$$

where r_t denotes the realized return at time t (and not the tth entry of the vector). The wealth sequence $\{w_t\}$ is a stochastic process that depends on the choice of the bet vector b, as well as the distribution of return vector r. Our goal is to choose b so that, roughly speaking, the wealth becomes large.

Note that $w_t \geq 0$, since $r \geq 0$ and $b \geq 0$. The event $w_t = 0$ is called *ruin*, and can only happen if $r^T b = 0$ has positive probability. The methods for choosing b that we discuss below all preclude ruin, so we will assume it does not occur, *i.e.*, $w_t > 0$ for all t, almost surely. Note that if $b_n > 0$, ruin cannot occur since $r^T b \geq b_n$ almost surely.

With $v_t = \log w_t$ denoting the logarithm of the wealth, we have

$$v_t = \log(r_1^T b) + \cdots + \log(r_{t-1}^T b).$$

Thus $\{v_t\}$ is a random walk, with increment distribution given by the distribution of $\log(r^T b)$. The drift of the random walk is $\mathbf{E} \log(r^T b)$; we have $\mathbf{E}\, v_t = (t-1)\, \mathbf{E} \log(r^T b)$, and $\mathbf{var}\, v_t = (t-1)\, \mathbf{var} \log(r^T b)$. The quantity $\mathbf{E} \log(r^T b)$ can be interpreted as the average growth rate of the wealth; it is the drift in the random walk $\{v_t\}$. The (expected) growth rate $\mathbf{E} \log(r^T b)$ is a function of the bet vector b.

3.2.2 Kelly gambling

In Kelly gambling, we choose b to maximize $\mathbf{E}\log(r^Tb)$, the growth rate of wealth. This leads to the optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\log(r^Tb) \\ \text{subject to} & \mathbf{1}^Tb = 1, \quad b \geq 0, \end{array} \tag{3.1}$$

with variable b. We call a solution $b^\star$ of this problem a set of *Kelly optimal* bets. The Kelly gambling problem is always feasible, since $b = e_n$ (which corresponds to not placing any bets) is feasible. This choice achieves objective value zero, so the optimal value of the Kely gambling problem is always nonnegative. The Kelly gambling problem (3.1) is a convex optimization problem, since the objective is concave and the constraints are convex.

Kelly optimal bets maximize growth rate of the wealth. If $\tilde{b}$ is a bet vector that is not Kelly optimal, with associated wealth sequence $\tilde{w}_t$, and $b^\star$ is Kelly optimal, with associated wealth sequence $w_t^\star$, then $w_t^\star/\tilde{w}_t \to \infty$ with probability one as $t \to \infty$. (This follows immediately since the random walk $\log w_t^\star - \log \tilde{w}_t$ has positive drift [49, §XII.2]. Also see [36, §16] for a general discussion of Kelly gambling.)

We note that the bet vector $b = e_n$ is Kelly optimal if and only if $\mathbf{E}\, r_i \leq 1$ for $i = 1, \ldots, n-1$. Thus we should not bet at all if all the bets are losers in expectation; conversely, if just one bet is a winner in expectation, the optimal bet is not the trivial one e_n, and the optimal growth rate is positive. We show this in the appendix.

3.2.3 Computing Kelly optimal bets

Here we describe methods to compute Kelly optimal bets, *i.e.*, to solve the Kelly optimization problem (3.1). It can be solved analytically or semi-analytically for simple cases; the general finite outcomes case can be handled by standard convex optimization tools, and the general case can be handled via stochastic optimization.

Two outcomes. For a simple bet with two outcomes, with win probability π and payoff P, we obtain the optimal bet with simple minimization of a univariate function.

We have

$$b^\star = \left(\frac{\pi P - 1}{P-1}, \frac{P - \pi P}{P-1}\right),$$

provided $\pi P > 1$; if $\pi P \leq 1$, the optimal bet is $b^\star = (0, 1)$. Thus we should bet a fraction $(\pi P - 1)/(P - 1)$ of our wealth each time, if this quantity is positive.

General finite outcomes. When the return distribution is finite the Kelly gambling problem reduces to

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^K \pi_i \log(r_i^T b) \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \end{array} \tag{3.2}$$

which is readily solved using convex optimization methods [21]. Convex optimization software systems like CVX [58], CVXPY [41] and Convex.jl [118], based on DCP (Disciplined Convex Programming) [59], or others like YALMIP [80], can handle such problems directly. In our numerical simulation we use CVXPY with the open source solver ECOS [43], recently extended to handle exponential cone constraints [112].

General returns. We can solve the Kelly gambling problem (3.1) even in the most general case, when r takes on an infinite number of values, provided we can generate samples from the distribution of r. In this case we can use a projected stochastic gradient method with averaging [109, 99, 106, 74, 23].

As a technical assumption we assume here that the Kelly optimal bet $b^\star$ satisfies $(b^\star)_n > 0$, *i.e.*, the optimal bet involves holding some cash. We assume we know $\varepsilon > 0$ that satisfies $\varepsilon < (b^\star)_n$ (which implies that $r^T b > \varepsilon$ a.s.) and define $\Delta_\varepsilon = \{b \mid \mathbf{1}^T b = 1,\ b \geq 0,\ b_n \geq \varepsilon\}$. Then the gradient of the objective is given by

$$\nabla_b \mathbf{E} \log(r^T b) = \mathbf{E} \nabla_b \log(r^T b) = \mathbf{E} \frac{1}{r^T b} r$$

for any $b \in \Delta_\varepsilon$. So if $r^{(k)}$ is an IID sample from the distribution,

$$\frac{1}{{r^{(k)}}^T b} r^{(k)} \tag{3.3}$$

is an unbiased estimate of the gradient, *i.e.*, a stochastic gradient, of the objective at b. (Another unbiased estimate of the gradient can be obtained by averaging the expression (3.3) over multiple return samples.)

The (projected) stochastic gradient method with averaging computes the iterates

$$\bar{b}^{(k+1)} = \Pi \left(\bar{b}^{(k)} + \frac{t_k}{({r^{(k)}}^T \bar{b}^{(k)})} r^{(k)} \right), \quad k = 1, 2, \ldots,$$

where the starting point $\bar{b}^{(1)}$ is any vector in Δ_ε, $r^{(k)}$ are IID samples from the distribution of r, and Π is (Euclidean) projection onto Δ_ε (which is readily computed; see Lemma 3). The step sizes $t_k > 0$ must satisfy

$$t_k \to 0, \qquad \sum_{k=1}^{\infty} t_k = \infty.$$

(For example, $t_k = C/\sqrt{k}$ with any $C > 0$ satisfies this condition.) Then the (weighted) running average

$$b^{(k)} = \frac{\sum_{i=1}^{k} t_i \bar{b}^{(i)}}{\sum_{i=1}^{k} t_i}$$

converges to Kelly optimal. The stochastic gradient method can be slow to converge, but it always works; that is $\mathbf{E} \log(r^T b^{(k)})$ converges to the optimal growth rate.

In practice, one does not know a priori how small ε should be. One way around this is to choose a small ε, and then check that $(b^{(k)})_n > \varepsilon$ holds for large k, in which case we know our guess of ε was valid. A more important practical variation on the algorithm is *batching*, where we replace the unbiased estimate of the gradient with the average over some number of samples. This does not affect the theoretical convergence of the algorithm, but can improve convergence in practice.

3.3 Drawdown

We define the *minimum wealth* as the infimum of the wealth trajectory over time,

$$W^{\min} = \inf_{t=1,2,\ldots} w_t.$$

This is a random variable, with distribution that depends on b. With $b = e_n$, we have $w_t = 1$ for all t, so $W^{\min} = 1$. With b for which $\mathbf{E}\log(r^T b) > 0$ (which we assume), $W^{\min}$ takes values in $(0, 1]$. Small $W^{\min}$ corresponds to a case where the initial wealth drops to a small value before eventually increasing.

The *drawdown* is defined as $1 - W^{\min}$. A drawdown of 0.3 means that our wealth dropped 30% from its initial value (one), before increasing (which it eventually must do, since $v_t \to \infty$ with probability one). Several other definitions of drawdown are used in the literature. A large drawdown means that $W^{\min}$ is small, *i.e.*, our wealth drops to a small value before growing.

The *drawdown risk* is defined as $\mathbf{Prob}(W^{\min} < \alpha)$, where $\alpha \in (0, 1)$ is a given target (undesired) minimum wealth. This risk depends on the bet vector b in a very complicated way. There is in general no formula for the risk in terms of b, but we can always (approximately) compute the drawdown risk for a given b using Monte Carlo simulation. As an example, a drawdown risk of 0.1 for $\alpha = 0.7$ means the probability of experiencing more than 30% drawdown is only 10%. The smaller the drawdown risk (with any target), the better.

3.3.1 Fractional Kelly gambling

It is well known that Kelly optimal bets can lead to substantial drawdown risk. One ad hoc method for handling this is to compute a Kelly optimal bet $b^\star$, and then use the so-called fractional Kelly [37] bet given by

$$b = f b^\star + (1 - f) e_n, \tag{3.4}$$

where $f \in (0, 1)$ is the fraction. The fractional Kelly bet scales down the (risky) bets by f. Fractional Kelly bets have smaller drawdowns than Kelly bets, at the cost of reduced growth rate. We will see that trading off growth rate and drawdown risk can be more directly (and better) handled.

3.3.2 Kelly gambling with drawdown risk

We can add a drawdown risk constraint to the Kelly gambling problem (3.1), to obtain the problem

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\log(r^T b) \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \\ & \mathbf{Prob}(W^{\min} < \alpha) < \beta, \end{array} \tag{3.5}$$

with variable b, where $\alpha, \beta \in (0, 1)$ are given parameters. The last constraint limits the probability of a drop in wealth to value α to be no more than β. For example, we might take $\alpha = 0.7$ and $\beta = 0.1$, meaning that we require the probability of a drawdown of more than 30% to be less than 10%. (This imposes a constraint on the bet vector b.)

Unfortunately the problem (3.5) is, as far as we know, a difficult optimization problem in general. In the next section we will develop a bound on the drawdown risk that results in a tractable convex constraint on b. We will see in numerical simulations that the bound is generally quite good.

3.4 Drawdown risk bound

In this section we derive a condition that bounds the drawdown risk. Consider any $\lambda > 0$ and bet b. For any $\alpha \in (0, 1)$ and $\beta \in (0, 1)$ that satisfies $\lambda = \log\beta/\log\alpha$ we have

$$\mathbf{E}(r^T b)^{-\lambda} \leq 1 \implies \mathbf{Prob}(W^{\min} < \alpha) < \beta. \tag{3.6}$$

In other words, if our bet satisfies $\mathbf{E}(r^T b)^{-\lambda} \leq 1$, then its drawdown risk $\mathbf{Prob}(W^{\min} < \alpha)$ less than β.

To see this, consider the stopping time

$$\tau = \inf\{t \geq 1 \,|\, w_t < \alpha\},$$

and note $\tau < \infty$ if and only if $W^{\min} < \alpha$. From Lemma 5 of the appendix, we get

$$1 \geq \mathbf{E}\left[\exp(-\lambda \log w_\tau - \tau \log \mathbf{E}(r^T b)^{-\lambda}) \mid \tau < \infty\right] \mathbf{Prob}(W^{\min} < \alpha).$$

Since $-\tau \log \mathbf{E}(r^T b)^{-\lambda} \geq 0$ when $\tau < \infty$, we have

$$1 \geq \mathbf{E}\left[\exp(-\lambda \log w_\tau) \mid \tau < \infty\right] \mathbf{Prob}(W^{\min} < \alpha).$$

Since $w_\tau < \alpha$ when $\tau < \infty$, we have

$$1 > \exp(-\lambda \log \alpha)\, \mathbf{Prob}(W^{\min} < \alpha).$$

So we have

$$\mathbf{Prob}(W^{\min} < \alpha) < \alpha^\lambda = \beta.$$

3.5 Risk-constrained Kelly gambling

Replacing the drawdown risk constraint in the problem (3.5) with the lefthand side of (3.6), with $\lambda = \log\beta/\log\alpha$, yields the risk-constrained Kelly gambling problem (RCK)

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\log(r^T b), \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \\ & \mathbf{E}(r^T b)^{-\lambda} \leq 1, \end{array} \tag{3.7}$$

with variable b. We refer to a solution of this problem as an RCK bet. The RCK problem is a *restriction* of problem (3.5), since it has a smaller feasible set: any b that is feasible for RCK must satisfy the drawdown risk constraint $\mathbf{Prob}(W^{\min} < \alpha) < \beta$. In the limit when either $\beta \to 1$ or $\alpha \to 0$ we get $\lambda \to 0$. For $\lambda = 0$, the second constraint is always satisfied, and the RCK problem (3.7) reduces to the (unconstrained) Kelly gambling problem (3.1).

Let us now show that the RCK problem (3.7) is convex. The objective is concave and the constraints $\mathbf{1}^T b = 1$, $b \geq 0$ are convex. To see that the function $\mathbf{E}(r^T b)^{-\lambda}$ is convex in b, we note that for $r^T b > 0$, $(r^T b)^{-\lambda}$ is a convex function of b; so the expectation $\mathbf{E}(r^T b)^{-\lambda}$ is a convex function of b (see [21, §3.2]). We mention that the last constraint can also be written as $\log \mathbf{E}(r^T b)^{-\lambda} \leq 0$, where the lefthand side is a convex function of b.

The RCK problem (3.7) is always feasible, since $b = e_n$ is feasible. Just as in the Kelly gambling problem, the bet vector $b = e_n$ is optimal for RCK (3.7) if and only if $\mathbf{E}\, r_i \leq 1$ for $i = 1, \ldots, n-1$. In other words we should not bet at all if all the bets are losers in expectation; conversely, if just one bet is a winner in expectation, the solution of the RCK problem will have positive growth rate (and of course respect the drawdown risk constraint). We show this in the appendix.

3.5.1 Risk aversion parameter

The RCK problem (3.7) depends on the parameters α and β only through $\lambda = \log\beta/\log\alpha$. This means that, for fixed λ, our one constraint $\mathbf{E}(r^T b)^{-\lambda} \leq 1$ actually gives us a family of drawdown constraints that must be satisfied:

$$\mathbf{Prob}(W^{\min} < \alpha) < \alpha^{\lambda} \tag{3.8}$$

holds for all $\alpha \in (0, 1)$. For example $\alpha = 0.7$ and $\beta = 0.1$ gives $\lambda = 6.46$; thus, our constraint also implies that the probability of a drop in wealth to $\alpha = 0.5$ (*i.e.*, we lose half our initial wealth) is no more than $(0.5)^{6.46} = 0.011$. Another interpretation of (3.8) is that our risk constraint actually bounds the entire CDF (cumulative distribution function) of $W^{\min}$: it stays below the function $\alpha \mapsto \alpha^{\lambda}$.

The RCK problem (3.7) can be used in two (related) ways. First, we can start from the original drawdown specifications, given by α and β, and then solve the problem using $\lambda = \log\beta/\log\alpha$. In this case we are guaranteed that the resulting bet satisfies our drawdown constraint. An alternate use model is to consider λ as a *risk-aversion parameter*; we vary it to trade off growth rate and drawdown risk.

This is very similar to traditional Markowitz portofolio optimization [83] [21, §4.4.1], where a risk aversion parameter is used to trade off risk (measured by portfolio return variance) and return (expected portfolio return). We will see another close connection between our method and Markowitz mean-variance portfolio optimization in § 3.6.

3.5.2 Light and heavy risk aversion regimes

In this section we give provide an interpretation of the drawdown risk constraint

$$\mathbf{E}(r^Tb)^{-\lambda} \leq 1 \tag{3.9}$$

in the light and heavy risk aversion regimes, which correspond to small and large values of λ, respectively.

Heavy risk aversion. In the heavy risk aversion regime, *i.e.*, in the limit $\lambda \to \infty$, the constraint (3.9) reduces to $r^Tb \geq 1$ almost surely. In other words, problem (3.7) only considers risk-free bets in this regime.

Light risk aversion. Next consider the light risk aversion regime, *i.e.*, in the limit $\lambda \to 0$. Note that constraint (3.9) is equivalent to

$$\frac{1}{\lambda}\log \mathbf{E}\exp(-\lambda\log(r^Tb)) \leq 0.$$

As $\lambda \to 0$ we have

$$\begin{aligned}\frac{1}{\lambda}\log \mathbf{E}\exp(-\lambda\log(r^Tb)) &= \frac{1}{\lambda}\log \mathbf{E}\left[1-\lambda\log(r^Tb)+\frac{\lambda^2}{2}(\log(r^Tb))^2+O(\lambda^3)\right]\\ &= -\mathbf{E}\log(r^Tb)+\frac{\lambda}{2}\mathbf{E}(\log(r^Tb))^2-\frac{\lambda}{2}(\mathbf{E}\log(r^Tb))^2+O(\lambda^2)\\ &= -\mathbf{E}\log(r^Tb)+\frac{\lambda}{2}\mathbf{var}\log(r^Tb)+O(\lambda^2).\end{aligned}$$

(In the context of stochastic control, $(1/\lambda)\log \mathbf{E}_X \exp(-\lambda X)$ is known as the exponential disutility or loss and λ as the risk-sensitivity parameter. This asymptotic expansion is well-known see *e.g.* [120, 121].) So constraint (3.9) reduces to

$$\frac{\lambda}{2}\,\mathbf{var}\log(r^Tb) \leq \mathbf{E}\log(r^Tb)$$

in the limit $\lambda \to 0$. Thus the (restricted) drawdown risk contraint (3.9) limits the ratio of variance to mean growth in this regime.

3.5.3 Computing RCK bets

Two outcomes. For the two outcome case we can easily solve the problem (3.7), almost analytically. The problem is

$$\begin{array}{ll} \text{maximize} & \pi\log(b_1P+(1-b_1))+(1-\pi)(1-b_1), \\ \text{subject to} & 0 \leq b_1 \leq 1, \\ & \pi(b_1P+(1-b_1))^{-\log\beta/\log\alpha}+(1-\pi)(1-b_1)^{-\log\beta/\log\alpha} \leq 1. \end{array} \tag{3.10}$$

If the solution of the unconstrained problem,

$$\left(\frac{\pi P-1}{P-1},\frac{P-\pi P}{P-1}\right),$$

satisfies the risk constraint, then it is the solution. Otherwise we reduce b_1 to find the value for which

$$\pi(b_1P+(1-b_1))^{-\lambda}+(1-\pi)(1-b_1)^{-\log\lambda}=1.$$

(This can be done by bisection since the lefthand side is monotone in b_1.) In this case the RCK bet is a fractional Kelly bet (3.4), for some $f < 1$.

Finite outcomes case. For the finite outcomes case we can restate the RCK problem (3.7) in a convenient and tractable form. We first take the log of the last constraint and get

$$\log\sum_{i=1}^{K}\pi_i(r_i^Tb)^{-\lambda} \leq 0,$$

we then write it as

$$\log\left(\sum_{i=1}^{K} \exp(\log \pi_i - \lambda \log(r_i^T b))\right) \leq 0.$$

To see that this constraint is convex we note that the log-sum-exp function is convex and increasing, and its arguments are all convex functions of b (since $\log(r_i^T b)$ is concave), so the lefthand side function is convex in b [21, §3.2]. Moreover, convex optimization software systems like CVX, CVXPY, and Convex.jl based on DCP (disciplined convex programming) can handle such compositions directly. Thus we have the problem

$$\begin{array}{ll} \text{maximize} & \sum_{i=1}^{K} \pi_i \log(r_i^T b), \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \\ & \log\left(\sum_{i=1}^{K} \exp(\log \pi_i - \lambda \log(r_i^T b))\right) \leq 0. \end{array} \tag{3.11}$$

In this form the problem is readily solved; its CVXPY specification is given in appendix B.

General returns. As with the Kelly gambling problem, we can solve the RCK problem (3.7) using a stochastic optimization method. We use a primal-dual stochastic gradient method (from [98, 97]) applied to the Lagrangian

$$L(b, \kappa) = -\mathbf{E} \log(r^T b) + \kappa(\mathbf{E}(r^T b)^{-\lambda} - 1), \tag{3.12}$$

with $b \in \Delta_\varepsilon$ and $\kappa \geq 0$. (As in the unconstrained Kelly optimization case, we make the technical assumption that $b_n^\star > \varepsilon > 0$.) In the appendix we show that the RCK problem (3.7) has an optimal dual variable $\kappa^\star$ for the constraint $\mathbf{E}(r^T b)^{-\lambda} \leq 1$, which implies that solving problem (3.7) is equivalent to finding a saddle point of (3.12). We also assume we know an upper bound M on the value of the optimal dual variable $\kappa^\star$.

Our method computes the iterates

$$\begin{aligned} \bar{b}^{(k+1)} =& \Pi\left(\bar{b}^{(k)} + t_k \frac{({r^{(k)}}^T\bar{b}^{(k)})^{\lambda} + \lambda\bar{\kappa}^{(k)}}{({r^{(k)}}^T\bar{b}^{(k)})^{\lambda+1}} r^{(k)}\right), \\ \bar{\kappa}^{(k+1)} =& \left(\bar{\kappa}^{(k)} + t_k \frac{1-({r^{(k)}}^T\bar{b}^{(k)})^{\lambda}}{({r^{(k)}}^T\bar{b}^{(k)})^{\lambda}}\right)_{[0,M]}, \end{aligned}$$

where the starting points $\bar{b}^{(1)}$ and $\bar{\kappa}^{(1)}$ are respectively in Δ_{ε} and $[0, M]$, $r^{(k)}$ are IID samples from the distribution of r, Π is Euclidean projection onto Δ_{ε}, and $(a)_{[0,M]}$ is projection onto $[0, M]$, *i.e.*,

$$(a)_{[0,M]} = \max\{0, \min\{M, a\}\}.$$

The step sizes $t_k > 0$ must satisfy

$$t_k \to 0, \qquad \sum_{k=1}^{\infty} t_k = \infty.$$

We use the (weighted) running averages

$$b^{(k)} = \frac{\sum_{i=1}^{k} t_i \bar{b}^{(i)}}{\sum_{i=1}^{k} t_i}, \qquad \kappa^{(k)} = \frac{\sum_{i=1}^{k} t_i \bar{\kappa}^{(i)}}{\sum_{i=1}^{k} t_i}$$

as our estimates of the optimal bet and $\kappa^{\star}$, respectively.

Again, this method can be slow to converge, but it always works; that is $\mathbf{E}\log(r^T b^{(k)})$ converges to the optimal value and $\max\{\mathbf{E}(r^T b^{(k)}))^{-\lambda} - 1, 0\} \to 0$. As in the unconstrained Kelly case, we do not know a priori how small ε should be, or how large M should be. We can choose a small ε and large M and later verify that $(\bar{b}^{(k)})_n > \varepsilon$, and $\bar{\kappa}^{(k)} < M$; if this holds, our guesses of ε and M were valid. Also as in the unconstrained case, batching can be used to improve the practical convergence. In this case, we replace our unbiased estimates of the gradients of the two expectations with an average over some number of them.

Finally, we mention that the optimality conditions can be independently checked.

As we show in Lemma 4 of the appendix, a pair $(b^\star, \kappa^\star)$ is a solution of the RCK problem if and only if it satisfies the following optimality conditions:

$$\begin{array}{l} \mathbf{1}^T b^\star = 1, \quad b^\star \geq 0, \quad \mathbf{E}(r^T b^\star)^{-\lambda} \leq 1 \\ \kappa^\star \geq 0, \quad \kappa^\star(\mathbf{E}(r^T b^\star)^{-\lambda} - 1) = 0 \\ \mathbf{E}\,\dfrac{r_i}{r^T b^\star} + \kappa^\star \lambda\, \mathbf{E}\,\dfrac{r_i}{(r^T b^\star)^{\lambda+1}} \left\{ \begin{array}{ll} \leq 1 + \kappa^\star\lambda & b_i = 0 \\ = 1 + \kappa^\star\lambda & b_i > 0. \end{array} \right. \end{array} \tag{3.13}$$

These conditions can be checked for a computed approximate solution of RCK, using Monte Carlo simulation to evaluate the expectations. (The method above guarantees that $\mathbf{1}^T b = 1$, $b \geq 0$, and $\kappa \geq 0$, so we only need to check the other three conditions.)

3.6 Quadratic approximation

In this section we form a quadratic approximation of the RCK problem (3.7), which we call the *quadratic RCK problem* (QRCK), and derive a close connection to Markowitz portfolio optimization. We use the notation $\rho = r - \mathbf{1}$ for the (random) excess return, so (with $\mathbf{1}^T b = 1$) we have $r^T b - 1 = \rho^T b$. Assuming $r^T b \approx 1$, or equivalently $\rho^T b \approx 0$, we have the (Taylor) approximations

$$\begin{aligned} \log(r^T b) &= \rho^T b - \frac{1}{2}(\rho^T b)^2 + O((\rho^T b)^3), \\ (r^T b)^{-\lambda} &= 1 - \lambda \rho^T b + \frac{\lambda(\lambda+1)}{2}(\rho^T b)^2 + O((\rho^T b)^3). \end{aligned}$$

Substituting these into the RCK problem (3.7) we obtain the QRCK problem

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\,\rho^T b - \frac{1}{2}\mathbf{E}(\rho^T b)^2 \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0 \\ & -\lambda\,\mathbf{E}\,\rho^T b + \frac{\lambda(\lambda+1)}{2}\mathbf{E}(\rho^T b)^2 \leq 0. \end{array} \tag{3.14}$$

This approximation of the RCK problem is a convex quadratic program (QP) which is readily solved. We expect the solution to be a good approximation of the RCK solution when the basic assumption $r^T b \approx 1$ holds.

This approximation can be useful when finding a solution to the RCK problem (3.7). We first estimate the first and second moments of ρ via Monte Carlo, and then solve the QRCK problem (3.14) (with the estimated moments) to get a solution $b^{\rm qp}$ and a Langrange multipler $\kappa^{\rm qp}$. We take these as good approximations for the solution of (3.7), and use them as the starting points for the primal-dual stochastic gradient method, *i.e.*, we set $b^{(1)} = b^{\rm qp}$ and $\kappa^{(1)} = \kappa^{\rm qp}$. This gives no theoretical advantage, since the method converges no matter what the initial points are; but it can speed up the convergence in practice.

We now connect the QRCK problem (3.14) to classical Markowitz portfolio selection. We start by defining $\mu = \mathbf{E}\,\rho$, the mean excess return, and

$$S = \mathbf{E}\,\rho\rho^T = \Sigma + (\mathbf{E}\,\rho)(\mathbf{E}\,\rho)^T,$$

the (raw) second moment of ρ (with Σ the covariance of the return). We say that an allocation vector b is a Markowitz portfolio if it solves

$$\begin{array}{ll} \text{maximize} & \mu^T b - \frac{\gamma}{2} b^T \Sigma b \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \end{array} \tag{3.15}$$

for some value of the (risk-aversion) parameter $\gamma \geq 0$. A solution to problem (3.14) is a Markowitz portfolio, provided there are no arbitrages. By no arbitrage we mean that $\mu^T b > 0$, $\mathbf{1}^T b = 1$, and $b \geq 0$, implies $b^T \Sigma b > 0$.

Let us show this. Let $b^{\rm qp}$ be the solution to the QRCK problem (3.14). By (strong) Lagrange duality [12], $b^{\rm qp}$ is a solution of

$$\begin{array}{ll} \text{maximize} & \mu^T b - \frac{1}{2} b^T S b + \nu(\mu^T b - \frac{\lambda+1}{2} b^T S b) \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0 \end{array}$$

for some $\nu \geq 0$, which we get by dualizing only the constraint $-\lambda\,\mathbf{E}\,\rho^T b + \frac{\lambda(\lambda+1)}{2}\,\mathbf{E}(\rho^T b)^2 \leq 0$. We divide the objective by $1+\nu$ and substitute $S = \mu\mu^T + \Sigma$ to get that $b^{\rm qp}$ is a solution of

$$\begin{array}{ll} \text{maximize} & \mu^T b - \frac{\eta}{2}(\mu^T b)^2 - \frac{\eta}{2} b^T \Sigma b \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \end{array} \tag{3.16}$$

for some $\eta > 0$. In turn, b^{qp} is a solution of

$$\begin{array}{ll} \text{maximize} & (1 - \eta \mu^T b^{\mathrm{qp}}) \mu^T b - \frac{\eta}{2} b^T \Sigma b \\ \text{subject to} & \mathbf{1}^T b = 1, \quad b \geq 0, \end{array} \tag{3.17}$$

since the objectives of problem (3.16) and (3.17) have the same gradient at b^{qp}. If $\mu^T b^{\mathrm{qp}} < 1/\eta$ then problem (3.17) is equivalent to problem (3.15) with $\gamma = \eta/(1 - \eta \mu^T b^{\mathrm{qp}})$.

Assume for contradiction that $\mu^T b^{\mathrm{qp}} \geq 1/\eta$, which implies $(b^{\mathrm{qp}})^T \Sigma b^{\mathrm{qp}} > 0$ by the no artibtrage assumption. Then for problem (3.17) the bet e_n achieves objective value 0, which is better than that of b^{qp}. As this contradicts the optimality of b^{qp}, we have $\mu^T b^{\mathrm{qp}} < 1/\eta$. So we conclude a solution to problem (3.14) is a solution to problem (3.15), *i.e.*, b^{qp} is a Markowitz portfolio.

3.7 Numerical simulations

In this section we report results for two specific problem instances, one with finite outcomes and one with infinite outcomes, but our numerical explorations show that these results are typical.

3.7.1 Finite outcomes

We consider a finite outcomes case with $n = 20$ (so there are 19 risky bets) and $K = 100$ possible outcomes. The problem data is generated as follows. The probabilities π_i, $i = 1, \ldots, K$ are drawn uniformly on $[0, 1]$ and then normalized so that $\sum_i \pi_i = 1$. The returns $r_{ij} \in \mathbf{R}_{++}$ for $i = 1, \ldots, K$ and $j = 1, \ldots, n-1$ are drawn from a uniform distribution in $[0.7, 1.3]$. Then, 30 randomly selected returns r_{ij} are set equal to 0.2 and other 30 equal to 2. (The returns r_{in} for $i = 1, \ldots, K$ are instead all set to 1.) The probability that a return vector r contains at least one "extreme" return (*i.e.*,

Bet	$\mathbf{E}\log(r^Tb)$	$e^{\lambda\log\alpha}$	$\mathbf{Prob}(W^{\min} < \alpha)$
Kelly	0.062	-	0.398
RCK, $\lambda = 6.456$	0.043	0.100	0.071
RCK, $\lambda = 5.500$	0.047	0.141	0.099

Table 3.1: Comparison of Kelly and RCK bets. Expected growth rate and drawdown risk are computed with Monte Carlo simulations.

equal to 0.2 or 2) is $1-(1-60/(19\cdot 100))^{n-1} \approx 0.45$. We generate a sample from this returns distribution for Monte Carlo simulations. For any given allocation vector b, these simulations consist of 10^5 trajectories of w_t for $t = 1, \ldots, 100$.

Comparison of Kelly and RCK bets

We compare the Kelly optimal bet with the RCK bets for $\alpha = 0.7$ and $\beta = 0.1$ ($\lambda = 6.456$). We then obtain the RCK bets for $\lambda = 5.500$, a value chosen so that we achieve risk close to the specified value $\beta = 0.1$ (as discussed in § 3.5.1). For each bet vector we carry out 10^5 Monte Carlo simulations of w_t for $t = 1, \ldots, 100$. This allows us to estimate (well) the associated risk probabilities. Table 3.1 shows the results. The second column gives the growth rate, the third column gives the bound on drawdown risk, and the last column gives the drawdown risk computed by Monte Carlo simulation.

The Kelly optimal bet experiences a drawdown exceeding our threshold $\alpha = 0.7$ around 40% of the time. For all the RCK bets the drawdown risk (computed by Monte Carlo) is less than our bound, but not dramatically so. (We have observed this over a wide range of problems.) The RCK bet with $\lambda = 6.456$ is *guaranteed* to have a drawdown probability not exceeding 10%; Monte Carlo simulation shows that it is (approximately) 7.3%. For the third bet vector in our comparison, we decreased the risk aversion parameter until we obtained a bet with (Monte Carlo computed) risk near the limit 10%.

The optimal value of the (hard) Kelly gambling problem with drawdown risk (3.9) must be less than 0.062 (the unconstrained optimal growth rate) and greater than

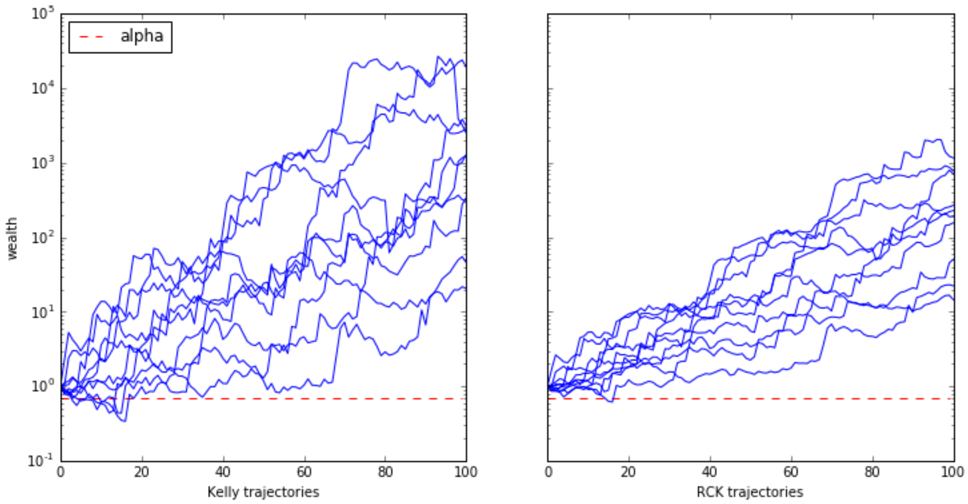

Figure 3.1: Wealth trajectories for the Kelly optimal bet (left) and the restricted risk-constrained bet with $\lambda = 5.5$.

0.043 (since our second bet vector is guaranteed to satisfy the risk constraint). Since our third bet vector has drawdown risk less than 10%, we can further refine this result to state that the optimal value of the (hard) Kelly gambling problem with drawdown risk (3.9) is between 0.062 and 0.047.

Figure 3.1 shows ten trajectories of w_t in our Monte Carlo simulations for the Kelly optimal bet (left) and the RCK bet obtained with $\lambda = 5.5$ (right). The dashed line shows the wealth threshold $\alpha = 0.7$. Out of these ten simulations, four of the Kelly trajectories dip below the threshold $\alpha = 0.7$, and one of the other trajectories does, which is consistent with the probabilities reported above.

Figure 3.2 shows the sample CDF of $W^{\min}$ over the 10^5 simulated trajectories of w_t, for the Kelly optimal bets and the RCK bets with $\lambda = 6.46$. The upper bound α^λ is also shown. We see that the risk bound is not bad, typically around 30% or so higher than the actual risk. We have observed this to be the case across many problem instances.

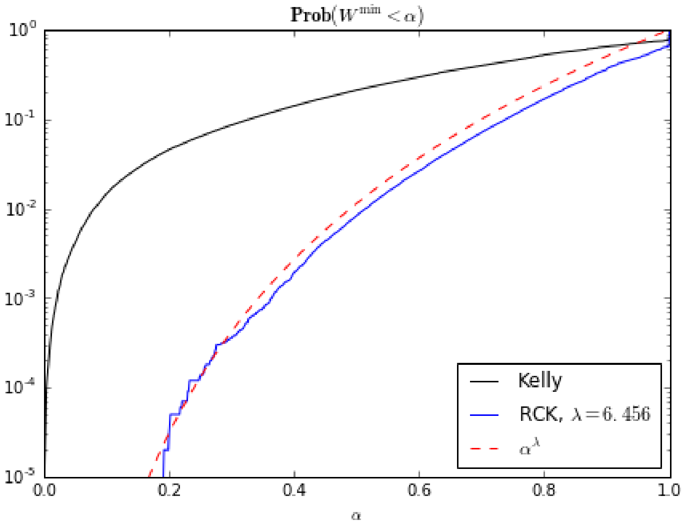

Figure 3.2: Sample CDF of $W^{\min}$ (*i.e.*, $\mathbf{Prob}(W^{\min} < \alpha)$) for the Kelly optimal bets and RCK bets with $\lambda = 6.46$. The upper bound α^λ is also shown.

Comparison of RCK and QRCK

Table 3.2 shows the Monte Carlo values of growth rate and drawdown risk for the QRCK bets with $\lambda = 6.456$ (to compare with the RCK solution in table 3.1). The QRCK bets come with no guarantee on the drawdown risk, but with $\lambda = 6.456$ the drawdown probability (evaluated by Monte Carlo) is less than $\beta = 0.1$. The value $\lambda = 2.800$ is selected so that the risk is approximately 0.10; we see that its growth rate is smaller than the growth rate of the RCK bet with the same drawdown risk.

Figure 3.3 shows the values of each b_i with $i = 1, \ldots, 20$, for the Kelly, RCK, and QRCK bets. We can see that the Kelly bet concentrates on outcome 4; the RCK and QRCK bets still make a large bet on outcome 4, but also spread their bets across other outcomes as well.

Bet	$\mathbf{E}\log(r^Tb)$	$e^{\lambda \log \alpha}$	$\mathbf{Prob}(W^{\min} < \alpha)$
QRCK, $\lambda = 0.000$	0.054	1.000	0.218
QRCK, $\lambda = 6.456$	0.027	0.100	0.027
QRCK, $\lambda = 2.800$	0.044	0.368	0.103

Table 3.2: Statistics for QRCK bets. Expected growth rate and drawdown risk are computed with Monte Carlo simulations.

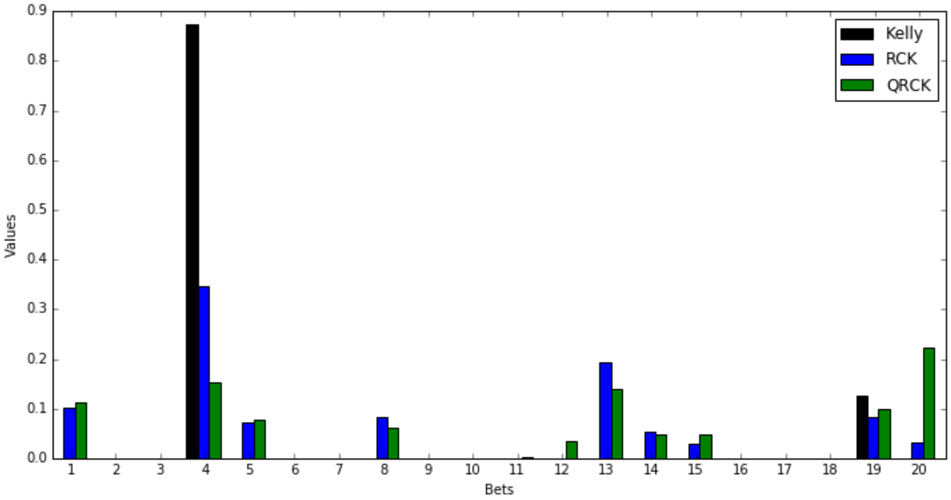

Figure 3.3: Comparison of the values b_i, $i = 1, \ldots, 20$, for different bet vectors. The RCK and QRCK bets are both obtained with $\lambda = 6.456$.

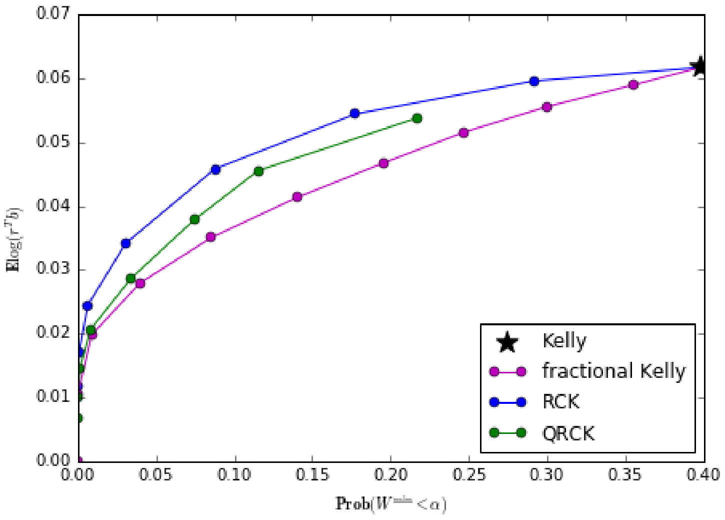

Figure 3.4: Trade-off of drawdown risk and expected growth rate for the RCK, QRCK, and fractional Kelly bets. The Kelly bet is also shown.

Risk-growth trade-off of RCK, QRCK, and fractional Kelly bets

In figure 3.4 we compare the trade-off between drawdown risk and expected growth rate of the RCK and QRCK problems for multiple choices of λ and the fractional Kelly bets (3.4) for multiple choices of f. The plots are akin to the risk-return tradeoff curves that are typical in Markowitz portfolio optimization. We see that RCK yields superior bets than QRCK and fractional Kelly (in some cases substantially better). For example, the Kelly fractional bet that achieves our risk bound 0.1 has a growth rate around 0.035, compared with RCK, which has a growth rate 0.047.

3.7.2 General returns

We show here an instance of the problem with an infinite distribution of returns, defined as a mixture of lognormals

$$r \sim \begin{cases} \log \mathcal{N}(\nu_1, \Sigma_1) & \text{w.p. } 0.5 \\ \log \mathcal{N}(\nu_2, \Sigma_2) & \text{w.p. } 0.5, \end{cases}$$

with $n = 20$, $\nu_1, \nu_2 \in \mathbf{R}^n$, and $\Sigma_1, \Sigma_2 \in \mathbf{S}^n_+$. We have $\nu_{1n} = \nu_{2n} = 0$, and the matrices Σ_1, Σ_2 are such that r_n has value 1 with probability 1.

We generate a sample of 10^6 observations from this returns distribution and use it to solve the Kelly, RCK, and QRCK problems. In the algorithms for solving Kelly and RCK we use step sizes $t_k = C/\sqrt{k}$ for some $C > 0$ and batching over 100 samples per iteration (so that we run 10^4 iterations). We initialize these algorithms with the QRCK solutions to speed up convergence. To solve the QRCK problem we compute the first and second moments of $\rho = r - 1$ using the same sample of 10^6 observations. We generate a separate sample of returns for Monte Carlo simulations, consisting of 10^4 simulated trajectories of w_t for $t = 1, \dots, 100$.

Comparison of RCK and Kelly bets

In our first test we compare the Kelly bet with RCK bets for two values of the parameter λ. The first RCK bet has $\lambda = 6.456$, which guarantees that the drawdown risk at $\alpha = 0.7$ is smaller or equal than $\beta = 0.1$. Table 3.3 shows that this is indeed the case, the Monte Carlo simulated risk is 0.08, not very far from the theoretical bound. The second value of λ is instead chosen so that the Monte Carlo risk is approximately equal to 0.1.

Figure 3.5 shows 10 simulated trajectories w_t for the Kelly bet and for the RCK bet with $\lambda = 5.700$. The wealth threshold $\alpha = 0.7$ is also shown. In this case 6 of the Kelly trajectories fall below $\alpha = 0.7$ and one of the RCK trajectories does, consistently with the values obtained above. Figure 3.6 shows the sample CDF of $W^{\min}$ for the Kelly bet and the RCK bet with $\lambda = 6.456$, and the theoretical bound given by α^λ. The bound α^λ is also shown.

Bet	$\mathbf{E}\log(r^Tb)$	$e^{\lambda\log\alpha}$	$\mathbf{Prob}(W^{\min}<\alpha)$
Kelly	0.077	-	0.569
RCK, $\lambda=6.456$	0.039	0.100	0.080
RCK, $\lambda=5.700$	0.043	0.131	0.101

Table 3.3: Comparison of simulated growth rate and drawdown risk for Kelly and RCK bets, in the infinite outcome case.

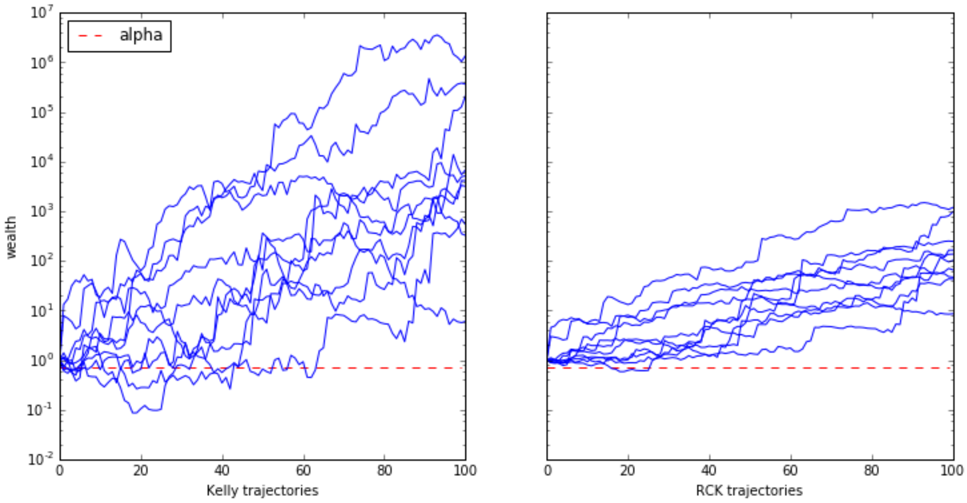

Figure 3.5: Wealth trajectories for the Kelly optimal bet (left) and the RCK bet with $\lambda = 5.700$, for the infinite returns distribution.

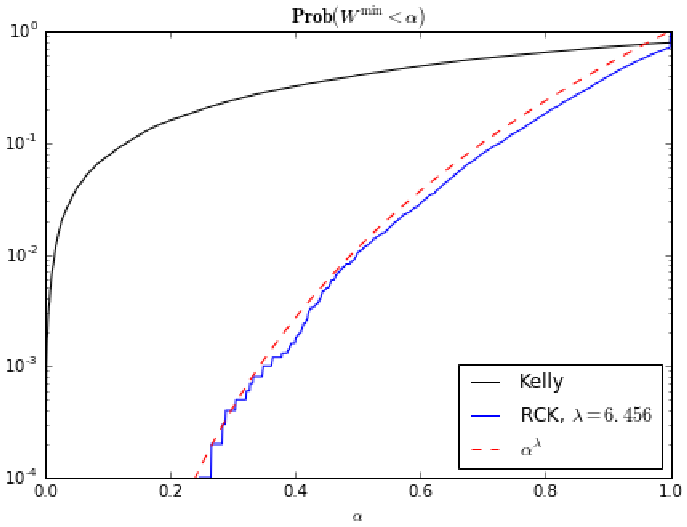

Figure 3.6: Sample CDF of $W^{\min}$ for the Kelly optimal bet and RCK bet with $\lambda = 6.46$, for the infinite returns distribution.

Risk-growth trade-off of RCK, QRCK, and fractional Kelly bets

We compare the Monte Carlo simulated drawdown risk ($\alpha = 0.7$) and expected growth rate of the Kelly, RCK, QRCK, and fractional Kelly bets. We select multiple values of λ (for RCK and QRCK) and f (for fractional Kelly) and plot them together in figure 3.7. We observe, as we did in the finite outcome case, that RCK yields superior bets than QRCK. This is particularly significant since QRCK is closely connected to the classic Markowitz portfolio optimization model, and this example resembles a financial portfolio selection problem (a bet with infinite return distributions). The fractional Kelly bets, in this case, show instead the (essentially) same performance as RCK.

Acknowledgments

We thank Andrea Montanari, B. Ross Barmish, and Minggao Gu for useful discussions.

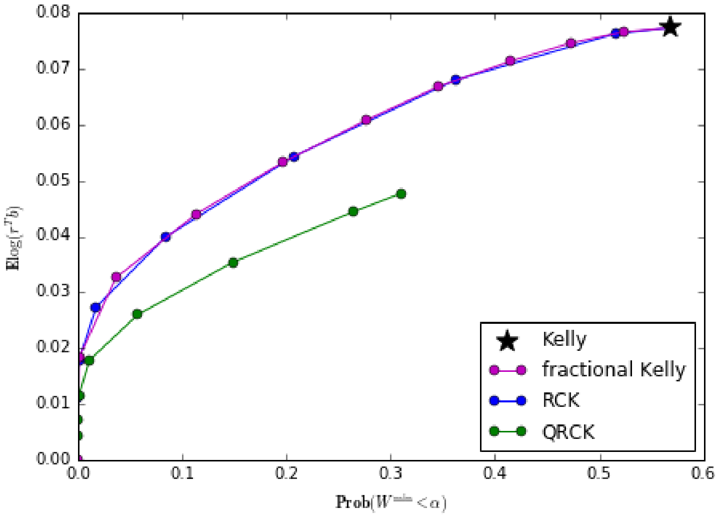

Figure 3.7: Trade-off of drawdown risk and expected growth rate for the RCK, QRCK, Kelly, and fractional Kelly bets, with the infinite return distribution.

Chapter 4

Volume Weighted Average Price Optimal Execution

In this chapter, we study the problem of optimal execution of a trading order under volume weighted average price (VWAP) benchmark, from the point of view of a risk-averse broker. The problem consists in minimizing mean-variance of the slippage, with quadratic transaction costs. We devise multiple ways to solve it, in particular we study how to incorporate the information coming from the market during the schedule. Most related works in the literature eschew the issue of imperfect knowledge of the total market volume. We instead incorporate it in our model. We validate our method with extensive simulation of order execution on real NYSE market data. Our proposed solution, using a simple model for market volumes, reduces by 10% the VWAP deviation RMSE of the standard "static" solution (and can simultaneously reduce transaction costs).

4.1 Introduction

Most literature on optimal execution focuses on the Implementation Shortfall (IS) objective, minimizing the execution price with respect to the market price at the moment the order is submitted. The seminal papers [13], [3] and [102] derive the optimal schedule for various risk preferences and market impact models. However most volume on the stock markets is traded with Volume Weighted Average Price (VWAP) orders, benchmarked to the average market price during the execution horizon [82]. Using this benchmark makes the problem much more compelling from a stochastic control standpoint and prompts the development of a richer model for the market dynamics. The problem of optimal trade scheduling for VWAP execution has been studied originally [73] in a static optimization setting (the schedule is fixed at the start of the day). This is intuitively suboptimal, since it ignores the new information coming as the schedule progresses. Some recent papers [65] [86] [51] extend the model and incorporate the new information coming to the market but rely on the crucial assumption that the total market volume is known beforehand. Other works [14] take a different route and focus on the empirical modeling of the market volumes. Other recent works [62] [29] [30, Chapter 9] study the stochastic control problem including a market impact term, while [77] takes a different approach and studies the optimal placement of market and limit orders for a VWAP objective. Our approach matches in complexity the most recent works in the literature with a key addition: we don't assume that the total market volume is known and instead treat it as a random variable. We also provide extensive empirical results to validate our work.

We define the problem and all relevant variables in § 4.2. In § 4.3 we derive a "static" optimal trading solution. In § 4.4 we develop a "dynamic" solution which uses the information coming from the market during the schedule in the best possible way: as our estimate of the total market volume improves we optimize our trading activity accordingly. In § 4.5 we detail the simulations of trading we performed, on real NYSE market data, using our VWAP solution algorithms. We conclude in § 4.6.

4.2 Problem formulation

We consider, from the point of view of a broker, the problem of executing a trading order issued by a client. The client decides to trade $C \in \mathbf{Z}_+$ shares of stock k over the course of a market day. By assuming $C > 0$ we restrict our analysis to "buy" orders. If we were instead interested in "sell" orders we would only need to change the appropriate signs. We don't explore the reasons for the client's order (it could be for rebalancing a portfolio, making new investments, *etc.*). The broker accepts the order and performs all the trades in the market to fulfill it. The broker has freedom in implementing the order (can decide when to buy and in what amount) but is constrained to cumulatively trade the amount C over the course of the day. When the order is submitted client and broker agree on an execution *benchmark* price which regulates the compensation of the broker and the sharing of risk. The broker is payed by the client an amount equal to the number of shares traded times the execution benchmark, plus fees (which we neglect). In turn, the broker pays for the trading activity in the market. Some choices of benchmark prices are:

- stock price at the start of the trading schedule. This gives rise to *implementation shortfall* execution ([13], [3]), in which the client takes no risk (since the benchmark price is fixed);

- stock price at day close. This type of execution can misalign the broker and client objectives. The broker may try to profit from the executions by pushing the closing price up or down, using the market impact of the trades;

- *volume weighted average price* (VWAP), the average stock price throughout the day weighted by market volumes. This is the most common benchmark price. It encourages the broker to spread the execution evenly across the market day, minimizing market impact and detectability of the order. It assigns most risk associated with market price movements to the client, so that the broker can focus exclusively on optimizing execution.

In this chapter we derive algorithms for optimal execution under the VWAP benchmark.

4.2.1 Definitions

We work for simplicity in discrete time. We consider a market day for a given stock, split in T intervals of the same length. In the following T is fixed to 390, so each interval is one minute long.

Volume We use the word *volume* to denote an integer number of traded shares (either by the market as a whole or by a single agent). We define $m_t \in \mathbf{R}_+$ for $t = 1, \dots, T$, the number of shares of the stock traded by the whole market in interval t, which is non-negative. We note that in reality the market volumes m_t are integer, not real numbers. This approximation is acceptable since the typical number of shares traded is much greater than 1 (if the interval length is 1 minute or more) so the integer rounding error is negligible. These market volumes are distributed according to a joint probability distribution

$$f_{m_{1:T}}(m_1, \dots, m_T).$$

In § 4.5.2 we propose a model for this joint distribution. We also define the total daily volume

$$V = \sum_{t=1}^{T} m_t$$

We call $u_t \in \mathbf{R}_+$ the number of shares of the stock that our broker trades in interval t, for $t = 1, \dots, T$. (Again we assume that the volumes are large enough so the rounding error is negligible.) By regulations these must be non-negative, so that all trades performed by the broker as part of the order have the same sign.

Price Let $p_t \in \mathbf{R}_{++}$ for $t = 1, \dots, T$ be the average market price for the stock in interval t. This is defined as the VWAP of all trades over interval t. (If during interval t there are $N_t > 0$ trades in the market, each one with volume $\omega_i \in \mathbf{Z}_{++}$ and price $\pi_i \in \mathbf{R}_{++}$, then $p_t = \sum_{i=1}^{N_t} \omega_i \pi_i / \sum_{i=1}^{N_t} \omega_i$.) If there are no trades during interval t then p_t is undefined and in practice we set it equal to the last available period price. We model this price process as a geometric random walk with zero drift. The initial price p_0 is a known constant. Then the price increments $\eta_t \equiv \frac{p_t - p_{t-1}}{p_{t-1}}$ for $t = 1, \dots, T$ are independent and distributed as

$$\eta_t \sim \mathcal{N}(0, \sigma_t),$$

where $\mathcal{N}$ is the Gaussian distribution. The period volatilities $\sigma_t \in \mathbf{R}_+$ for $t = 1, \dots, T$ are constants known from the start of the market day. We define the market VWAP price as

$$p_{\mathrm{VWAP}} = \frac{\sum_{t=1}^{T} m_t p_t}{V}. \tag{4.1}$$

Transaction costs We model the transaction costs by introducing the *effective* price $\hat{p}_t$, defined so that the whole cost of the trade at interval t is $u_t\hat{p}_t$. Our model captures instantaneous transaction costs, in particular the cost of the bid-ask spread, not the cost of long-term market impact. (For a detailed literature review on transaction costs and market impact see [16].) Let $s_t \in \mathbf{R}_{++}$ be the average fractional (as ratio of the stock price) bid-ask spread in period t. We assume the broker trades the volume u_t using an optimized trading algorithm that mixes optimally market and limit orders. The cost or proceeding per share of a buy market order is on average $p_t(1 + s_t/2)$ while for a limit order it is on average $p_t(1 - s_t/2)$. Let u_{LO} and u_{MO} be the portions of u_t executed via limit orders and market orders, respectively, so that $u_{\mathrm{LO}} + u_{\mathrm{MO}} = u_t$. We require that the algorithm uses trades of the same sign, so u_{LO}, u_{MO}, and u_t are all non-negative (consistently with the constraint we introduce in § 4.2.3). We assume that the fraction of market orders over the traded volume is proportional to the *participation rate*, defined as u_t/m_t. So

$$\frac{u_{\mathrm{MO}}}{u_t} = \frac{\alpha}{2}\frac{u_t}{m_t}$$

where the proportionality factor $\alpha \in \mathbf{R}_+$ depends on the specifics of the trading algorithm used. This is a reasonable assumption, especially in the limit of small participation rate. The whole cost or proceedings of the trade is

$$u_t\hat{p}_t = p_t\left(u_{\mathrm{LO}}\left(1 - \frac{s_t}{2}\right) + u_{\mathrm{MO}}\left(1 + \frac{s_t}{2}\right)\right)$$

which implies

$$\hat{p}_t = p_t\left(1 - \frac{s_t}{2} + \alpha\frac{s_t}{2}\frac{u_t}{m_t}\right). \tag{4.2}$$

We thus have a simple model for the effective price $\hat{p}_t$, linear in u_t. This gives rise to *quadratic* transaction costs, a reasonable approximation for the stock markets ([16], [78]).

4.2.2 Problem objective

Consider the cash flow for the broker, equal to the payment he receives from the client minus the cost of trading

$$Cp_{\text{VWAP}} - \sum_{t=1}^{T} u_t \hat{p}_t .$$

In practice there would also be fees but we neglect them. The trading industry usually defines the *slippage* as the negative of this cash flow. It represents the amount by which the order execution price misses the benchmark. (The choice of sign is conventional so that the optimization problem consists in minimizing it). We instead define the slippage as

$$S \equiv \frac{\sum_{t=1}^{T} u_t \hat{p}_t - Cp_{\text{VWAP}}}{Cp_{\text{VWAP}}}, \tag{4.3}$$

normalizing by the value of the order. We need this in order to compare the slippage between different orders. By substituting the expressions defined above we get

$$\begin{aligned} S = \left(\sum_{t=1}^{T} \left[u_t p_t \left(1 - \frac{s_t}{2} + \alpha \frac{s_t}{2} \frac{u_t}{m_t} \right) \right] - C \frac{\sum_{t=1}^{T} m_t p_t}{V} \right) \Big/ Cp_{\text{VWAP}} = \\ \sum_{t=1}^{T} \left[\frac{p_t}{p_{\text{VWAP}}} \left(\frac{u_t}{C} - \frac{m_t}{V} \right) \right] + \sum_{t=1}^{T} \frac{p_t s_t}{2 p_{\text{VWAP}}} \left(\alpha \frac{u_t^2}{C m_t} - \frac{u_t}{C} \right) \simeq \\ \sum_{t=1}^{T-1} \left[\eta_{t+1} \left(\frac{\sum_{\tau=1}^{t} m_\tau}{V} - \frac{\sum_{\tau=1}^{t} u_\tau}{C} \right) \right] + \sum_{t=1}^{T} \frac{s_t}{2} \left(\alpha \frac{u_t^2}{C m_t} - \frac{u_t}{C} \right) \end{aligned} \tag{4.4}$$

where we used the two approximations (both first order, reasonable on a trading horizon of one day)

$$\frac{p_t - p_{t-1}}{p_{\text{VWAP}}} \simeq \frac{p_t - p_{t-1}}{p_{t-1}} = \eta_t \tag{4.5}$$

$$\frac{p_t s_t}{p_{\text{VWAP}}} \simeq s_t . \tag{4.6}$$

We model the broker as a standard risk-averse agent, so that the objective function is to minimize

$$\mathbf{E}\, S + \lambda\, \mathbf{var}(S)$$

for a given risk-aversion parameter $\lambda \geq 0$. These expectation and variance operators apply to all sources of randomness in the system, *i.e.*, the market volumes m and market prices p, which are independent under our model. The expected value of the slippage is

$$\underset{m,p}{\mathbf{E}}\, S = \underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{E}}\, S = \underset{m}{\mathbf{E}}\left[\sum_{t=1}^{T} \frac{s_t}{2}\left(\alpha \frac{u_t^2}{C m_t} - \frac{u_t}{C}\right)\right] \tag{4.7}$$

since the price increments have zero mean. Note that we leave expressed the expectation over market volumes. The variance of the slippage is

$$\underset{m,p}{\mathbf{var}}\, S = \underset{m,p}{\mathbf{E}}\left[\left(S - \underset{m,p}{\mathbf{E}}\, S\right)^2\right] = \underset{m,p}{\mathbf{E}}\, S^2 - \left(\underset{m,p}{\mathbf{E}}\, S\right)^2 =$$

$$\underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{E}}\, S^2 - \left(\underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{E}}\, S\right)^2 - \underset{m}{\mathbf{E}}(\underset{p}{\mathbf{E}}\, S)^2 + \underset{m}{\mathbf{E}}(\underset{p}{\mathbf{E}}\, S)^2 = \underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{var}}(S) + \underset{m}{\mathbf{var}}(\underset{p}{\mathbf{E}}\, S). \tag{4.8}$$

The first term is

$$\underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{var}}(S) = \underset{m}{\mathbf{E}}\,\underset{p}{\mathbf{E}}\left[\left(\sum_{t=1}^{T-1} \eta_{t+1}\left(\frac{\sum_{\tau=1}^{t} m_t}{V} - \frac{\sum_{\tau=1}^{t} u_t}{C}\right)\right)^2\right] =$$

$$\underset{m}{\mathbf{E}}\left[\sum_{t=1}^{T-1} \sigma_{t+1}^2 \left(\frac{\sum_{\tau=1}^{t} m_t}{V} - \frac{\sum_{\tau=1}^{t} u_t}{C}\right)^2\right] \tag{4.9}$$

which follows from independence of the price increment. The second term is

$$\underset{m}{\mathbf{var}}(\underset{p}{\mathbf{E}}\, S) = \underset{m}{\mathbf{var}}\left(\sum_{t=1}^{T} \frac{s_t}{2}\left(\alpha \frac{u_t^2}{C m_t} - \frac{u_t}{C}\right)\right). \tag{4.10}$$

We drop the second term and only keep the first one, so that the resulting optimization problem is tractable. We motivate this by assuming, as in [51], that the second term of the variance is negligible when compared to the first. This is validated *ex-post*[1] by

[1] In the rest of the chapter we derive multiple ways to solve the optimization problem of minimizing the objective (4.11). For these different solution methods, the empirical value of (4.10) is between 1% and 5% of the value of (4.9), so our approximation is valid. The results are detailed in § 4.5.6.

our empirical studies in § 4.5. We thus get

$$\mathop{\mathbf{E}}_{m,p} S + \lambda \mathop{\mathbf{var}}_{m,p}(S) \simeq \sum_{t=1}^{T} \mathop{\mathbf{E}}_{m} \left[\frac{s_t}{2} \left(\alpha \frac{u_t^2}{C m_t} - \frac{u_t}{C} \right) + \lambda \sigma_t^2 \left(\frac{\sum_{\tau=1}^{t-1} m_t}{V} - \frac{\sum_{\tau=1}^{t-1} u_t}{C} \right)^2 \right]. \tag{4.11}$$

We note that the objective function separates in a sum of terms per each time step, a key feature we will use to apply the *dynamic programming* optimization techniques in § 4.4.

4.2.3 Constraints

We consider the constraints that apply to the optimization problem. The optimization variables are u_t for $t = 1, \dots, T$. We require that the executed volumes sum to the total order size C

$$\sum_{t=1}^{T} u_t = C. \tag{4.12}$$

We then impose that all trades have positive sign (buys)

$$u_t \geq 0, \quad t = 1, \dots, T. \tag{4.13}$$

(If we were executing a sell order, $C < 0$, we would have all $u_t \leq 0$.) This is a regulatory requirement for institutional brokers in most markets, essentially as a precaution against market manipulation. It is a standard constraint in the literature about VWAP execution.

4.2.4 Optimization paradigm

The price increments η_t and market volumes m_t are stochastic. The volumes u_t instead are chosen as the solution of an optimization problem. This problem can be cast in several different ways. We define the information set I_t available at time t

$$I_t \equiv \{(p_1, m_1, u_1), \dots, (p_{t-1}, m_{t-1}, u_{t-1})\}. \tag{4.14}$$

By causality, we know that when we choose the value of u_t we can use, at most, the information contained in I_t. In § 4.3 we formulate the optimization problem and provide an optimal solution for the variables u_t in the case we *do not* access anything from the information set I_t when choosing u_t. The u_t are chosen using only information available before the trading starts. We call this a *static* solution (or *open loop* in the language of control). In § 4.4 instead we develop an optimal *policy* which can be seen as a sequence of functions ψ_t of the information set available at time t

$$u_t = \psi_t(I_t).$$

We develop it in the framework on dynamic programming and we call it *dynamic* solution (or *closed loop*).

4.3 Static solution

We consider a procedure to solve the problem described in § 4.2 without accessing the information sets I_t. We call this solution *static* since it is fixed at the start of the trading period. (It is computed using only information available before the trading starts.) This is the same assumption of [73] and corresponds to the approach used by many practitioners. Our model is however more flexible than [73], it incorporates variable bid-ask spread and a sophisticated transaction cost model. Still, it has an extremely simple numericaly solution that leverages convex optimization [21] theory and software.

We start by the optimization problem with objective function (4.11) and the two constraints (4.12) and (4.13)

$$\begin{array}{ll} \text{minimize}_u & \mathbf{E}_{m,p} S + \lambda \, \mathbf{var}_{m,p}(S) \\ \text{s.t.} & \sum_{t=1}^T u_t = C \\ & u_t \geq 0, \quad t = 1, \ldots, T. \end{array}$$

We remove a constant term from the objective and write the problem in the equivalent form

$$\begin{array}{ll} \text{minimize}_u & \sum_{t=1}^T \left[\frac{s_t}{2C} \left(\alpha u_t^2 \kappa_t - u_t \right) + \lambda \sigma_t^2 \left(\left(\frac{\sum_{\tau=1}^{t-1} u_t}{C} \right)^2 - 2M_t \frac{\sum_{\tau=1}^{t-1} u_t}{C} \right) \right] \\ \text{s.t.} & \sum_{t=1}^T u_t = C \\ & u_t \geq 0, \quad t = 1, \dots, T \end{array} \tag{4.15}$$

where M_t and κ_t are the constants

$$M_t = \mathop{\mathbf{E}}_m \left[\frac{\sum_{\tau=1}^{t-1} m_t}{V} \right], \quad \kappa_t = \mathop{\mathbf{E}}_m \left[\frac{1}{m_t} \right]$$

for $t = 1, \dots, T$. In this form, the problem is a standard quadratic program [21] and can be solved efficiently by open-source solvers such as ECOS [43] using a symbolic convex optimization suite like CVX [58] or CVXPY [41].

4.3.1 Constant spread

We consider the special case of constant spread, $s_1 = \cdots = s_T$, which leads to a great simplification of the solution. The convex problem (4.15) has the form

$$\begin{array}{ll} \text{minimize}_u & \sum_{t=1}^T \frac{s_t}{2C} \left(\alpha u_t^2 \kappa_t - u_t \right) + \lambda \left(\sum_{t=1}^T \sigma_t^2 \left(U_t^2 - 2M_t U_t / C \right) \right) \equiv \phi(u) + \lambda \psi(u) \\ \text{s.t.} & u \in \mathcal{C} \end{array}$$

where $U_t = \sum_{\tau=1}^{t-1} u_t / C$ for each $t = 1, \dots, T$, and $\mathcal{C}$ is the convex feasible set. We separate the problem into two subproblems considering each of the two terms of the objective. The first one is

$$\begin{array}{ll} \text{minimize}_u & \phi(u) \\ \text{s.t.} & u \in \mathcal{C} \end{array}$$

which is equivalent to (since the spread is constant and $\alpha > 0$)

$$\begin{array}{ll} \text{minimize}_u & \sum_{t=1}^T u_t^2 \kappa_t \\ \text{s.t.} & u \in \mathcal{C} \end{array}$$

The optimal solution is ([21], Lagrange duality)

$$u_t^\star = C\frac{1/\kappa_t}{\sum_{t=1}^T 1/\kappa_t}, \quad t = 1, \dots, T.$$

We approximate $\kappa_t = \mathbf{E}_m\left[1/m_t\right] \simeq 1/\mathbf{E}_m[m_t]$ and thus

$$u_t^\star \simeq C\frac{\mathbf{E}_m[m_t]}{\sum_{t=1}^T \mathbf{E}_m[m_t]} \simeq C\,\mathbf{E}_m\left[\frac{m_t}{V}\right], \quad t = 1, \dots, T.$$

The second problem is

$$\begin{array}{ll} \text{minimize}_u & \psi(u) \equiv \sum_{t=1}^T \sigma_t^2 \left(U_t^2 - 2M_tU_t/C\right) \\ & u \in \mathcal{C} \end{array}$$

we choose the U_t such that $\sigma_t^2\left(U_t - M_t\right) = 0$ so $U_t = M_t$ for $t = 1, \dots, T$. The values of $u_1, \dots, u_{T-1}$ are thus fixed, and we choose the final volume u_T so that $u_T = C - CU_T$. The first order condition of the objective function is satisfied, and these values of $u_1, \dots, u_T$ are feasible (since M_t is non-decreasing in t and $M_T \leq 1$). It follows that this is an optimal solution, it has values $u_t^\star = C\,\mathbf{E}_m\left[m_t/V\right]$ for $t = 1, \dots, T$.

Consider now the original problem. Its objective is a convex combination (apart from a constant factor) of the objectives of two convex problem above and all three have the same constraints set. Since the two subproblems share an optimal solution $u^\star$, it follows that $u^\star$ is also an optimal solution for the combined problem. Thus, an the optimal solution of (4.15) in the case of constant spread is

$$u_t^\star = C\,\mathbf{E}_m\left[\frac{m_t}{V}\right] \quad t = 1, \dots, T. \tag{4.16}$$

This is equivalent to the solution derived in [73] and is the standard in the brokerage industry. In our model this solution arises as the special case of constant spread, in general we could derive more sophisticated static solutions. We also note that we introduced the approximation $\kappa_t = \mathbf{E}_m\left[1/m_t\right] \simeq 1/\mathbf{E}_m[m_t]$. (In practice, estimating $\mathbf{E}_m\left[1/m_t\right]$ would require a more sophisticated model of market volumes than $\mathbf{E}_m\left[m_t/V\right]$). We thus expect to lose some efficiency in the optimization of the trading costs. However, with respect to the minimization of the variance of S (if $\lambda \to \infty$ or $s = 0$), this solution is indeed optimal. In the following we compare the performances of (4.16) and of the *dynamic solution* developed in § 4.4.

4.4 Dynamic solution

We develop a solution of the problem that uses all the information available at the time each decision is made, *i.e.*, a sequence of functions $\psi_t(I_t)$ where I_t is the information set available at time t (as defined in (4.14)). We work in the framework of Dynamic Programming (DP) [11], summarized in § 4.4.1. In particular we fit our problem in the special case of linear dynamics and quadratic costs, described in § 4.4.2. However we can't apply standard DP because the random shocks affecting the system at different times are not conditionally independent (the market volumes have a joint distribution). We instead use the approximate procedure of [116], summarized in § 4.4.3. In § 4.4.4 we finally write our optimization problem, defining the state, action and costs, and in § 4.4.5 we derive its solution.

4.4.1 Dynamic programming

We summarize here the standard formalism of dynamic programming, following [11]. Suppose we have a state variable $x_t \in \mathcal{X}$ defined for $t = 1, ..., T + 1$ with x_1 known. Our decision variables are $u_t \in \mathcal{U}$ for $t = 1, ..., T$ and each u_t is chosen as a function of the current state, $u_t = \mu_t(x_t)$. (We use the same symbol as the volumes traded at time t since in the following they coincide.) The randomness of the system is modeled by a series of IID random variables $w_t \in \mathcal{W}$, for $t = 1, ..., T$. The dynamics is described by a series of functions

$$x_{t+1} = f_t(x_t, u_t, w_t),$$

at every stage we incur the cost

$$g_t(x_t, u_t, w_t),$$

and at the end of the decision process we have a final cost

$$g_{T+1}(x_{T+1}).$$

Our objective is to minimize

$$J = \mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t, u_t, w_t) + g_{T+1}(x_{T+1})\right].$$

We solve the problem by *backward induction*, defining the *cost-to-go* function v_t at each time step t

$$v_t(x) = \min_u \mathbf{E}[g_t(x, u, w_t) + v_{t+1}(f_t(x, u, w_t))], \quad t = 1, \ldots, T. \tag{4.17}$$

This recursion is known as *Bellman* equation. The final condition is fixed by

$$v_{T+1}(\cdot) = g_{T+1}(\cdot).$$

It follows that the optimal action at time t is given by the solution

$$u_t = \operatorname*{argmin}_u \mathbf{E}[g_t(x_t, u, w_t) + v_{t+1}(f_t(x_t, u, w_t))]. \tag{4.18}$$

In general, these equations are not solvable since the iteration that defines the functions v_t requires an amount of computation exponential in the dimension of the state space, action space, and number of time steps (*curse of dimensionality*). However some special forms of this problem have closed form solutions. We see one in the following section.

4.4.2 Linear-quadratic stochastic control

Whenever the dynamics functions f_t are stochastic affine and the cost functions are stochastic quadratic, the problem of § 4.4.1 has an analytic solution [18]. We call this *Linear-Quadratic Stochastic Control* (LQSC). We define the state space $\mathcal{X} = \mathbf{R}^n$, the action space $\mathcal{U} = \mathbf{R}^m$ for some $n, m > 0$. The disturbances are independent with known distributions and belong to a general set $\mathcal{W}$. For $t = 1, \ldots, T$ the system dynamics is described by

$$x_{t+1} = f_t(x_t, u_t, w_t) = A_t(w_t)x_t + B_t(w_t)u_t + c_t(w_t), \quad t = 1, ..., T$$

with matrix functions $A_t(\cdot) : \mathcal{W} \to \mathbf{R}^{n \times n}$, $B_t(\cdot) : \mathcal{W} \to \mathbf{R}^{n \times m}$, and $c_t(\cdot) : \mathcal{W} \to \mathbf{R}^n$. The stage costs are

$$g_t(x_t, u_t, w_t) = x_t^T Q_t(w_t)x_t + q_t(w_t)^T x_t + u_t^T R_t(w_t)u_t + r_t(w_t)^T u_t$$

with matrix functions $Q_t(\cdot) : \mathcal{W} \to \mathbf{R}^{n \times n}$, $q_t(\cdot) : \mathcal{W} \to \mathbf{R}^n$, $R_t(\cdot) : \mathcal{W} \to \mathbf{R}^{m \times m}$, and $r_t(\cdot) : \mathcal{W} \to \mathbf{R}^m$. The final cost is a quadratic function of the final state

$$g_{T+1}(x_{T+1}) = x_{T+1}^T Q_{T+1} x_{T+1} + q_{T+1}^T x_{T+1}.$$

The main result of the theory on linear-quadratic problems [11] is that the optimal policy $\mu_t(x_t)$ is a simple affine function of the problem parameters and can be obtained analytically

$$\mu_t(x_t) = K_t x_t + l_t, \quad t = 0, ..., T - 1, \tag{4.19}$$

where $K_t \in \mathbf{R}^{m \times n}$ and $l_t \in \mathbf{R}^m$ depend on the problem parameters. In addition, the cost-to-go function is a quadratic function of the state

$$v_t(x_t) = x_t^T D_t x_t + d_t^T x_t + b_t \tag{4.20}$$

where $D_t \in \mathbf{R}^{n \times n}$, $d_t \in \mathbf{R}^n$, and $b_t \in \mathbf{R}$ for $t = 1, \ldots, T$. We derive these results solving the Bellman equations (4.17) by backward induction. These are known as *Riccati equations*, reported in Appendix C.1.

4.4.3 Conditionally dependent disturbances

We now consider the case in which the disturbances are *not* independent, and we can't apply the Bellman iteration of § 4.4.1. Specifically, we assume that the disturbances have a joint distribution described by a density function

$$f_w(\cdot) : \mathcal{W} \times \cdots \times \mathcal{W} \to [0, 1].$$

One approach to solve this problem is to *augment* the state x_t, by including the disturbances observed up to time t. This causes the computational complexity of the solution to grow exponentially with the increased dimensionality (curse of dimensionality). Some *approximate dynamic programming* techniques can be used to solve the augmented problem [11] [107]. We take instead the approximate approach developed in [116], called *shrinking-horizon dynamic programming* (SHDP), which performs reasonably well in practice and leads to a tractable solution. (It can be seen as an extension of *model predictive control*, known to perform well in a variety of scenarios [7] [75] [85] [19]).

We now summarize the approach. Assume we know the density of the future disturbances $w_t, \ldots, w_T$ conditioned on the observed ones

$$f_{w|t}(w_t, \ldots, w_T) : \mathcal{W} \times \cdots \times \mathcal{W} \to [0, 1].$$

(If $t = 1$ this is the unconditional density.) We derive the marginal density of each future disturbance, by integrating over all others,

$$\hat{f}_{w_t|t}(w_t), \ldots, \hat{f}_{w_T|t}(w_T).$$

We use the product of these marginals to approximate the density of the future disturbances, so they all are independent. We then compute the cost-to-go functions with backwards induction using the Bellman equations (4.17) and (4.18), where the expectations over each disturbance w_τ are taken on the conditional marginal density

$\hat{f}_{w_\tau|t}$. The equations (4.17) for the cost-to-go function become (note the subscript $\cdot|t$)

$$v_{\tau|t}(x) = \min_u \mathop{\mathbf{E}}_{\hat{f}_{w_\tau|t}} [g_\tau(x,u,w_\tau) + v_{\tau+1|t}(f_\tau(x,u,w_\tau))], \tag{4.21}$$

for all times $\tau = t, \ldots, T$, with the usual final condition. Similarly, the equations (4.18) for the optimal action become

$$u_t = \operatorname*{argmin}_u \mathop{\mathbf{E}}_{\hat{f}_{w_t|t}} [g_t(x_t,u,w_t) + v_{t+1|t}(f_t(x_t,u,w_t))] \tag{4.22}$$

for all times $\tau = t, \ldots, T$. We *only* use the solution u_t at time t. In fact when we proceed to the next time step $t+1$ we rebuild the whole sequence of cost-to-go functions $v_{t+1|t+1}(x), \ldots, v_{T|t+1}(x)$ using the updated marginal conditional densities and then solve (4.22) to get u_{t+1}. With this framework we can solve the VWAP problem we developed in § 4.2.

4.4.4 VWAP problem as LQSC

We now formulate the problem described in § 4.2 in the framework of § 4.4.2. For $t = 1, \ldots, T+1$ we define the state as:

$$x_t = \begin{pmatrix} \sum_{\tau=1}^{t-1} u_\tau \\ \sum_{\tau=1}^{t-1} m_\tau \end{pmatrix}, \tag{4.23}$$

so that $x_1 = (0,0)$. The action is u_t, the volume we trade during interval t, as defined in § 4.2.

The disturbance is

$$w_t = \begin{pmatrix} m_t \\ V \end{pmatrix} \tag{4.24}$$

where the second element is the total market volume $V = \sum_{t=1}^T m_t$. With this definition the disturbances are not conditionally independent. In § 4.4.5 we study their joint and marginal distributions. We note that V, the second element of each w_t, is not observed after time t. (The theory we developed so far does not require the disturbances w_t to be observed, the Bellman equations only need expected values of functions of w_t.) For $t = 1, ..., T$ the state transition consists in

$$x_{t+1} = x_t + \begin{pmatrix} u_t \\ m_t \end{pmatrix}.$$

So that the dynamics matrices are

$$\begin{aligned} A_t(w_t) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \equiv I, \\ B_t(w_t) &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \equiv e_1, \\ c_t(w_t) &= \begin{pmatrix} 0 \\ m_t \end{pmatrix}. \end{aligned}$$

The objective funtion (4.11) can be written as $\mathbf{E}_m \sum_{t=1}^{T} g_t(x_t, u_t, w_t)$ where each stage cost is given by

$$g_t(x_t, u_t, w_t) = \frac{s_t}{2}\left(\alpha \frac{u_t^2}{Cm_t} - \frac{u_t}{C}\right) + \lambda \sigma_t^2 x_t^T \begin{pmatrix} \frac{1}{C^2} & -\frac{1}{CV} \\ -\frac{1}{CV} & \frac{1}{V^2} \end{pmatrix} x_t.$$

The quadratic cost function terms are thus

$$\begin{aligned} Q_t(w_t) &= \lambda \sigma_t^2 \begin{pmatrix} \frac{1}{C^2} & -1/CV \\ -1/CV & 1/V^2 \end{pmatrix} \\ q_t(w_t) &= 0 \\ R_t(w_t) &= \frac{\alpha s_t}{2Cm_t} \\ r_t(w_t) &= -\frac{s_t}{2C} \end{aligned}$$

for $t = 1, \ldots, T$. The constraint that the total executed volume is equal to C imposes the last action

$$u_T = \mu_T(x_T) = C - \sum_{t=1}^{T-1} u_t, \equiv K_T x_t + l_t$$

with

$$\begin{aligned} K_T &= -e_1^T \\ l_T &= C. \end{aligned}$$

This in turn fixes the value function at time T

$$v_T(x_T) = \mathbf{E}\, g_T(x_T, K_T x_t + l_t, w_t), \tag{4.25}$$

so we can treat x_T as our final state and only consider the problem of choosing actions up to u_{T-1}. We are left with the constraint $u_t \geq 0$ for $t = 1, \ldots, T$. Unfortunately this can not be enforced in the LQSC formalism. We instead take the *approximate dynamic programming* approach of [70]. We allow u_t to get negative sign and then project it on the set of feasible solutions. For every $t = 1, \ldots, T$ we compute

$$\max(u_t, 0)$$

and use it, instead of u_t, for our trading schedule. This completes the formulation of our optimization problem into the linear-quadratic stochastic control framework. We now focus on its solution, using the approximate approach of § 4.4.3.

4.4.5 Solution in SHDP

We provide an approximate solution of the problem defined in § 4.4.4 using the framework of shinking-horizon dynamic programming (summarized in § 4.4.3). Consider a fixed time $t = 1, \ldots, T-1$. We note that (unlike the assumption of [116]) we do not observe the sequence of disturbances $w_1, \ldots, w_{t-1}$, because the total volume V is not known until the end of the day. We only observe the sequence of market volumes $m_1, \ldots, m_{t-1}$.

If $f_m(m_1, \ldots, m_T)$ is the joint distribution of the market volumes, then the joint distribution of the disturbances is

$$f_w(w_1,\dots,w_t) = f_m(e_1^T w_1,\dots,e_1^T w_T) \times \mathbf{1}_{\{e_2^T w_1 = V\}} \times \cdots \times \mathbf{1}_{\{e_2^T w_T = V\}} \times \mathbf{1}_{\{V = \sum_{\tau=1}^T e_1^T w_\tau\}}$$

where $e_1 = (1,0)$, $e_2 = (0,1)$, and the function $\mathbf{1}_{\{\cdot\}}$ has value 1 when the condition is true and 0 otherwise. We assume that our market volumes model also provides the conditional density $f_{m|t}(m_t,\dots,m_T)$ of $m_t,\dots,m_T$ given $m_1,\dots,m_{t-1}$. The conditional distribution of V given $m_1,\dots,m_{t-1}$ is

$$f_{V|t}(V) = \int \cdots \int f_{m|t}(m_t,\dots,m_T)\mathbf{1}_{\{V = \sum_{\tau=1}^T m_\tau\}} dm_t \cdots dm_T$$

(where the first $t-1$ market volumes are constants and the others are integration variables). Let the marginal densities be

$$\hat{f}_{m_t|t}(m_t),\dots,\hat{f}_{m_T|t}(m_T).$$

The marginal conditional densities of the disturbances are thus

$$\hat{f}_{w_\tau|t}(\cdot) = \hat{f}_{m_\tau|t}(\cdot) \times f_{V|t}(\cdot) \tag{4.26}$$

for $\tau = t,\dots,T$.

We use these to apply the machinery of § 4.4.3, solve the Bellman equations and obtain the suboptimal SHDP policy at time t. We compute the whole sequence of cost-to-go functions and policies at times $\tau = t,\dots,T$. The cost-to-go functions are

$$v_{\tau|t}(x_\tau) = x_\tau^T D_{\tau|t} x_\tau + d_{\tau|t} x_\tau + b_{\tau|t} \tag{4.27}$$

for $\tau = t,\dots,T-1$. The only difference with equation (4.20) is the condition $|t$ in the subscript, because expected values are taken over the marginal conditional densities $\hat{f}_{w_\tau|t}(\cdot)$. Similarly, the policies are

$$\mu_{\tau|t}(x_\tau) = K_{\tau|t} x_\tau + l_{\tau|t} \tag{4.28}$$

for $\tau = t, \ldots, T-1$. We report the equations for this recursion in Appendix C.2. At every time step t we compute the whole sequence of cost-to-go and policies, in order to get the optimal action

$$u_t^\star = \mu_{t|t}(x_t) = K_{\tau|t} x_\tau + l_{\tau|t}. \tag{4.29}$$

We then move to the next time step and repeat the whole process. If we are not interested in computing the cost-to-go $v_{t|t}(x_t)$ the equations simplify somewhat (we disregard large part of the recursion and only compute what we need). We develop these simplified formulas in Appendix C.3.

4.5 Empirical results

We study the performance of the *static solution* of § 4.3 versus the *dynamic solution* of § 4.4 by simulating execution or stock orders, using real NYSE market price and volume data. We describe in § 4.5.1 the dataset and how we process it. The dynamic solution requires a model for the joint distribution of market volumes, here we use a simple model, explained in § 4.5.2. (We expect that a more sophisticated model for market volumes would improve the solution performance significantly.) In § 4.5.3 we describe the "rolling testing" framework in which we operate. Our procedure is made up of two parts: the historical estimation of model parameters, explained in § 4.5.4, and the actual simulation of order execution, in § 4.5.5. Finally in § 4.5.6 we show our aggregate results.

4.5.1 Data

We simulate execution on data from the NYSE stock market. Specifically, we use the $K = 30$ different stocks which make up the Dow Jones Industrial Average (DJIA), on $N = 60$ market days corresponding to the last quarter of 2012, from September 24 to December 20 (we do not consider the last days of December because the market was either closed or had reduced trading hours). The 30 symbols in that quarter are: MMM, AXP, T, BA, CAT, CVX, CSCO, KO, DD, XOM, GE, HD, INTC, IBM, JNJ, JPM, MCD, MRK, MSFT, PFE, PG, TRV, UNH, UTX, VZ, WMT, DIS, AA, BAC, HPQ. We use raw Trade and Quotes (TAQ) data from Wharton Research Data Services (WRDS) [1]. We processe the raw data to obtain daily series of market volumes $m_t \in \mathbf{Z}_+$ and average period price $p \in \mathbf{R}_{++}$, for $t = 1, \ldots, T$ where $T = 390$, so that each interval is one minute long. We clean the raw data by filtering out trades meeting any of the following conditions:

- *correction code* greater than 1, trade data incorrect;
- *sales condition* "4", "@4", "C4", "N4", "R4", *derivatively priced*, *i.e.*, the trade was executed over-the-counter (or in an external facility like a Dark Pool);
- *sales condition* "T" or "U", *extended hours* trades (before or after the official market hours);
- *sales condition* "V", *stock option* trades (which are also executed over-the-counter);
- *sales condition* "Q", "O", "M", "6", *opening trades* and *closing trades* (the opening and closing auctions).

In other words we focus exclusively on the continuous trading activity without considering market opening and closing nor any over-the-counter trade. In Figure 4.1 we plot an example of market volumes and prices.

4.5.2 Market volumes model

We have so far assumed that the distribution of market volumes

$$f_m(m_1, \ldots, m_T)$$

is known from the start of the day. In reality a broker has a parametric family of distributions and each day (or less often) selects the parameters for the distribution with some statistical procedure. For simplicity, we assume such procedure is based on historical data. We found few works in the literature concerned with intraday

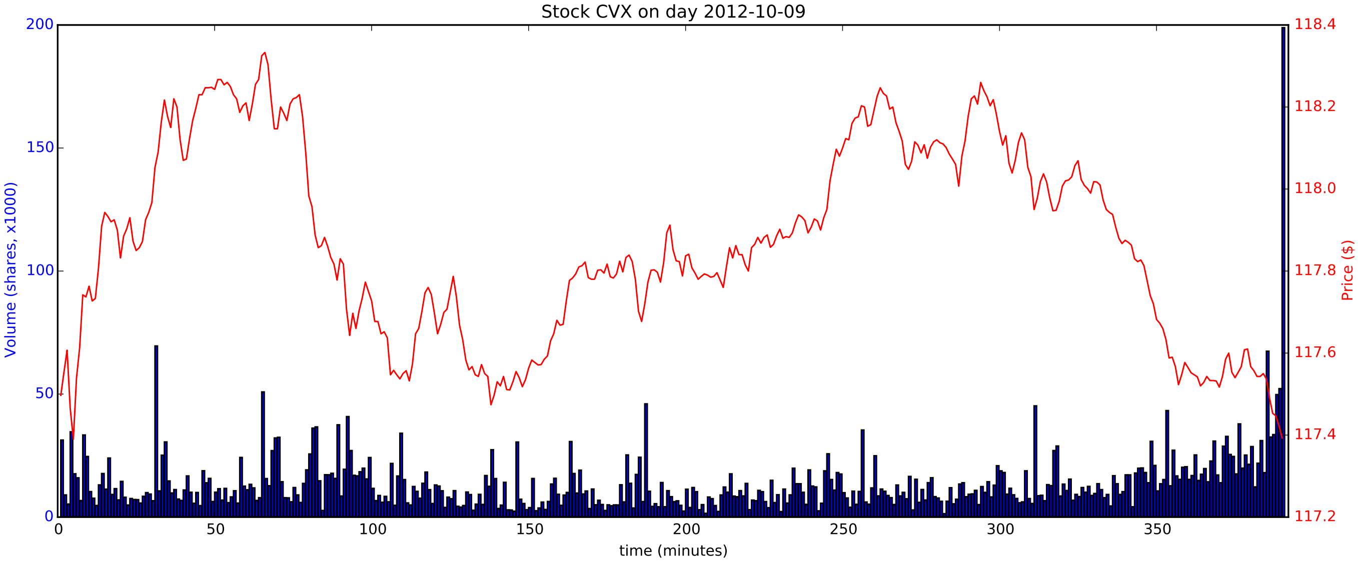

Figure 4.1: Example of a trading day. The blue bars are the market volumes traded every minute (in number of shares) and the red line is the period price p_t.

market volumes modeling ([14]). We thus develop our own market volume model. This is composed of a parametric family of market volume distributions and an *ad hoc* procedure to choose the parameters with historical data.

For each stock we model the vector of market volumes as a multivariate log-normal. If the superscript (k) refers to the stock k (*i.e.*, $m_t^{(k)}$ is the market volume for stock k in interval t), we have

$$f_{m^{(k)}}(m_1^{(k)}, \dots, m_T^{(k)}) \sim \ln \mathcal{N}(\mu + \mathbf{1} b^{(k)}, \Sigma) \tag{4.30}$$

where $b^{(k)} \in \mathbf{R}$ is a constant that depends on the stock k (each stock has a different typical daily volume), $\mu \in \mathbf{R}^T$ is an average "volume profile" (normalized so that $\mathbf{1}^T \mu = 0$) and $\Sigma \in \mathbf{S}_{++}^T$ is a covariance matrix. The volume process thus separates into a per-stock deterministic component, modeled by the constant $b^{(k)}$, and a stochastic component with the same distribution for all stocks, modeled as a multivariate log-normal. We report in Appendix D the *ad hoc* procedure we use to estimate the parameters of this volume model on historical data and the formulas for the conditional expectations $\mathbf{E}_t\left[1/V\right]$, $\mathbf{E}_t[m_\tau]$, $\mathbf{E}_t\left[1/m_\tau\right]$ for $\tau = t, \ldots, T$ (which we need for the solution (4.29)). The procedure for estimating the volume model on past data requires us to provide a parameter, which we estimate with cross-validation on the initial section of the data. The details are explained in Appendix D.

4.5.3 Rolling testing

We organize our simulations according to a “rolling testing” or “moving window” procedure: for every day used to simulate order execution we estimate the various parameters on data from a “window” covering the preceding $W > 0$ days. (It is commonly assumed that the most recent historical data are most relevant for model calibration since the systems underlying the observed phenomena change over time). We thus simulate execution on each day $i = W + 1, \ldots, N$ using data from the days $i - W, \ldots, i - 1$ for historical estimation.

In this way every time we test a VWAP solution algorithm, we use model parameters calibrated on historical data exclusively. In other words the performance of our models are estimated *out-of-sample.* In addition since all the order simulations use the same amount of historical data for calibration it is fair to compare them.

We fix the window lenght of the historical estimation to $W = 20$, corresponding roughly to one month. We set aside the first $W_{CV} = 10$ simulation days for cross-validating a feature of the volume model, as explained in Appendix D.2. In Figure 4.2 we describe the procedure. We iterate over the dataset, simulating execution on any day $i = W + 1, \ldots, N$ and estimating the model parameters on the preceding $W = 20$ days. The first $W_{CV} = 10$ days used to simulate orders are reserved for cross validation (as explained in Appendix D.2). The aggregate results from the remaining $W + W_{CV} + 1, \ldots, N$ days (30 days in total) are presented in § 4.5.6. In the next two sections we explain how we perform the estimation of model parameters and simulation of orders execution.

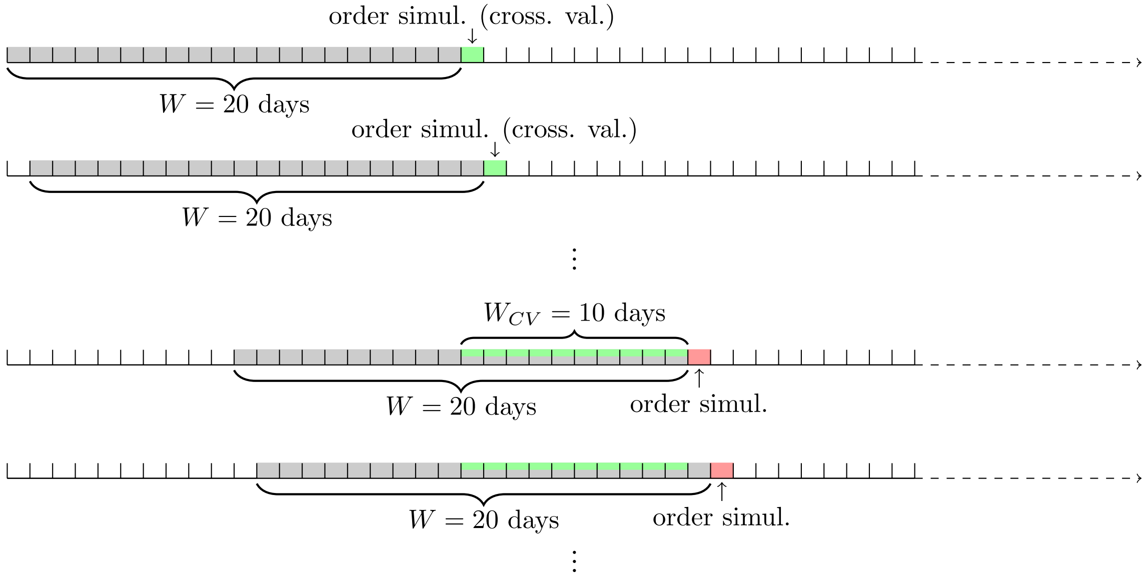

Figure 4.2: Description of the rolling testing procedure.

4.5.4 Models estimation

We describe the estimation, on historical data, of the parameters of all relevant models for our solution algorithms. We append the superscript (i, k) to any quantity that refers to market day i and stock k. We start by the market volumes per interval as a fraction of the total daily volume (which we need for (4.16)). We use the sample average

$$\mathbf{E}\left[\frac{m_t}{V}\right] \simeq \frac{\sum_{j=i-W}^{i-1} \sum_{k=1}^{K} m_t^{(j,k)} / V^{(j,k)}}{WK}$$

for every $t = 1, \ldots, T$. An example of this estimation (on the first $W = 20$ days of the dataset) is shown in Figure 4.3. The dynamic solution (4.29) requires an estimate of the volatilites σ_t, we use the sample average of the squared price changes

$$\sigma_t^2 \simeq \frac{\sum_{j=i-W}^{i-1} \sum_{k=1}^{K} \left((p_{t+1}^{(j,k)} - p_t^{(j,k)}) / p_t^{(j,k)} \right)^2}{WK}$$

for every $t = 1, \ldots, T$. In Figure 4.4 we show an example of this estimation (on the first W days of the dataset). For each period of one minute these are the estimated

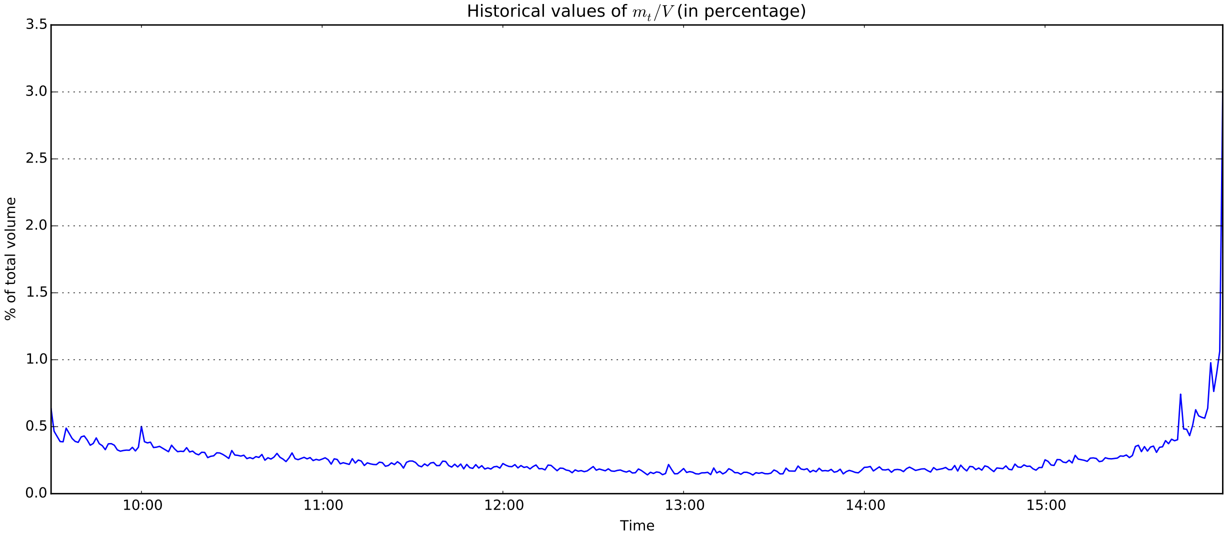

Figure 4.3: Estimated values of $\mathbf{E}\left[\frac{m_t}{V}\right]$ using the first $W = 20$ days of our dataset, shown in percentage points.

standard deviation of the price increments, shown in basis points (one basis point is 0.0001). We then choose the volume distribution $f_m(m_1, \ldots, m_T)$ among the parametric family defined in § 4.5.2 (using the *ad hoc* procedure described in Appendix D.1). We estimate the expected daily volume for each stock as the sample average

$$\mathbf{E}[V^{(i,k)}] \simeq \frac{\sum_{j=i-W}^{i-1} V^{(j,k)}}{W}$$

for every $k = 1, \ldots, K$. We use this to choose the size of the simulated orders.

Finally, we consider the parameters $s_1, \ldots, s_T$, and α of the transaction cost model (4.2). We do not estimate them empirically since we would need additional data, market quotes for the spread and proprietary data of executed orders for α (confidential for fiduciary reasons). We instead set them to exogenous values, kept constant across all stocks and days (to simplify comparison of execution costs). We assume for simplicity that the fractional spread is constant in time and equal to 2 basis points, $s_1 = \cdots = s_T = 2$ b.p. (one basis point is 0.0001). That is reasonable for liquid

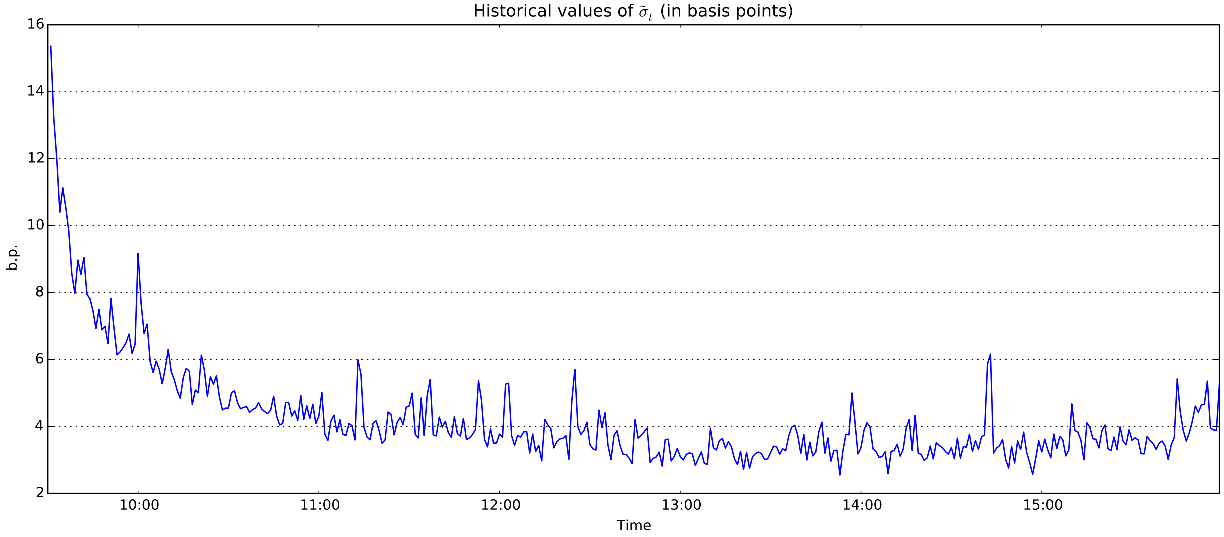

Figure 4.4: Estimated values of the period volatilities, $\hat{\sigma}_t$ using the first $W = 20$ days of our dataset.

stocks such as the ones from the DJIA. We choose the parameter α following a rule-of-thumb of transaction costs: trading one day's volume costs approximately on day's volatility [71]. We estimate empirically over the first 20 days of the dataset the open-to-close volatility for our stocks, equal to approximately 90 basis points, and thus from equation (4.2) we set $\alpha = 90$.

4.5.5 Simulation of execution with VWAP solution algorithms

For each day $i = W+1, \ldots, N$ and each stock $k = 1, \ldots, K$ we simulate the execution of a trading order. We fix the size of the order equal to 1% of the expected daily volume for the given stock on the given day

$$C^{(i,k)} = \mathbf{E}[V^{(i,k)}]/100.$$

Such orders are small enough to have negligible impact on the price of the stock [16], as we need for (4.2) to hold.

We repeat the simulation with different solution methods: the static solution (4.16) and the dynamic solution (4.29) with risk-aversion parameters $\lambda = 0, 1, 10, 100, 1000, \infty$. We use the symbol a to index the solution methods. For each simulation we solve the appropriate set of equations, setting all historically estimated parameters to the values obtained with the procedures of § 4.5.4. For each solution method we obtain a simulated trading schedule

$$u_t^{(i,k,a)}, \quad t = 1, \ldots, T$$

where the superscript a indexes the solution methods. We then compute the slippage incurred by the schedule using (4.4)

$$S^{(i,k,a)} = \frac{\sum_{t=1}^{T} p_t^{(i,k)} u_t^{(i,k,a)} - C^{(i,k)} p_{\text{VWAP}}^{(i,k)}}{C^{(i,k)} p_{\text{VWAP}}^{(i,k)}} + \sum_{t=1}^{T} \frac{s_t}{2} \left(\alpha \frac{\left(u_t^{(i,k,a)}\right)^2}{C^{(i,k)} m_t^{(i,k)}} - \frac{u_t^{(i,k,a)}}{C^{(i,k)}} \right). \quad (4.31)$$

Note that we are simulating the transaction costs. Measuring them directly would require to actually execute $u_t^{(i,k,a)}$. This test of transaction costs optimization has value as a comparison between the static solution (4.16) and the dynamic solution (4.29). Our transaction costs model (4.2) is similar to the ones of other works in the literature (*e.g.*, [51]) but involves the market volumes m_t. The static solution only uses the market volumes distribution known before the market opens, while the dynamic solution uses the SHDP procedure to incorporate real time information and improve modeling of market volumes. In the following we show that the dynamic solution achieves lower transaction costs than the static solution, such gains are due to the better handlng of information on market volumes.

In practice a broker would use a different model of transaction costs, perhaps more complicated than ours. We think that a good model should incorporate the market volumes m_t as a key variable [16]. Our test thus suggests that also in that setting the dynamic solution would perform better than the static solution.

We show in Figure 4.5 the result of the simulation on a sample market day, using the static solution (4.16) and the dynamic solution (4.29) for $\lambda = 0$ and ∞. We also plot the market volumes $m_t^{(i,k)}$. We report all volume processes as cumulative fraction of their total. At every time τ we plot $\sum_{t=1}^{\tau} \frac{m_t}{V}$ for the market volumes m_t and $\sum_{t=1}^{\tau} \frac{u_t}{C}$ for the various solutions u_t. We only show the dynamic solution for $\lambda = 0$ and $\lambda = \infty$ since for all other values of λ the solution falls in between.

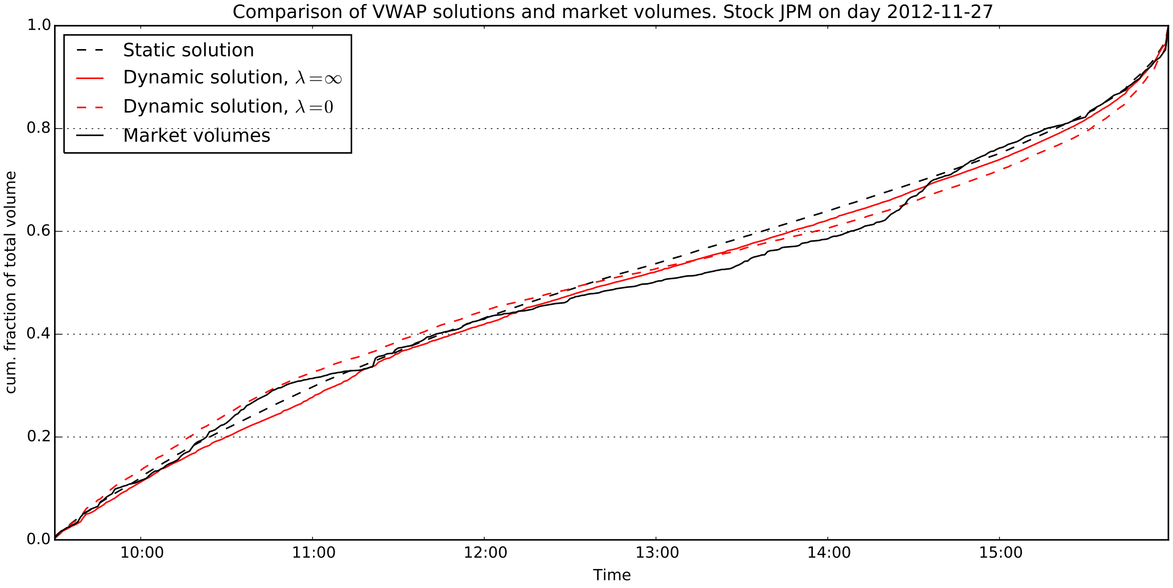

Figure 4.5: Simulation of order execution on a sample market day.

4.5.6 Aggregate results

We report the aggregate results from the simulation of VWAP execution on all the days reserved for orders simulation (minus the ones used for cross-validation). For any day $i = W + W_{CV} + 1, \ldots, N$, stock $k = 1, \ldots, K$, and solution method a (either the static solution (4.16) or the dynamic solution (4.29) for various values of λ) we obtain the simulated slippage $S^{(i,k,a)}$ using (4.31). Then, for each solution method a we define the empirical expected value of S as

$$\mathbf{E}[S^{(a)}] = \frac{\sum_{i=W+W_{CV}+1}^{N} \sum_{k=1}^{K} S^{(i,k,a)}}{(N - W - W_{CV})K}$$

and the empirical variance

$$\mathbf{var}(S^{(a)}) = \frac{\sum_{i=W+W_{CV}+1}^{N} \sum_{k=1}^{K} \left(S^{(i,k,a)}\right)^2 - \mathbf{E}[S^{(a)}]^2}{(N-W-W_{CV})K-1}.$$

In Figure 4.6 we show the values of these on a risk-reward plot. (We show the square root of the variance for simplicity, so that both axes are expressed in basis points). Each dot represents one solution method, either the static solution (4.16) or the *dynamic solution* (4.29) with risk-aversion parameters $\lambda = 0, 1, 10, 100, 1000, \infty$. We show the sample average of the simulated slippages, which represents the execution costs, and the sample standard deviation, *i.e.*, the root mean square error (RMSE) of tracking the VWAP. The orders have size equal to 1% of the expected daily volume. The dynamic solution improves over the static solution in both dimensions, we can choose the preferred behaviour by fixing the risk-aversion parameter λ. We observe that the dynamic solution improves over the static solution on both VWAP tracking (variance of S) and transaction costs (expected value of S), and we can select between the different behaviors by choosing different values of λ.

We introduced in § 4.2.2 the approximation that the value of (4.10) is negligible when compared to (4.9). The empirical results validate this. For the static solution the empirical average value of (4.9) is $4.45e-07$ while (4.10) is $6.34e-09$, about 1%. For the dynamic solution with $\lambda = \infty$ the average value of (4.10) is $3.60e-07$ and (4.9) is $1.92e-08$, about 5%. For the dynamic solution with $\lambda = 0$ instead the average value of (4.10) is $4.76e-07$ and (4.9) is $5.50e-09$, about 1%. The dynamic solutions for other values of λ sit in between. Thus the approximation is generally valid, becoming less tight for high values of λ. In fact in Figure 4.6 we see that the empirical variance of S for the dynamic solution with $\lambda = \infty$ is somewhat larger than the one with $\lambda = 1000$, probably because of the contribution of (4.9). (We can interpret this as a *bias-variance* tradeoff since by going from $\lambda = \infty$ to $\lambda = 1000$ we effectively introduce a regularization of the solution.)

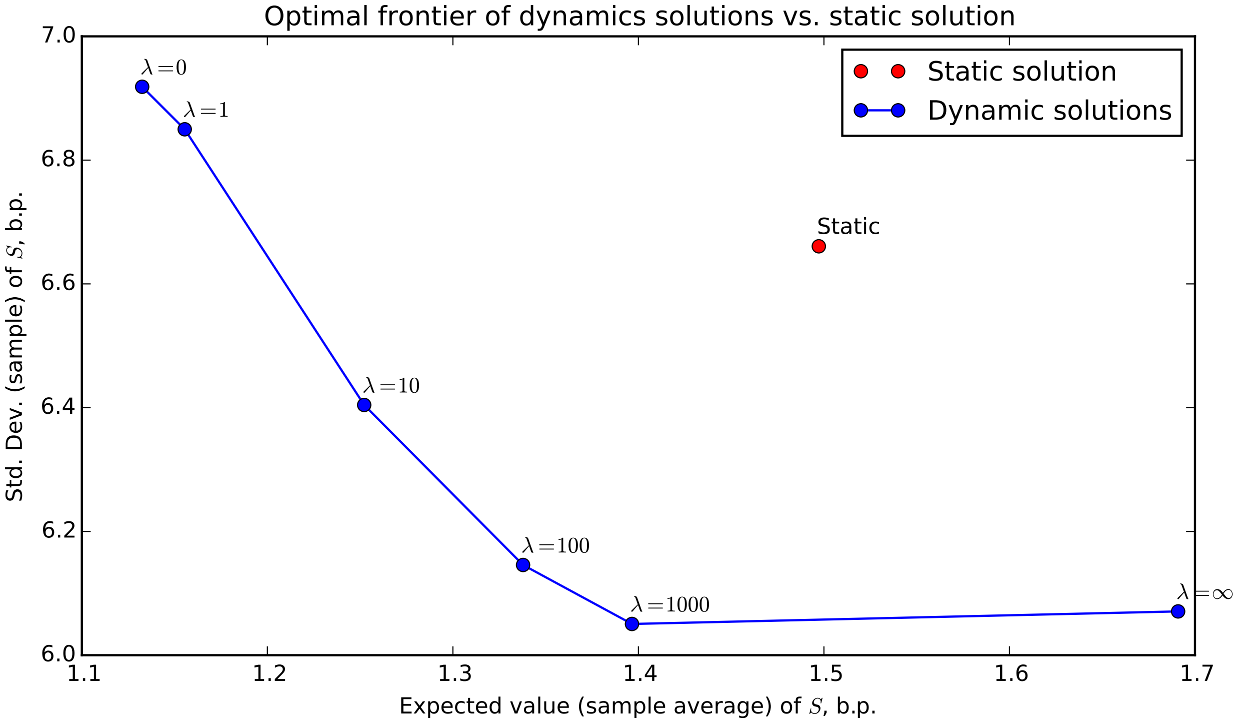

Figure 4.6: Risk-reward plot of the aggregate results of our simulations on real market data.

4.6 Conclusions

We studied the problem of optimal execution under VWAP benchmark and developed two broad families of solutions.

The static solution of § 4.3, although derived with similar assumptions as the classic [73], is more flexible and accommodates more sophisticated models of bid-ask spread and volume than the comparable approaches in the literature. By formulating the problem as a quadratic program it is easy to add other convex constraints (see [94] for a comprehensive list) with a guaranteed straightforward and fast solution [21].

The dynamic solution of § 4.4 is the biggest contribution of this work. One one side we manipulate the problem to fit it into the standard formalism of linear-quadratic stochastic control. On the other we model the uncertainty on the total market volume (which is eschewed in all similar works we found in the literature) in a principled way, building on a recent result in optimal control [116].

In § 4.5 we design and run a simulation or orders execution on real data. We use sound statistical practices, the rolling testing of § 4.5.3 ensures that all results are obtained out-of-sample. We compare the performance of our dynamic solution over the simple static solution which is standard in the trading industry. Our dynamic solution is built around a model for the joint distribution of market volumes, we provide a simple one in § 4.5.2 (along with *ad hoc* procedures to use it). This is supposed to be a proof-of-concept, in practice a broker would have a more robust market volume model (which could further improve performance of the dynamic solution). Even with this simple market volume model the simulation shows that our dynamic solution has significantly better performance than the benchmark (Figure 4.6). On one side we can reduce the RMSE of VWAP tracking by 10%. On the other we can lower the execution costs by around 25%. We can as well reduce both performance metrics over the benchmark. This result validates the approximations involved in the derivation of the dynamic solution and thus shows its value.

Appendix A

Miscellaneous lemmas

Here we collect some technical details and derivations of results from the text.

Lemma 1. *The bet e_n is optimal for problem* (3.1) *if and only if* $\mathbf{E}\, r_i \leq 1$ *for* $i = 1, \ldots, n-1$. *Likewise, the bet e_n is optimal for problem* (3.7) *if and only if* $\mathbf{E}\, r_i \leq 1$ *for* $i = 1, \ldots, n-1$.

Proof. Assume $\mathbf{E}\, r_i \leq 1$ for $i = 1, \ldots, n-1$. Then by Jensen's inequality,

$$\mathbf{E} \log(r^T b) \leq \log \mathbf{E}\, r^T b \leq 0$$

for any $b \in \{b \mid \mathbf{1}^T b = 1,\ b \geq 0\}$. So e_n, which achieves objective value 0, is optimal for both problem (3.1) and problem (3.7).

Next assume $\mathbf{E}\, r_i > 1$ for some i. Consider the bet $b = f e_i + (1-f) e_n$ and

$$\phi(f) = \mathbf{E} \log(r^T b)$$

for $f \in [0, 1)$. By [10, Proposition 2.1] we have the right-sided derivative

$$\partial_+ \phi(f) = \mathbf{E}\, \frac{r(e_i - e_n)}{r^T b},$$

and we have

$$\partial_+ \phi(0) = \mathbf{E}\, r(e_i - e_n) = \mathbf{E}\, r_i - 1 > 0.$$

Since $\phi(0) = 0$, we have $\phi(f) > 0$ for a small enough f. So $b = fe_i + (1-f)e_n$ has better objective value than e_n for small enough f, *i.e.*, e_n is not optimal for problem (3.1).

Again, assume $\mathbf{E}\, r_i > 1$ for some i. Write $b^\star$ for the Kelly optimal bet. (We have already established that $b^\star \neq e_n$ and that $\mathbf{E}\log(r^T b^\star) > 0$.) Consider the bet $\tilde{b} = fb^\star + (1-f)e_n$ and

$$\psi(f) = \mathbf{E}(r^T\tilde{b})^{-\lambda}$$

for $f \in [0,1)$. By a similar argument as before, we have

$$\partial_+\psi(f) = -\lambda\, \mathbf{E}(r^T\tilde{b})^{-\lambda-1} r^T(b^\star - e_n),$$

and

$$\partial_+\psi(0) = -\lambda\left(\mathbf{E}(r^T b^\star) - 1\right).$$

By Jensen's inequality we have

$$0 \leq \mathbf{E}\log(r^T b^\star) \leq \log \mathbf{E}\, r^T b^\star.$$

So $\partial_+\psi(0) < 0$. Since $\psi(0) = 1$, we have $\psi(f) < 1$ for small enough f. So $\tilde{b} = fb^\star + (1-f)e_n$ is feasible for small enough f. Furthermore, $\tilde{b}$ has strictly better objective value than e_n for $f \in (0,1)$ since the objective is concave. So e_n is not optimal. □

Lemma 2. *Problem* (3.7) *always has an optimal dual variable.*

Proof. Assume e_n is optimal. By Lemma 1, e_n is optimal even without the constraint $\mathbf{E}(r^T b)^{-\lambda} \leq 1$. So $\kappa^\star = 0$ is an optimal dual variable.

Now assume e_n is not optimal. By the reasoning with the function ψ of Lemma 1's proof, there is a point strictly feasible with respect to the constraint $\mathbf{E}(r^T b)^{-\lambda} \leq 1$. So by [110, Theorem 17], we conclude a dual solution exists. □

Lemma 3. *Let* $\varepsilon \in [0,1]$*, and write* $\Pi(\cdot)$ *for the projection onto* Δ_ε*. Define the function* $(\cdot)_{+,\varepsilon} : \mathbf{R}^n \to \mathbf{R}^n$ *as* $((x)_{+,\varepsilon})_i = \max\{x_i, 0\}$ *for* $i = 1, \ldots, n-1$ *and*

$((x)_{+,\varepsilon})_n = \max\{x_n, \varepsilon\}$. *Then*

$$\Pi(z) = (z - \nu\mathbf{1})_{+,\varepsilon},$$

where ν *is a solution of the equation*

$$\mathbf{1}^T(z - \nu\mathbf{1})_{+,\varepsilon} = 1.$$

The left-hand side is a nonincreasing function of ν*, so* ν *can be obtained efficiently via bisection on with the starting interval* $[\max_i z_i - 1, \max_i z_i]$.

Proof. By definition of the projection, $x = \Pi(z)$ is the solution of

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|x - z\|_2^2 \\ \text{subject to} & \mathbf{1}^T x = 1 \\ & x \geq 0, \quad x_n \geq \varepsilon. \end{array}$$

This problem is equivalent to

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|x - z\|_2^2 + \nu(\mathbf{1}^T x - 1) \\ \text{subject to} & x \geq 0, \quad x_n \geq \varepsilon \end{array}$$

for an optimal dual variable $\nu \in \mathbf{R}$. This follows from dualizing with respect to the constraint $\mathbf{1}^T x = 1$ (and not the others), applying strong Lagrange duality, and using fact that the objective is strictly convex [12]. This problem is in turn equivalent to

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|x - (z - \nu\mathbf{1})\|_2^2 \\ \text{subject to} & x \geq 0, \quad x_n \geq \varepsilon, \end{array}$$

which has the analytic solution

$$x^\star = (z - \nu\mathbf{1})_{+,\varepsilon}.$$

An optimal dual variable ν must satisfy

$$\mathbf{1}^T(z-\nu\mathbf{1})_{+,\varepsilon}=1,$$

by the KKT conditions. Write $h(\nu)=(z-\nu\mathbf{1})_{+,\varepsilon}$. Then $h(\nu)$ a continuous nonincreasing function with

$$h(\max_i z_i-1)\geq 1, \quad h(\max_i z_i)=\varepsilon.$$

So a solution of $h(\nu)=1$ is in the interval $[\max_i z_i-1, \max_i z_i]$.

□

Lemma 4. *A pair $(b^\star,\kappa^\star)$ is a solution of the RCK problem if and only if it satisfies conditions* (3.13)*.*

Proof. Assume $(b^\star,\kappa^\star)$ is a solution of the RCK problem. This immediately gives us $\mathbf{1}^Tb^\star=1$, $b^\star\geq 0$, $\mathbf{E}(r^Tb^\star)^{-\lambda}\leq 1$, and $\kappa^\star\geq 0$. Moreover $b^\star$ is a solution to

$$\begin{array}{ll}\text{minimize} & -\mathbf{E}\log(r^Tb)+\kappa^\star\,\mathbf{E}(r^Tb)^{-\lambda}\\ \text{subject to} & \mathbf{1}^Tb=1, \quad b\geq 0\end{array} \tag{A.1}$$

and $\kappa^\star(\mathbf{E}(r^Tb^\star)^{-\lambda}-1)=0$, by Lagrange duality.

Consider $b^\varepsilon=(1-\varepsilon)b^\star+\varepsilon b$, where $b\in\mathbf{R}^n$ satisfies $\mathbf{1}^Tb=1$ and $b\geq 0$. Define $\varphi(\varepsilon)=-\mathbf{E}\log(r^Tb^\varepsilon)+\kappa^\star\,\mathbf{E}(r^Tb^\varepsilon)^{-\lambda}$. Since b^ε is feasible for problem (A.1) for $\varepsilon\in[0,1)$ and since $b^\star$ is optimal, we have $\varphi(0)\leq\varphi(\varepsilon)$ for $\varepsilon\in[0,1)$. We later show $\varphi(\varepsilon)<\infty$ for $\varepsilon\in[0,1)$. So $\partial_+\varphi(0)\geq 0$, where ∂_+ denotes the right-sided derivative.

By [10, Proposition 2.1] we have

$$\partial_+\varphi(\varepsilon)=-\mathbf{E}\,\frac{r^T(b-b^\star)}{r^Tb^\varepsilon}-\kappa^\star\lambda\,\mathbf{E}\,\frac{r^T(b-b^\star)}{(r^Tb^\varepsilon)^{\lambda+1}}.$$

So we have

$$0\leq\partial_+\varphi(0)=1+\kappa^\star\lambda\,\mathbf{E}(r^Tb^\star)^{-\lambda}-\mathbf{E}\,\frac{r^Tb}{r^Tb^\star}-\kappa^\star\lambda\,\mathbf{E}\,\frac{r^Tb}{(r^Tb^\star)^{\lambda+1}}.$$

Using $\kappa^\star(\mathbf{E}(r^Tb^\star)^{-\lambda} - 1) = 0$, and reorganizing we get

$$\left(\mathbf{E}\,\frac{r}{r^Tb^\star} + \kappa^\star\lambda\,\mathbf{E}\,\frac{r}{(r^Tb^\star)^{\lambda+1}}\right)^T b \le 1 + \kappa^\star\lambda, \tag{A.2}$$

for any $b \in \mathbf{R}^n$ that satisfies $\mathbf{1}^Tb = 1$ and $b \ge 0$. With $b = e_i$ for $i = 1, \dots, n$, we get

$$\mathbf{E}\,\frac{r_i}{r^Tb^\star} + \kappa^\star\lambda\,\mathbf{E}\,\frac{r_i}{(r^Tb^\star)^{\lambda+1}} \le 1 + \kappa^\star\lambda, \tag{A.3}$$

for $i = 1, \dots, n$.

Now assume $b_i^\star > 0$ for some i and let $b = e_i$. Then b^ε is feasible for problem (A.1) for small negative ε. We later show $\varphi(\varepsilon) < \infty$ for small negative ε. This means $\partial_+\varphi(0) = 0$ in this case, and we conclude

$$\mathbf{E}\,\frac{r_i}{r^Tb^\star} + \kappa^\star\lambda\,\mathbf{E}\,\frac{r_i}{(r^Tb^\star)^{\lambda+1}} \begin{cases} \le 1 + \kappa^\star\lambda & b_i = 0 \\ = 1 + \kappa^\star\lambda & b_i > 0. \end{cases}$$

It remains to show that $\varphi(\varepsilon) < \infty$ for appropriate values of ε. First note that $b^\star$, a solution, has finite objective value, *i.e.*, $-\mathbf{E}\log(r^Tb^\star) < \infty$ and $\mathbf{E}(r^Tb^\star)^{-\lambda} < \infty$. So for $\varepsilon \in [0, 1)$, we have

$$\begin{aligned} \varphi(\varepsilon) &= -\mathbf{E}\log((1-\varepsilon)r^Tb^\star + \varepsilon r^Tb) + \kappa^\star\,\mathbf{E}((1-\varepsilon)r^Tb^\star + \varepsilon r^Tb)^{-\lambda} \\ &\le -\mathbf{E}\log((1-\varepsilon)r^Tb^\star) + \kappa^\star\,\mathbf{E}((1-\varepsilon)r^Tb^\star)^{-\lambda} \\ &= -\log(1-\varepsilon) - \mathbf{E}\log(r^Tb^\star) + \kappa^\star(1-\varepsilon)^{-\lambda}\,\mathbf{E}(r^Tb^\star)^{-\lambda} < \infty. \end{aligned}$$

Next assume $b_i > 0$ for some $i \in \{1, \dots, n\}$. Then for $\varepsilon \in (-b_i, 0)$ we have

$$\begin{aligned} \varphi(\varepsilon) &\le -\mathbf{E}\log(r^Tb^\star + \varepsilon r_i) + \kappa^\star\,\mathbf{E}(r^Tb^\star + \varepsilon r_i)^{-\lambda} \\ &\le -\mathbf{E}\log(((b_i+\varepsilon)/b_i)r^Tb^\star) + \kappa^\star\,\mathbf{E}(((b_i+\varepsilon)/b_i)r^Tb^\star)^{-\lambda} \\ &= -\log((b_i+\varepsilon)/b_i) - \mathbf{E}\log(r^Tb^\star) + \kappa^\star((b_i+\varepsilon)/b_i)^{-\lambda}\,\mathbf{E}(r^Tb^\star)^{-\lambda} < \infty. \end{aligned}$$

Finally, assume conditions (3.13) for $(b^\star, \kappa^\star)$, and let us go through the argument in reverse order to show the converse. Conditions (3.13) implies condition (A.3), which in turn implies condition (A.2) as $\mathbf{1}^T b = 1$. Note $b^\star$ has finite objective value becuase $\mathbf{E}(r^T b^\star)^{-\lambda} < \infty$ by assumption and $-\mathbf{E}\log(r^T b^\star) \leq (1/\lambda)\log\mathbf{E}(r^T b^\star)^{-\lambda} < \infty$ by Jensen's inequality. So $\varphi(0) \leq \varphi(\varepsilon)$ for $\varepsilon \in [0,1)$ by the same argument as before, *i.e.*, $b^\star$ is optimal for problem (A.1). This fact, together with conditions (3.13), give us the KKT conditions of the RCK problem, and we conclude $(b^\star, \kappa^\star)$ is a solution of the RCK problem by Lagrange duality [12]. □

Lemma 5. *Consider an IID sequence $X_1, X_2, \ldots$ from probability measure μ, its random walk $S_n = X_1 + X_2 + \cdots + X_n$, a stopping time τ, and*

$$\psi(\lambda) = \log\mathbf{E}\exp(-\lambda X) = \log\int\exp(-\lambda x)\, d\mu(x).$$

Then we have

$$\mathbf{E}\left[\exp(-\lambda S_\tau - \tau\psi(\lambda)) \mid \tau < \infty\right]\mathbf{Prob}(\tau < \infty) \leq 1.$$

This lemma is a modification of an identity from [119], [49, §XVIII.2], and [54, §9.4].

Proof. Consider the tilted probability measure

$$d\mu_\lambda(x) = \exp(-\lambda x - \psi(\lambda))d\mu(x)$$

and write $\mathbf{Prob}_{\mu_\lambda}$ for the probability under the tilted measure μ_λ. Then we have

$$\begin{aligned}\underset{\mu_\lambda}{\mathbf{Prob}}(\tau = n) &= \int I_{\{\tau=n\}} d\mu_\lambda(x_1, x_2, \ldots, x_n)\\ &= \int I_{\{\tau=n\}}\exp(-\lambda s_n - n\psi(\lambda))d\mu(x_1, x_2, \ldots, x_n)\\ &= \mathbf{E}\left[\exp(-\lambda S_\tau - \tau\psi(\lambda))I_{\{\tau=n\}}\right]\end{aligned}$$

By summing through $n = 1, 2, \ldots$ we get

$$\begin{aligned}\underset{\mu_\lambda}{\mathbf{Prob}}(\tau < \infty) &= \mathbf{E}\left[\exp(-\lambda S_\tau - \tau\psi(\lambda)) I_{\{\tau<\infty\}}\right] \\ &= \mathbf{E}\left[\exp(-\lambda S_\tau - \tau\psi(\lambda)) \mid \tau < \infty\right] \mathbf{Prob}(\tau < \infty).\end{aligned}$$

Since $\mathbf{Prob}_{\mu_\lambda}(\tau < \infty) \leq 1$, we have the desired result. $\square$

Appendix B

DCP specification

The finite outcome RCK problem (3.11) can be formulated and solved in CVXPY as

```
b = Variable(n)
lambda_risk = Parameter(sign = ’positive’)
growth = pi.T*log(r.T*b)
risk_constraint = (log_sum_exp (log(pi) - lambda_risk * log(r.T*b)) <= 0)
constraints = [ sum_entries(b) == 1, b >= 0, risk_constraint ]
risk_constr_kelly = Problem(Maximize(growth),constraints)
risk_constr_kelly.solve()
```

Here `r` is the matrix whose columns are the return vectors, and `pi` is the vector of probabilities. The second to last line forms the problem (object), and in the last line the problem is solved. The optimal bet is written into `b.value`. We note that if we set `lambda_risk` equal to 0 this problem formulation is equivalent (computationally) to the Kelly problem.

Appendix C

Dynamic programming equations

C.1 Riccati equations for LQSC

We derive the recursive formulas for (4.19) and (4.20). We know the final condition

$$v_{T+1}(x_{T+1}) = g_{T+1}(x_{T+1}),$$

so $D_{T+1} = Q_{T+1}$, $d_{T+1} = d_{T+1}$, and $b_{T+1} = 0$. Now for the inductive step, assume $v_{t+1}(x_{t+1})$ is in the form of (4.20) with known D_{t+1}, d_{t+1}, and b_{t+1}. Then the optimal action at time t is, according to (4.18),

$$u_t = \underset{u}{\operatorname{argmin}}\, \mathbf{E}[g_t(x_t, u, w_t) + v_{t+1}(A_t(w_t)x_t + B_t(w_t)u + c_t(w_t))] = K_t x_t + l_t,$$

with

$$\begin{aligned} K_t &= -\frac{\mathbf{E}[B_t(w_t)^T D_{t+1} A_t(w_t)]}{(\mathbf{E}\, R_t(w_t) + \mathbf{E}[B_t(w_t)^T D_{t+1} B_t(w_t)]} \\ l_t &= -\frac{\mathbf{E}\, r_t + 2\,\mathbf{E}[B_t(w_t)^T D_{t+1} c(w_t)] + d_{t+1}^T\, \mathbf{E}\, B_t(w_t)}{2(\mathbf{E}\, R_t(w_t) + \mathbf{E}[B_t(w_t)^T D_{t+1} B_t(w_t)]}. \end{aligned}$$

It follows that the value function at time t is also in the form of (4.20), and it has value

$$v_t(x_t) = \mathbf{E}\left[g_t(x_t, K_t x_t + l_t, w_t) + v_{t+1}(A_t(w_t)x_t + B_t(w_t)(K_t x_t + l_t) + c_t(w_t))\right] = \\ x_t^T D_t x_t + d_t^T x_t + b_t$$

with

$$\begin{aligned}
D_t &= \mathbf{E}\, Q_t(w_t) + K_t^T\, \mathbf{E}\left[R_t(w_t) + B_t(w_t)^T D_{t+1} B_t(w_t)\right] K_t + \\
&\quad \mathbf{E}[A_t(w_t)^T D_{t+1} A_t(w_t)] + K_t^T\, \mathbf{E}[B_t(w_t)^T D_{t+1} A_t(w_t)] + \\
&\quad \mathbf{E}[A_t(w_t)^T D_{t+1} B_t(w_t)] K_t \\
d_t &= \mathbf{E}\, q_t(w_t) + K_t^T\, \mathbf{E}\, r_t(w_t) + 2\, \mathbf{E}\, K_t^T R_t(w_t) l_t + \\
&\quad \mathbf{E}[A_t(w_t) + B_t(w_t) K_t)^T (d_{t+1} + 2 D_{t+1}(B_t(w_t) l_t + \mathbf{E}\, c(w_t))] \\
b_t &= b_{t+1} + \mathbf{E}\, R_t(w_t) l_t^2 + \mathbf{E}\, r_t(w_t) l_t + \\
&\quad \mathbf{E}[(B_t(w_t) l_t + c(w_t))^T D_{t+1} + d_{t+1}^T)(B_t(w_t) l_t + c(w_t))].
\end{aligned}$$

We thus completed the induction step, and so the value function is quadratic and the policy affine at every time step $t = 1, \ldots, T$. The recursion can be solved as long as we know the functional form of the problem parameters and the distribution of the disturbances w_t.

C.2 SHDP Solution

We derive the recursive formulas for (4.27) and (4.28). These are equivalent to the Riccati equations we derived in Appendix C.1, but the expected values are taken over the marginal conditional densities $\hat{f}_{w_\tau|t}(\cdot)$. We write $\mathbf{E}_t$ to denote such expectation. In addition, these equations are somewhat simpler since our problem has $A_t(w_t) = I$, $B_t(w_t) = e_1$, $q_t = 0$, and $r_t(w_t) = r_t$ for $t = 1, \ldots, T$. The final conditions are fixed by (4.25)

$$
\begin{aligned}
D_{T|t} &= \underset{t}{\mathbf{E}}\, Q_T(w_T) + e_1 \underset{t}{\mathbf{E}}\, R_T(w_T) e_1^T \\
d_{T|t} &= -r_T e_1^T - 2C \underset{t}{\mathbf{E}}\, R_T(w_T) e_1^T, \\
b_{T|t} &= r_T C.
\end{aligned}
$$

And the recursive equations are

$$
\begin{aligned}
K_{\tau|t} &= -\frac{e_1^T D_{\tau+1|t}}{\mathbf{E}_t\, R_\tau(w_\tau) + e_1^T D_{\tau+1} e_1} \\
l_{\tau|t} &= -\frac{r_\tau + d_{\tau+1|t}^T e_1 + 2 e_1^T D_{\tau+1|t}\, \mathbf{E}_t\, c(w_\tau)}{2(\mathbf{E}_t\, R_\tau(w_\tau) + e_1^T D_{\tau+1|t} e_1)} \\
D_{\tau|t} &= \underset{t}{\mathbf{E}}\, Q_\tau(w_\tau) + K_{\tau|t}^T \left(\underset{t}{\mathbf{E}}[R_\tau(w_\tau)] + e_1^T D_{\tau+1|t} e_1 \right) K_{\tau|t} + \\
&\quad D_{\tau+1|t} + K_{\tau|t}^T e_1^T D_{\tau+1|t} + D_{\tau+1|t} e_1 K_{\tau|t} = \\
&\quad \underset{t}{\mathbf{E}}\, Q_\tau(w_\tau) + D_{\tau+1|t} + K_{\tau|t}^T e_1^T D_{\tau+1|t} \\
d_{\tau|t} &= K_{\tau|t}^T r_\tau + 2 K_{\tau|t}^T \underset{t}{\mathbf{E}}\, R_\tau(w_\tau) l_{\tau|t} + \\
&\quad (I + e_1 K_{\tau|t})^T (d_{\tau+1|t} + 2 D_{\tau+1|t}(e_1 l_{\tau|t} + \underset{t}{\mathbf{E}}\, c(w_\tau))) = \\
&\quad d_{\tau+1|t} + 2 D_{\tau+1|t}(e_1 l_{\tau|t} + \underset{t}{\mathbf{E}}\, c(w_\tau)) \\
b_{\tau|t} &= b_{\tau+1|t} + r_\tau l_{\tau|t} + \underset{t}{\mathbf{E}}\, R_\tau(w_\tau) l_{\tau|t}^2 + \underset{t}{\mathbf{E}}[c(w_\tau) D_{\tau+1|t} c(w_\tau)] + \\
&\quad d_{\tau+1|t}^T (e_1 l_{\tau|t} + \underset{t}{\mathbf{E}}\, c(w_\tau)) + l_{\tau|t} e_1^T D_{\tau+1|t} (e_1 l_{\tau|t} + 2 \underset{t}{\mathbf{E}}\, c(w_\tau)) = \\
&\quad b_{\tau+1|t} + \underset{t}{\mathbf{E}}[c(w_\tau) D_{\tau+1|t} c(w_\tau)] + d_{\tau+1|t}^T \underset{t}{\mathbf{E}}\, c(w_\tau)
\end{aligned}
$$

for $\tau = t, \ldots, T-1$.

C.3 SHDP simplied solution (without value function)

Parts of the equations derived in Appendix C.2 are superfluous in case we are not interested in the cost-to-go functions $v_{\tau|t}(x_t)$ for $\tau = t, \ldots, T-1$. (In fact, we only want to compute the optimal action (4.29).) We disregard the constant term $b_{\tau|t}$, and we only compute the three scalar elements that we need from $D_{\tau|t}$ and $d_{\tau|t}$. For any $t = 1, \ldots, T$ and $\tau = t, \ldots, T-1$ we define

$$\begin{aligned} e_1^t D_{\tau|t} e_1 &\equiv \beta_{\tau|t} \\ e_1^t D_{\tau|t} e_2 = e_2^t D_{\tau|t} e_1 &\equiv \gamma_{\tau|t} \\ e_1^t d_{\tau|t} &\equiv \delta_{\tau|t} \end{aligned}$$

where $e_1 = (1, 0)$ and $e_2 = (0, 1)$ are the unit vectors. The final values are

$$\begin{aligned} \beta_{T|t} &= \lambda \frac{\sigma_T^2}{C^2} + \frac{\alpha s_T}{2C} \underset{t}{\mathbf{E}}[1/m_T] \\ \gamma_{T|t} &= -\lambda \frac{\sigma_T^2}{C} \underset{t}{\mathbf{E}}[1/V] \\ \delta_{T|t} &= \frac{s_T}{2C} - \alpha s_T \underset{t}{\mathbf{E}}[1/m_T]. \end{aligned}$$

The policy

$$\begin{aligned} K_{\tau|t} &= -\frac{(\beta_{\tau+1|t}, \gamma_{\tau+1|t})}{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t}} \\ l_{\tau|t} &= -\frac{-s_\tau/(2C) + \delta_{\tau+1|t} + 2\gamma_{\tau+1|t}\, \mathbf{E}_t\, m_\tau}{2((\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t})}. \end{aligned}$$

We restrict the Riccati equations to these three scalars. They are independent from the rest of the recursion and we obtain

$$\begin{aligned} \beta_{\tau|t} &= \lambda \frac{\sigma_\tau^2}{C^2} - \frac{\beta_{\tau+1|t}^2}{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t}} + \beta_{\tau+1|t} = \\ &\quad \lambda \frac{\sigma_\tau^2}{C^2} + \frac{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] \beta_{\tau+1|t}}{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t}} \\ \gamma_{\tau|t} &= -\lambda \frac{\sigma_\tau^2}{C} \underset{t}{\mathbf{E}}[1/V] - \frac{\beta_{\tau+1|t}\gamma_{\tau+1|t}}{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t}} + \gamma_{\tau+1|t} = \\ &\quad -\lambda \frac{\sigma_\tau^2}{C} \underset{t}{\mathbf{E}}[1/V] + \frac{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] \gamma_{\tau+1|t}}{(\alpha s_\tau/2C)\, \mathbf{E}_t[1/m_\tau] + \beta_{\tau+1|t}} \\ \delta_{\tau|t} &= \delta_{\tau+1|t} + 2\beta_{\tau+1|t} l_{\tau|t} + 2\gamma_{\tau+1|t} \underset{t}{\mathbf{E}}\, m_t. \end{aligned}$$

C.3.1 Negligible spread

We study the case where $s_t = 0$ for all $t = 1, \dots, T$, equivalent to the limit $\lambda \to \infty$. From the equations above we get that for all $t = 1, \dots, T$ and $\tau = t, \dots, T$

$$\begin{aligned}
\beta_{\tau|t} &= \lambda \frac{\sigma_\tau^2}{C^2} \\
\gamma_{\tau|t} &= -\lambda \frac{\sigma_\tau^2}{C} \mathop{\mathbf{E}}_t [1/V] \\
\delta_{\tau|t} &= 0.
\end{aligned}$$

So for every $t = 1, \dots, T$

$$\mu_{t|t}(x_t) = K_{t|t} x_t + l_t = \frac{-(\beta_{\tau+1|t}, \gamma_{\tau+1|t}) x_t - \mathbf{E}_t \, m_t \gamma_{\tau+1|t}}{\beta_{t+1|t}} =$$

$$C \mathop{\mathbf{E}}_t [1/V] \left(\sum_{\tau=1}^{t-1} m_\tau + \mathop{\mathbf{E}}_t m_t \right) - \sum_{\tau=1}^{t-1} u_\tau .$$

In other words, at every point in time we look at the difference between the fraction of order volume we have executed and the fraction of daily volume the market has traded (using our most recent estimate of the total volume). We trade the expected fraction for next period $C \, \mathbf{E}_t \, [1/V] \, \mathbf{E}_t \, m_t$, plus this difference.

Appendix D

Volume model

We explain here the details of the volume model (4.30), which we use for the dynamic VWAP solution. In §D.1 we describe the *ad hoc* procedure we use to estimate the parameters of the model on historical data. Then in §D.2 we detail the cross-validation of a particular feature of the model. Finally in §D.3 we derive formulas for the expected values of some functions of the volume, which we need for the solution (4.29).

D.1 Estimation on historical data

We consider estimation of the volume model parameters $b^{(k)}$, μ and Σ using data from days $i - W, \ldots, i - 1$ (we are solving the problem at day i). We append the superscript (i, k) to any quantity that refers to market day i and stock k.

Estimation of b^k We first estimate the value of $b^{(k)}$ for each stock k, as:

$$\hat{b}^{(k)} = \frac{\sum_{j=i-W}^{i-1} \sum_{t=1}^{T} \log m_t^{(j,k)}}{TW}$$

We show in Table D.1 the values of $\hat{b}^{(k)}$ obtained on the first W days of our dataset.

Stock	$\hat{b}^{(k)}$	Stock	$\hat{b}^{(k)}$
AA	4.338	JPM	4.599
AXP	3.910	KO	4.312
BA	3.845	MCD	4.017
BAC	5.309	MMM	3.701
CAT	4.118	MRK	4.176
CSCO	4.693	MSFT	4.848
CVX	3.986	PFE	4.586
DD	3.990	PG	4.088
DIS	4.055	T	4.566
GE	4.784	TRV	3.546
HD	4.139	UNH	3.902
HPQ	4.577	UTX	3.782
IBM	3.788	VZ	4.225
INTC	4.860	WMT	3.992
JNJ	4.244	XOM	4.260

Table D.1: Empirical estimate $\hat{b}^{(k)}$ of the per-stock component of the volume model, using data from the first $W = 20$ days.

Estimation of μ Since each observation $\log m^{(j,k)} - \mathbf{1} b^{(j,k)}$ is distributed as a multivariate Gaussian we use this empirical mean as estimator of μ:

$$\hat{\mu}_t = \frac{\sum_{j=i-W}^{i-1} \sum_{k=1}^{K} \log m_t^{(j,k)} - \hat{b}^{(k)}}{WK}.$$

We plot in Figure (D.1) the value of $\hat{\mu}$ obtained on the first W days of our dataset.

Estimation of Σ We finally turn to the estimation of the covariance matrix $\Sigma \in \mathbf{S}^T_{++}$, using historical data. In general, empirical estimation of covariance matrices is a complicated problem. Typically one has not access to enough data to avoid overfitting (a covariance matrix has $O(N^2)$ degrees of freedom, where N is the dimension of a sample). Many approximate approaches have been developed in the econometrics and statistics literature. We designed an *ad hoc* procedure, inspired by works such

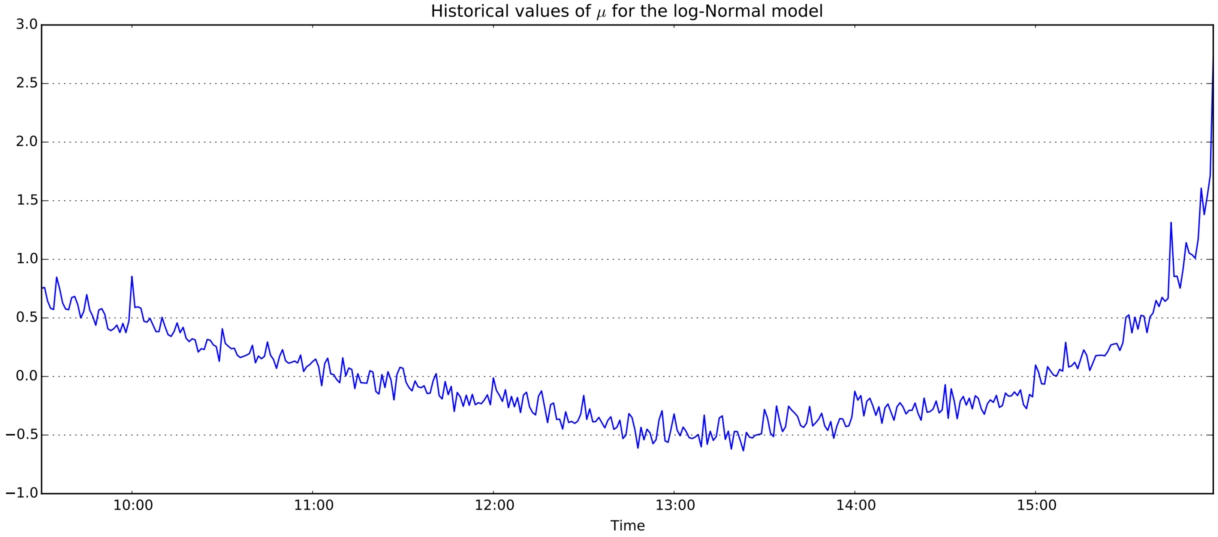

Figure D.1: Empirical estimate $\hat{\mu}$ of the cross-time component of the volume model, using data from the first $W = 20$ days.

as [47]. We look for a matrix of the form

$$\Sigma = ff^T + S,$$

where $f \in \mathbf{R}^T$ and $S \in \mathbf{S}^T_{++}$ is sparse. We first build the empirical covariance matrix. Let $X \in \mathbf{R}^{T \times (WK)}$ be the matrix whose columns are vectors of the form:

$$\log m^{(j,k)} - \mathbf{1}\hat{b}^{(k)} - \hat{\mu}$$

for each day $j = i - W, \ldots, i - 1$ and stock $k = 1, \ldots, K$. Then the empirical covariance matrix is

$$\hat{\Sigma} = \frac{1}{WK - 1} XX^T.$$

We perform the singular value decomposition of X

$$X = U \cdot \mathbf{diag}(s_1, s_2, \ldots, s_T) \cdot V^T,$$

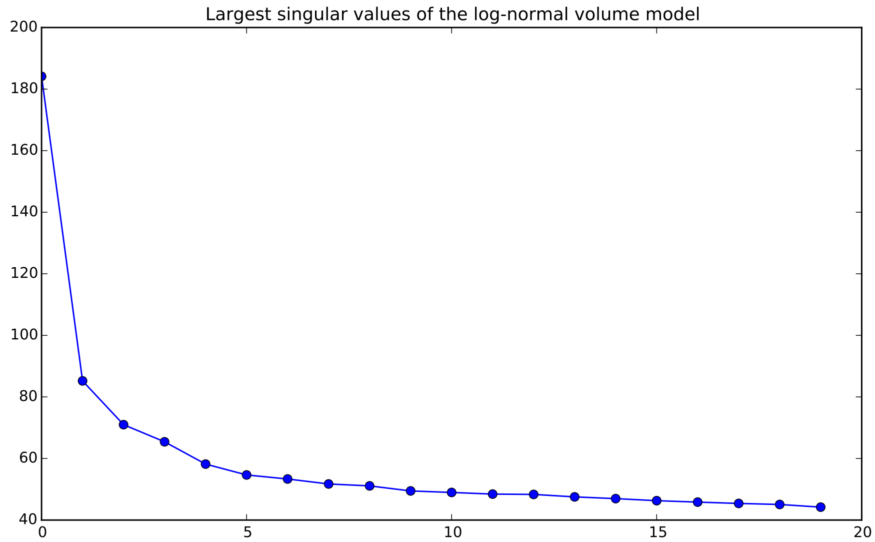

Figure D.2: First 20 singular values of the matrix X of observations $\log m^{(j,k)} - \mathbf{1}\hat{b}^{(k)} - \hat{\mu}$.

where $s \in \mathbf{R}^T$, $s_1 \geq s_2 \geq ... \geq s_T \geq 0$, $U \in \mathbf{R}^{T \times T}$, and $V \in \mathbf{R}^{(WK) \times T}$ (because in practice we have $WK > T$, since $W = 20$, $K = 30$, and $T = 390$). We have

$$\hat{\Sigma} = \frac{1}{WK - 1} U \cdot \mathbf{diag}(s_1^2, s_2^2, \ldots, s_T^2) \cdot U^T.$$

We show in Figure D.2 the first singular values $s_1, s_2, \ldots, s_{20}$ computed on data from the first W days. It is clear that the first singular value is much larger than all the others. We thus build the rank 1 approximation of the empirical covariance matrix by keeping the first singular value and first (left) singular vector

$$f = \frac{s_1 U_{:,1}}{\sqrt{WK - 1}},$$

so that ff^T is the best (in Frobenius norm) rank-1 approximation of $\hat{\Sigma}$. We now need to provide an approximation for the sparse part S of the covariance matrix. We assume that S is a *banded* matrix of bandwidth $b > 0$, which is non-zero only on the main diagonal and on $b-1$ diagonals above and below it (in total it has $2b-1$ non-zero diagonals). The value of b is chosen by cross-validation, as explained in §D.2. The assumption that S is banded is inspired by the intuition that elements of $\log m^{(j,k)} - \mathbf{1}b^{(k)} - \mu$ are correlated (in time) for short delays. We find S by simply copying the diagonal elements of the empirical covariance matrix:

$$\cdot S_{i,j} = \begin{cases} (\hat{\Sigma} - ff^T)_{i,j} & \text{if } |j-i| \leq b \\ 0 & \text{otherwise.} \end{cases}$$

We thus have built a matrix of the form $\Sigma = ff^T + S$. Note that this procedure does not guarantee that Σ is positive definite. However in our empirical tests we always got positive definite Σ for any $b = 1, 2, \ldots$.

D.2 Cross validation

As explained in §D.1, we need to choose the value of the parameter $b \in \mathbb{N}$ (used for empirical estimation of the covariance matrix Σ). We choose it by cross-validation, reserving the first $W_{CV} = 10$ testing days of the dataset. We show in Figure 4.2 the way we partition the data (so that the empirical testing is performed out-of-sample with respect to the cross-validation). We simulate trading according to the solution (4.29) with $\lambda = \infty$ (*i.e.*, the special case of Appendix C.3.1), for various values of b. We then compute the empirical variance of S, and choose the value of b which minimizes it. (We are mostly interested in optimizing the variance of S, rather than the transaction costs.) In Figure D.3 we show the result of this procedure (we show the standard deviations instead of variances, for simplicity), along with the result using the static solution (4.16), for comparison. Since the difference in performance between $b = 3$ and $b = 5$ is small (and we want to avoid overfitting), we choose $b = 3$.

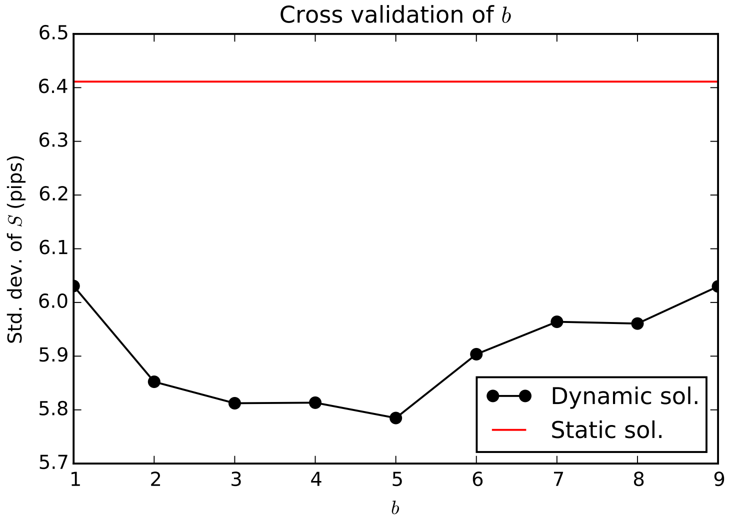

Figure D.3: Cross-validation of the volume model parameter b.

D.3 Expected values of interest

We consider the problem at any fixed time $t = 1, \dots, T-1$, for a given stock k and day i. (We have observed market volumes $m_1, \dots, m_{t-1}$.) We obtain the conditional distribution of the unobserved volumes $m_t, \dots, m_T$ and derive expressions for $\mathbf{E}_t\, m_\tau$, $\mathbf{E}_t\left[\frac{1}{m_\tau}\right]$, and $\mathbf{E}_t\left[\frac{1}{V}\right]$ for any $\tau = t, \dots, T$. We need these for the numerical solution (4.29), as developed in Appendix C.3.1.

Conditional distribution We divide the covariance matrix in blocks:

$$\Sigma = \begin{pmatrix} \Sigma_{1:(t-1),1:(t-1)} & \Sigma_{1:(t-1),t:T} \\ \Sigma_{t:T,1:(t-1)} & \Sigma_{t:T,t:T} \end{pmatrix}.$$

Then we get the marginal distribution

$$m_{t:T} \sim \log \mathcal{N}(\nu|t, \Sigma|t)$$

by taking the *Schur complement* (*e.g.*, [21]) of the covariance matrix

$$\begin{aligned}\nu|t &\equiv \mu_{t:T} + b^{(k)} + \Sigma^T_{1:(t-1),t:T}\Sigma^{-1}_{1:(t-1),1:(t-1)}(\log m_{1:(t-1)} - \mu_{1:(t-1)} - b^{(k)}) \\ \Sigma|t &\equiv \Sigma_{t:T,t:T} - \Sigma^T_{1:(t-1),t:T}\Sigma^{-1}_{1:(t-1),1:(t-1)}\Sigma_{1:(t-1),t:T}.\end{aligned}$$

Note that $\nu|1 = \mu + b^{(k)}$ and $\Sigma|1 = \Sigma$, *i.e.*, the unconditional distribution of the market volumes. We now develop the conditional expectation expressions.

Volumes The expected value of the remaining volumes m_τ

$$\mathop{\mathbf{E}}_t m_\tau = \exp\left((\nu|t)_{\tau-t+1} + \frac{(\Sigma|t)_{\tau-t+1,\tau-t+1}}{2}\right), \quad \tau = t,\ldots,T.$$

(Because the $(\tau - t + 1)$-th element of $\nu|t$ corresponds to the τ-th volume.)

Inverse volumes The expected value of the inverse of the remaining volumes m_τ

$$\mathop{\mathbf{E}}_t \left[\frac{1}{m_\tau}\right] = \exp\left(-(\nu|t)_{\tau-t+1} + \frac{(\Sigma|t)_{\tau-t+1,\tau-t+1}}{2}\right), \quad \tau = t,\ldots,T.$$

Total volume We have, since we already observed $m_1, \ldots, m_{t-1}$

$$\mathop{\mathbf{E}}_t V = \sum_{\tau=1}^{t-1} m_\tau + \sum_{\tau=t}^{T} \mathop{\mathbf{E}}_t m_\tau.$$

We also express its variance, which we need later

$$\mathop{\mathbf{var}}_t(V) = \mathop{\mathbf{var}}_t \sum_{\tau=t}^{T} m_\tau = \sum_{\tau=t}^{T}\sum_{\tau'=t}^{T} \mathbf{cov}(m_\tau, m_{\tau'}) = \\ \sum_{\tau=t}^{T}\sum_{\tau'=t}^{T} \mathop{\mathbf{E}}_t m_\tau \mathop{\mathbf{E}}_t m_{\tau'} \left(\exp((\Sigma|t)_{\tau-t+1,\tau'-t+1}) - 1\right).$$

Inverse total volume We use the following approximation, derived from the Taylor expansion formula. Consider a random variable z and a smooth function $\phi(\cdot)$, then

$$\mathbf{E}\,\phi(z) \simeq \phi(\mathbf{E}\,z) + \frac{\phi''(\mathbf{E}\,z)}{2}\,\mathbf{var}\,z.$$

So the inverse total volume

$$\mathop{\mathbf{E}}_{t}\left[\frac{1}{V}\right] \simeq \frac{1}{\mathbf{E}_t\,V} + \frac{\mathbf{var}_t(V)}{\mathbf{E}_t[V]^3}.$$

Bibliography

[1] Wharton Research Data Services, TAQ Dataset. `https://wrds-web.wharton.upenn.edu/wrds/`.

[2] R. Almgren. High frequency volatility. Available at `http://cims.nyu.edu/~almgren/timeseries/notes7.pdf`, 2009.

[3] R. Almgren and N. Chriss. Optimal execution of portfolio transactions. *Journal of Risk*, 3(2):5–39, 2001.

[4] C. Bacon. *Practical Portfolio Performance Measurement and Attribution*. Wiley, 2nd edition, 2008.

[5] D. Bailey, J. Borwein, M. de Prado, and Q. Zhu. The probability of backtest overfitting. *Journal of Computational Finance*, 20(4):39–69, 2017.

[6] R. Bellman. Dynamic programming and Lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

[7] A. Bemporad. Model predictive control design: New trends and tools. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 6678–6683. IEEE, 2006.

[8] A. Bemporad, L. Bellucci, and T. Gabbriellini. Dynamic option hedging via stochastic model predictive control based on scenario simulation. *Quantitative Finance*, 14(10):1739–1751, 2014.

[9] N. Bershova and D. Rakhlin. The non-linear market impact of large trades: Evidence from buy-side order flow. *Quantitative Finance*, 13(11):1759–1778, 2013.

[10] D. Bertsekas. Stochastic optimization problems with nondifferentiable cost functionals. *Journal of Optimization Theory and Applications*, 12(2):218–231, 1973.

[11] D. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[12] D. Bertsekas. *Convex Optimization Theory*. Athena Scientific, 2009.

[13] D. Bertsimas and A. Lo. Optimal control of execution costs. *Journal of Financial Markets*, 1(1):1–50, 1998.

[14] J. Białkowski, S. Darolles, and G. Le Fol. Improving vwap strategies: A dynamic volume approach. *Journal of Banking & Finance*, 32(9):1709–1722, 2008.

[15] F. Black. Studies of stock price volatility changes. In *Proceedings of the 1976 Meetings of the American Statistical Association, Business and Economic Section*, pages 177–181. American Statistical Association, 1976.

[16] J. Bouchaud, J. Farmer, and F. Lillo. How markets slowly digest changes in supply and demand. In T. Hens and K. Schenk-Hoppe, editors, *Handbook of Financial Markets: Dynamics and Evolution*, pages 57–160. Elsevier, 2009.

[17] S. Boyd, E. Busseti, S. Diamond, R. Kahn, K. Koh, P. Nystrup, and J. Speth. Multi-period trading via convex optimization. *Foundations and Trends in Optimization*, 3(1):1–76, August 2017.

[18] S. Boyd, S. Lall, and B. Van Roy. Ee365: Stochastic control. `http://stanford.edu/class/ee365/lectures.html`. 2012.

[19] S. Boyd, M. Mueller, B. O'Donoghue, and Y. Wang. Performance bounds and suboptimal policies for multi-period investment. *Foundations and Trends in Optimization*, 1(1):1–69, 2014.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[21] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[22] S. Browne. Risk-constrained dynamic active portfolio management. *Management Science*, 46(9):1188–1199, 2000.

[23] S. Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015.

[24] E. Busseti and S. Boyd. Volume weighted average price optimal execution. *arXiv preprint arXiv:1509.08503*, 2015.

[25] E. Busseti, S. Diamond, S. Boyd, and Blackrock. `CVXPortfolio`, 2017. Available at `https://github.com/cvxgrp/cvxportfolio`.

[26] E. Busseti, E. Ryu, and S. Boyd. Risk-constrained Kelly gambling. *Journal of Investing*, 25(3), 2016.

[27] J. Campbell, A. Lo, and A. MacKinlay. *The Econometrics of Financial Markets.* Princeton University Press, 1997.

[28] J. Campbell and L. Viceira. *Strategic Asset Allocation: Portfolio Choice for Long-Term Investors.* Oxford University Press, 2002.

[29] A. Cartea and S. Jaimungal. A closed-form execution strategy to target VWAP. *Available at SSRN 2542314*, 2014.

[30] A. Cartea, S. Jaimungal, and J. Penalva. *Algorithmic and High-Frequency Trading.* Cambridge University Press, 2015.

[31] L. Chan, J. Karceski, and J. Lakonishok. On portfolio optimization: Forecasting covariances and choosing the risk model. *The Review of Financial Studies*, 12(5):937–974, 1999.

[32] E. Cho, K. Thoney, T. Hodgson, and R. King. Supply chain planning: Rolling horizon scheduling of multi-factory supply chains. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, pages 1409–1416. Winter Simulation Conference, 2003.

[33] V. Chopra and W. Ziemba. The effect of errors in means, variances, and covariances on optimal portfolio choice. *Journal of Portfolio Management*, 19(2):6–11, 1993.

[34] G. Constantinides. Multiperiod consumption and investment behavior with convex transactions costs. *Management Science*, 25(11):1127–1137, 1979.

[35] G. Cornuejols and R. Tütüncü. *Optimization Methods in Finance.* Cambridge University Press, 2006.

[36] T. Cover and J. Thomas. *Elements of Information Theory.* Wiley, 2012.

[37] M. Davis and S. Lleo. Fractional Kelly strategies in continuous time: Recent developments. In L. MacLean and W. Ziemba, editors, *Handbook of the Fundamentals of Financial Decision Making*, pages 753–788. World Scientific Publishing, 2012.

[38] M. Davis and A. Norman. Portfolio selection with transaction costs. *Mathematics of Operations Research*, 15(4):676–713, 1990.

[39] V. DeMiguel, L. Garlappi, F. Nogales, and R. Uppal. A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. *Management Science*, 55(5):798–812, 2009.

[40] V. DeMiguel, L. Garlappi, and R. Uppal. Optimal versus naïve diversification: How inefficient is the $1/N$ portfolio strategy? *Review of Financial Studies*, 22(5):1915–1953, 2009.

[41] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 2016.

[42] S. Diamond, R. Takapoui, and S. Boyd. A general system for heuristic solution of convex problems over nonconvex sets. *Optimization Methods and Software*, 33(1):165–193, 2018.

[43] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013.

[44] B. Dumas and E. Luciano. An exact solution to a dynamic portfolio choice problem under transactions costs. *Journal of Finance*, 46(2):577–595, 1991.

[45] E. Elton, M. Gruber, and C. Blake. Survivor bias and mutual fund performance. *The Review of Financial Studies*, 9(4):1097–1120, 1996.

[46] F. Fabozzi, D. Huang, and G. Zhou. Robust portfolios: Contributions from operations research and finance. *Annals of Operations Research*, 176(1):191–220, 2010.

[47] J. Fan, Y. Liao, and M. Mincheva. High dimensional covariance matrix estimation in approximate factor models. *Annals of statistics*, 39(6):3320, 2011.

[48] B. Fastrich, S. Paterlini, and P. Winker. Constructing optimal sparse portfolios using regularization methods. *Computational Management Science*, 12(3):417–434, 2015.

[49] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 2. Wiley, 2nd edition, 1971.

[50] C. Fougner and S. Boyd. Parameter selection and pre-conditioning for a graph form solver. In R. Tempo, S. Yurkovich, and P. Misra, editors, *Emerging Applications of Control and System Theory*. Springer, 2017.

[51] C. Frei and N. Westray. Optimal execution of a VWAP order: a stochastic control approach. *Mathematical Finance*, 25(3):612–639, 2015.

[52] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*. Springer, 2001.

[53] M. Frittelli and E. Gianin. Putting order in risk measures. *Journal of Banking & Finance*, 26(7):1473–1486, 2002.

[54] R. Gallager. *Stochastic Processes: Theory for Applications.* Cambridge University Press, 2013.

[55] N. Gârleanu and L. Pedersen. Dynamic trading with predictable returns and transaction costs. *Journal of Finance*, 68(6):2309–2340, 2013.

[56] D. Goldsmith. Transactions costs and the theory of portfolio selection. *Journal of Finance*, 31(4):1127–1139, 1976.

[57] C. Gomes and H. Waelbroeck. Is market impact a measure of the information value of trades? Market response to liquidity vs. informed metaorders. *Quantitative Finance*, 15(5):773–793, 2015.

[58] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[59] M. Grant, S. Boyd, and Y. Ye. Disciplined convex programming. In L. Liberti and N. Maculan, editors, *Global Optimization: From Theory to Implementation*, Nonconvex Optimization and its Applications, pages 155–210. Springer, 2006.

[60] R. Grinold. A dynamic model of portfolio management. *Journal of Investment Management*, 4(2):5–22, 2006.

[61] R. Grinold and R. Kahn. *Active Portfolio Management: A Quantitative Approach for Providing Superior Returns and Controlling Risk.* McGraw-Hill, 2nd edition, 1999.

[62] O. Guéant and G. Royer. Vwap execution and guaranteed vwap. *arXiv preprint arXiv:1306.2832*, 2013.

[63] F. Herzog, G. Dondi, and H. Geering. Stochastic model predictive control and portfolio optimization. *International Journal of Theoretical and Applied Finance*, 10(2):203–233, 2007.

[64] M. Ho, Z. Sun, and J. Xin. Weighted elastic net penalized mean–variance portfolio design and computation. *SIAM Journal on Financial Mathematics*, 6(1):1220–1244, 2015.

[65] M. Humphery-Jenner. Optimal VWAP trading under noisy conditions. *Journal of Banking & Finance*, 35(9):2319–2329, 2011.

[66] R. Jagannathan and T. Ma. Risk reduction in large portfolios: Why imposing the wrong constraints helps. *Journal of Finance*, 58(4):1651–1683, 2003.

[67] P. Jorion. International portfolio diversification with estimation risk. *Journal of Business*, 58(3):259–278, 1985.

[68] R. Kan and G. Zhou. Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis*, 42(3):621–656, 2007.

[69] J. Kelly. A new interpretation of information rate. *IRE Transactions on Information Theory*, 2(3):185–189, 1956.

[70] A. Keshavarz and S. Boyd. Quadratic approximate dynamic programming for input-affine systems. *International Journal of Robust and Nonlinear Control*, 24(3):432–449, 2014.

[71] R. Kissell and M. Glantz. *Optimal Trading Strategies: Quantitative Approaches for Managing Market Impact and Trading Risk.* AMACOM, 2003.

[72] P. Kolm, R. Tütüncü, and F. Fabozzi. 60 years of portfolio optimization: Practical challenges and current trends. *European Journal of Operational Research*, 234(2):356–371, 2014.

[73] H. Konishi. Optimal slice of a VWAP trade. *Journal of Financial Markets*, 5(2):197–221, 2002.

[74] H. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications.* Springer, 2nd edition, 2003.

[75] W. Kwon and S. Han. *Receding Horizon Control: Model Predictive Control for State Models*. Springer, 2006.

[76] J. Li. Sparse and stable portfolio selection with parameter uncertainty. *Journal of Business & Economic Statistics*, 33(3):381–392, 2015.

[77] T. Li. *Dynamic Programming and Trade Execution.* PhD thesis, Princeton University, 2013.

[78] F. Lillo, J. Farmer, and R. Mantegna. Econophysics: Master curve for price-impact function. *Nature*, 421(129):176–190, 2003.

[79] M. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):341–365, 2007.

[80] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the IEEE International Symposium on Computer Aided Control Systems Design*, pages 284–289, 2004.

[81] L. MacLean, E. Thorp, and W. Ziemba. The Kelly capital growth investment criterion: Theory and practice. volume 3 of *World Scientific Handbook in Financial Economics*. World Scientific Publishing, 2011.

[82] A. Madhavan. VWAP strategies. *Trading*, 2002(1):32–39, 2002.

[83] H. Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[84] J. Mattingley and S. Boyd. CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, 2012.

[85] J. Mattingley, Y. Wang, and S. Boyd. Receding horizon control. *Control Systems*, 31(3):52–65, 2011.

[86] J. McCulloch and V. Kazakov. Mean variance optimal vwap trading. *Available at SSRN 1803858*, 2012.

[87] W. McKinney. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.* O'Reilly Media, 2012.

[88] R. Merton. Lifetime portfolio selection under uncertainty: The continuous-time case. *Review of Economics and Statistics*, 51(3):247–257, 1969.

[89] R. Merton. Optimum consumption and portfolio rules in a continuous-time model. *Journal of Economic Theory*, 3(4):373–413, 1971.

[90] R. Merton. *Continuous-Time Finance.* Wiley, 1990.

[91] A. Meucci. *Risk and Asset Allocation.* Springer, 2005.

[92] A. Meucci. Historical scenarios with fully flexible probabilities. *GARP Risk Professional*, pages 47–51, 2010.

[93] R. Michaud. The Markowitz optimization enigma: Is 'optimized' optimal? *Financial Analysts Journal*, 45(1):31–42, 1989.

[94] C. Moallemi and M. Sağlam. Dynamic portfolio choice with linear rebalancing rules. *Journal of Financial and Quantitative Analysis*, 52:1247–1278, 2017.

[95] E. Moro, J. Vicente, L. Moyano, A. Gerig, J. Farmer, G. Vaglica, F. Lillo, and R. Mantegna. Market impact and trading profile of hidden orders in stock markets. *Physical Review E*, 80(6):066102, 2009.

[96] R. Narang. *Inside the Black Box: A Simple Guide to Quantitative and High Frequency Trading.* Wiley, 2013.

[97] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[98] A. Nemirovski and D. Yudin. On Cesari's convergence of the steepest descent method for approximating saddle points of convex-concave functions. *Doklady Akademii Nauk SSSR*, 239:1056–1059, 1978.

[99] A. Nemirovski and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983.

[100] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[101] P. Nystrup, H. Madsen, and E. Lindström. Dynamic portfolio optimization across hidden market regimes. Working paper, Technical University of Denmark, 2016.

[102] A. Obizhaeva and J. Wang. Optimal trading strategy and supply/demand dynamics. *Journal of Financial Markets*, 16(1):1–32, 2013.

[103] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

[104] A. Perold. Large-scale portfolio optimization. *Management Science*, 30(10):1143–1160, 1984.

[105] A. Perold. The implementation shortfall: Paper versus reality. *Journal of Portfolio Management*, 31:106, 1998.

[106] B. Polyak. *Introduction to Optimization*. Optimization Software, New York, 1987.

[107] W. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2007.

[108] Quandl. WIKI end-of-day data, 2016. Available at `https://www.quandl.com/data/WIKI`.

[109] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[110] R. T. Rockafellar. *Conjugate Duality and Optimization*. SIAM, 1974.

[111] P. Samuelson. Lifetime portfolio selection by dynamic stochastic programming. *Review of Economics and Statistics*, 51(3):239–246, 1969.

[112] S. Serrano. *Algorithms for Unsymmetric Cone Optimization and an Implementation for Problems with the Exponential Cone*. PhD thesis, Stanford University, 2015.

[113] W. Sharpe. Mutual fund performance. *Journal of Business*, 39(1):119–138, 1966.

[114] W. Sharpe. The arithmetic of active management. *Financial Analysts Journal*, 47(1):7–9, 1991.

[115] W. Sharpe. The Sharpe ratio. *Journal of Portfolio Management*, 21(1):49–58, 1994.

[116] J. Skaf, S. Boyd, and A. Zeevi. Shrinking-horizon dynamic programming. *International Journal of Robust and Nonlinear Control*, 20(17):1993–2002, 2010.

[117] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, pages 267–288, 1996.

[118] M. Udell, K. Mohan, D. Zeng, J. Hong, S. Diamond, and S. Boyd. Convex optimization in Julia. *SC14 Workshop on High Performance Technical Computing in Dynamic Languages*, 2014.

[119] A. Wald. On cumulative sums of random variables. *The Annals of Mathematical Statistics*, 15(3):283–296, 1944.

[120] P. Whittle. Risk-sensitive linear/quadratic/Gaussian control. *Advances in Applied Probability*, 13(4):764–777, 1981.

[121] P. Whittle. *Risk-sensitive Optimal Control*. Wiley, 1990.