Foundations and Trends® in Optimization
Vol. 3, No. 1 (2016) 1–76
© 2017 S. Boyd, E. Busseti, S. Diamond, R. N. Kahn, K. Koh, P. Nystrup, J. Speth
DOI: 10.1561/2400000023

Multi-Period Trading via Convex Optimization

Stephen Boyd
Stanford University
boyd@stanford.edu

Enzo Busseti
Stanford University
ebusseti@stanford.edu

Steven Diamond
Stanford University
stevend2@stanford.edu

Ronald N. Kahn
Blackrock
ron.kahn@blackrock.com

Kwangmoo Koh
Blackrock
kwangmoo.koh@blackrock.com

Peter Nystrup
Technical University of Denmark
pnys@dtu.dk

Jan Speth
Blackrock
jan.speth@blackrock.com

Contents

Abstract

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 paper we do not address a critical component in a trading algorithm, the predictions or forecasts of future quantities. The methods we describe in this paper 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 paper.

S. Boyd, E. Busseti, S. Diamond, R. N. Kahn, K. Koh, P. Nystrup, J. Speth. *Multi-Period Trading via Convex Optimization*. Foundations and Trends® in Optimization, vol. 3, no. 1, pp. 1–76, 2016.
DOI: 10.1561/2400000023.

1

Introduction

Single and multi-period portfolio selection. Markowitz [54] 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 [49]. Goldsmith [38] 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 [53, 63].

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 [74] and Merton [58, 59]. Constantinides [22] extended Samuelson's discrete-time formulation to problems with proportional transaction costs. Davis and Norman [24] and Dumas and Lucian [30] 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 [5, 9], which properly takes into account the idea of recourse and updated information available as the sequence of trades are chosen (see [37] 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' [72, 11]. 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.*, [18] 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 [11]. We will focus on these simple multi-period methods in this paper. 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 [11], 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 tutorial paper 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 [13] 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.*, [42]). This paper 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 chapter 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 chapter 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 chapter 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 chapter 5 we consider *multi-period optimiza-*

tion (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 paper 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 [42], Meucci [60], or Narang [65]. We also assume that the reader has seen some basic mathematical optimization, specifically convex optimization [13]. The reader certainly does not need to know more than the very basic ideas of convex optimization, for example the overview material covered in chapter 1 of [13]. In a nutshell, our target reader is a quantitative trader, or someone who works with or for, or employs, one.

2

The Model

In this chapter 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.

2.1 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 $t = 1, \ldots, T$. 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 paper 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 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.

2.3 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 [41].

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 [42]. 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 [52, 64, 8, 39]. 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 $\phi_t^{\text{trade}}(z_t)$. When the argument to the transaction cost function is normalized, we use the version where asset volume is also normalized. The normalized transaction cost $\phi_t^{\text{trade}}(z_t)$ 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 [2, 41, 37]. 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.

2.4 Holding cost

We will hold the post-trade portfolio h_t^+ over the tth period. This will incur a holding-based cost (in dollars) $\phi_t^{\text{hold}}(h_t^+)$, where $\phi_t^{\text{hold}} : \mathbf{R}^{n+1} \to \mathbf{R}$ 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

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

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.

2.5 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

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

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

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

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

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

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

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

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,
- $-\phi_t^{\mathrm{trade}}(z_t)$ is the transaction cost, and
- $-\phi_t^{\mathrm{hold}}(w_t+z_t)$ 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 $w_{t+1} \approx w_t + z_t$.

2.7 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^{req}, the requested trade, and u_t, the actual realized trade [71]. 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 [2, 68]. 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.

2.8 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 chapter 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.

3

Metrics

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

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^{p}_t is very close to R^{p}_t.

The return and growth rates given above are per-period. For interpretability they are typically *annualized* [3]: 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* [10] is the standard deviation of the portfolio return time series,

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

(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^{p}_t are small (in comparison to 1), a good approximation of the quadratic risk is the second moment of the return,

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

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.)

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 $w^{\mathrm{b}}_t \in \mathbf{R}^{n+1}$, which are fractions of the assets (including cash), and satisfy $\mathbf{1}^T w^{\mathrm{b}}_t = 1$. We will assume the benchmark weights are nonnegative, *i.e.*, the entries in w^{b}_t are nonnegative. The benchmark weight $w^{\mathrm{b}}_t = e_{n+1}$ (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 [76, 42] (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} [42],

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

In the special case of a cash benchmark this is known as *Sharpe ratio* (SR) [75, 77]

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

Both IR and SR are typically given using the annualized values of the return and risk [3].

4

Single-Period Optimization

In this chapter 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 [54], who was the first to formulate the choice of a portfolio as an optimization problem. (We will consider multi-period optimization in the next section.)

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 $\hat{\phi}_t^{\text{trade}}$ 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 [17, 42]. In this paper, 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 paper 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

$$\hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t), \tag{4.1}$$

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

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

$$\begin{array}{ll} \text{maximize} & \hat{R}_t^{\mathrm{p}} - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t \\ & \mathbf{1}^T z_t + \hat{\phi}_t^{\mathrm{trade}}(z_t) + \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) = 0, \end{array} \tag{4.2}$$

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 (4.2) 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 (4.2). 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 (4.2) with $\hat{R}_t^{\mathrm{e}}$ or $\hat{R}_t^{\mathrm{a}}$. By (4.1), 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

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T z_t - \hat{\phi}_t^{\mathrm{trade}}(z_t) - \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t \\ & \mathbf{1}^T z_t + \hat{\phi}_t^{\mathrm{trade}}(z_t) + \hat{\phi}_t^{\mathrm{hold}}(w_t + z_t) = 0, \end{array} \tag{4.3}$$

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 (4.3). 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 (4.3). 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 (4.3) 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 (4.3) by replacing the self-financing constraint

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

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

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T z_t - \hat{\phi}_t^{\text{trade}}(z_t) - \hat{\phi}_t^{\text{hold}}(w_t + z_t) - \gamma_t \psi_t(w_t + z_t) \\ \text{subject to} & \mathbf{1}^T z_t = 0, \quad z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t. \end{array} \tag{4.4}$$

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 (4.4), $w_t + z_t$ is a bona fide set of weights, *i.e.*, $\mathbf{1}^T(w_t + z_t) = 1$; whereas in (4.3), $\mathbf{1}^T(w_t + z_t)$ is (typically) slightly less than one.

We can re-write the problem (4.4) in terms of the variable $w_{t+1} = w_t + z_t$, which we interpret as the post-trade portfolio weights:

$$\begin{array}{ll} \text{maximize} & \hat{r}_t^T w_{t+1} - \hat{\phi}_t^{\text{trade}}(w_{t+1} - w_t) - \hat{\phi}_t^{\text{hold}}(w_{t+1}) - \gamma_t \psi_t(w_{t+1}) \\ \text{subject to} & \mathbf{1}^T w_{t+1} = 1, \quad w_{t+1} - w_t \in \mathcal{Z}_t, \quad w_{t+1} \in \mathcal{W}_t, \end{array} \tag{4.5}$$

with variable w_{t+1}.

4.2 Risk measures

The risk measure ψ_t in (4.3) or (4.4) is traditionally an estimate of the variance of the return, using a stochastic model of the returns [54, 49]. But it can be any function that measures our perceived risk of holding a portfolio [36]. 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 $R_t^{\rm a}$ is

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

This gives the traditional quadratic active risk measure

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

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 (4.3) or (4.4) 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 $R_t^{\rm p} = w^T r_t$ is a (scalar) random variable. The weight vector that maximizes the expected portfolio growth rate $\mathbf{E} \log(1 + R_t^{\rm p})$ (subject to $\mathbf{1}^T w = 1$, $w \geq 0$) is called the *Kelly optimal portfolio* or *log-optimal portfolio* [48, 16]. Using the quadratic approximation of the logarithm $\log(1 + a) \approx a - (1/2)a^2$ we obtain

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

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 (4.3) or (4.4) 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 [16], 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 [19] (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 (4.3) or (4.4), it can offer a very substantial increase in the speed of solution [70, 13]. 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 section 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 (4.3) 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 $\Sigma^{(i)}$, $i = 1, \ldots, M$, 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 [61]. They could be an analyst's best guess for what the asset covariance would be in a situation that could occur.

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 section 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 [46, 62, 21, 47, 26, 32, 49, 4].

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

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

The first term here is our original estimate (including the constant terms we neglect in (4.3) and (4.4)); 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

$$\psi_t(x) = \rho^T |x - w_t^{\mathrm{b}}|. \tag{4.6}$$

(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 [78, 33, 44, 51].

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*

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

where Δ is a symmetric perturbation matrix with

$$|\Delta_{ij}| \leq \kappa \left(\Sigma_{ii}\Sigma_{jj}\right)^{1/2}, \tag{4.7}$$

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

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

This shows that the worst-case covariance, over all perturbed covariance matrices consistent with our risk forecast error assumption (4.7), is given by

$$\psi_t(x) = (x - w_t^{\mathrm{b}})^T \Sigma (x - w_t^{\mathrm{b}}) + \kappa \left(\sigma^T |x - w_t^{\mathrm{b}}|\right)^2, \qquad (4.8)$$

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 [44, 51]. 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.)

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 $R^{\rm p}$ is uncorrelated with the benchmark return $R^{\rm b}$, according to our estimate Σ_t of $\mathbf{cov}(r_t)$. The constraint that $w_t + z_t$ be β neutral takes the form

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

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

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

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

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

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 $(w_t + z_t)/T^{\text{liq}}$ each period. The constraint that the liquidation loss is no more than the fraction δ of the portfolio value is given by

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

(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 [13, section 3.2.3]. It is easily extended to the case where K is not an integer.)

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.

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.

4.7 Convexity

The portfolio optimization problem (4.3) 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 (4.3). Alternatively, the self-financing constraint can be replaced with the simplified version $\mathbf{1}^T z_t = 0$ as in problem (4.4).

Solution times. The SPO problems described above, except for the multi-covariance risk model, can be solved using standard interior-point methods [66] 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$\times$ 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$\times$ 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 [69]. For example the first order operator-splitting method implemented in POGS [34] 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 [55] 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 [34], CVXPY [27], and Convex.jl [79], based on the idea of disciplined convex programming (DCP) [40], 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 [28]. 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.

4.8 Using single-period optimization

The idea. In this section 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.*, [23, chapter 8], [45, 25, 51]), 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 [41]. 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 (4.4) with scaling parameters for transaction and holding costs, in addition to the traditional risk aversion parameter, which yields the SPO problem

$$\begin{array}{ll} \text{maximize} & \Big(\hat{r}_t^T z_t - \gamma_t^{\text{trade}} \hat{\phi}_t^{\text{trade}}(z_t) \\ & \qquad - \gamma_t^{\text{hold}} \hat{\phi}_t^{\text{hold}}(w_t + z_t) - \gamma_t^{\text{risk}} \psi_t(w_t + z_t)\Big) \\ \text{subject to} & \mathbf{1}^T z_t = 0, \quad z_t \in \mathcal{Z}_t, \quad w_t + z_t \in \mathcal{W}_t. \end{array} \tag{4.9}$$

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 [35]. For general discussion of how convex optimization is used in this spirit, in applications such as control or estimation, see [13].

Judging value of forecasts. In this paper 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.

5

Multi-Period Optimization

5.1 Motivation

In this chapter 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.*, [37, 67]). 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.*, [17, 41, 37]).

- *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.

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 section 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 (4.3). 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,

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

(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),

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

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

$$\begin{array}{ll} \text{maximize} & \sum_{\tau=t}^{t+H-1} \Big(\hat{r}_{\tau|t}^T (w_\tau + z_\tau) - \gamma_\tau \psi_\tau (w_\tau + z_\tau) \\ & \qquad - \hat{\phi}_\tau^{\text{hold}}(w_\tau + z_\tau) - \hat{\phi}_\tau^{\text{trade}}(z_\tau) \Big) \\ \text{subject to} & \mathbf{1}^T z_\tau = 0, \quad z_\tau \in \mathcal{Z}_\tau, \quad w_\tau + z_\tau \in \mathcal{W}_\tau, \\ & w_{\tau+1} = w_\tau + z_\tau, \quad \tau = t, \ldots, t+H-1, \end{array} \tag{5.1}$$

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 (4.4). (We can ignore the constant $\hat{r}_{t|t}^T w_t$, which does not depend on the variables, that appears in (5.1) but not (4.4).)

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

$$\begin{array}{ll} \text{maximize} & \sum_{\tau=t+1}^{t+H} \Big(\hat{r}_{\tau|t}^T w_\tau - \gamma_\tau \psi_\tau (w_\tau) \\ & \qquad - \hat{\phi}_\tau^{\text{hold}}(w_\tau) - \hat{\phi}_\tau^{\text{trade}}(w_\tau - w_{\tau-1}) \Big) \\ \text{subject to} & \mathbf{1}^T w_\tau = 1, \quad w_\tau - w_{\tau-1} \in \mathcal{Z}_\tau, \quad w_\tau \in \mathcal{W}_\tau, \\ & \tau = t+1, \ldots, t+H, \end{array} \tag{5.2}$$

with variables $w_{t+1}, \ldots, w_{t+H}$, the planned weights over the next H periods. This is the multi-period analog of (4.5).

Both MPO formulations (5.1) and (5.2) 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 (5.1) or (5.2) 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 (5.1) or (5.2) 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) [50, 56], supply chain optimization [20], and others. Applications of MPC in finance include [43, 11, 7, 14, 67].

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, $w_{t+H} = w^{\text{term}}$. 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.*, $w^{\text{term}} = e_{n+1}$. 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 (5.1), 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.

5.3 Computation

The MPO problem (5.2) 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 (5.2) should be a factor H slower than solving the SPO problem (4.5). 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 [12, 11] 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.

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.

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 section 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 (5.1). 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 (5.1), 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 $z^{\mathrm{long}} = z_{t+T^{\mathrm{long}}}$ 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 $T^{\mathrm{med}} = 5$ and $T^{\mathrm{long}} = 21$, 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 (5.1) 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 (5.1), 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 $t + H - 1$.

6
Implementation

We have developed an open-source Python package `CVXPortfolio` [15] that implements the portfolio simulation and optimization concepts discussed in the paper. The package relies on Pandas [57] 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 [27] 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 chapter 7 we give some simple numerical examples that use `CVXPortfolio`.

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 section 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 section 4.4 and section 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 rebalancing.

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 chapter 3.

7

Examples

In this chapter 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.

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 [31].) We collect open-source market data from Quandl [73]. 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 [1] we approximate the daily volatility with a simple estimator, $(\sigma_t)_i = |\log(p_t^{\text{open}})_i - \log(p_t^{\text{close}})_i|$, where $(p_t^{\text{open}})_i$ and $(p_t^{\text{close}})_i$ 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.

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 $w^{\text{b}} = (\mathbf{1}/n, 0)$, *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 $w_1 = w^{\text{b}}$, and due to asset returns drifts from this weight vector. We periodically re-balance, which means using trade vector $z_t = w^{\text{b}} - w_t$. 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 $\bar{R}^{\text{a}}$ and active risk σ^{a} (defined in section 3.2), the annualized average transaction cost $\frac{250}{T}\sum_{t=1}^{T}\phi_t^{\text{trade}}(z_t)$, and the annualized average turnover $\frac{250}{T}\sum_{t=1}^{T}\|(z_t)_{1:n}\|_1/2$.

Table 7.1 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 7.1 shows, separately for the two portfolio sizes, the active risk versus

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

the transaction cost.

7.3 Single-period optimization

In this section we show a simple example of the single-period optimization model developed in chapter 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 section 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 [1]. 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

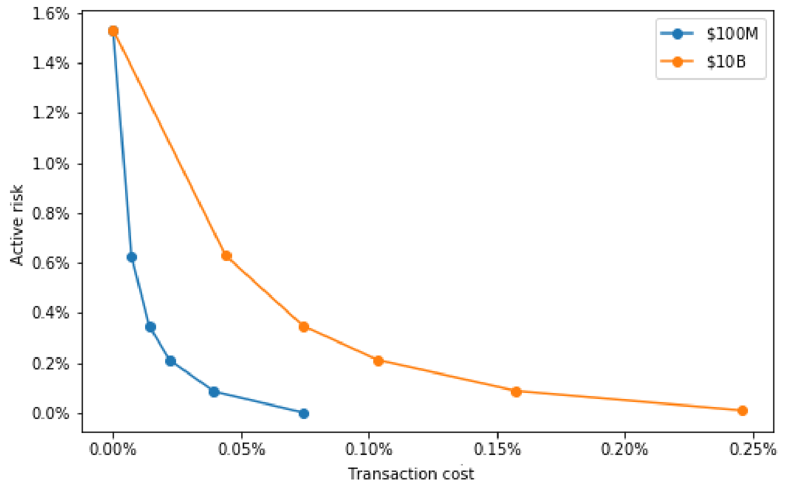

Figure 7.1: Active risk versus transaction cost, for the two initial portfolio sizes. The points on the lines correspond to re-balancing frequencies.

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

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

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

$$(\hat{r}_t)_{1:n} = \alpha \left((r_t)_{1:n} + \epsilon_t \right), \tag{7.1}$$

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 $\pm 0.3\%$. This corresponds to an information ratio $\sqrt{\alpha} \approx 0.15$, which is on the high end of what might be expected in practice [42].

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 section 4.8). We first perform a coarse grid search in the hyper-parameter space, testing all combinations of

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

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

Figure 7.2 shows mean excess portfolio return $\overline{R^{\text{e}}}$ versus excess volatility σ^{e} (defined in section 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.

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 7.3 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 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 7.4 as points in the risk-return plane. The Pareto optimal points, *i.e.*,

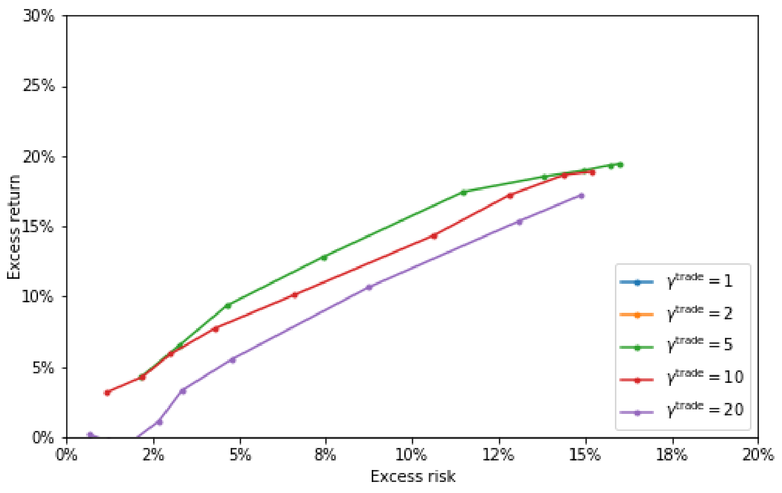

Figure 7.2: SPO example, coarse hyper-parameter grid search. (Some curves do not fit in the plot.)

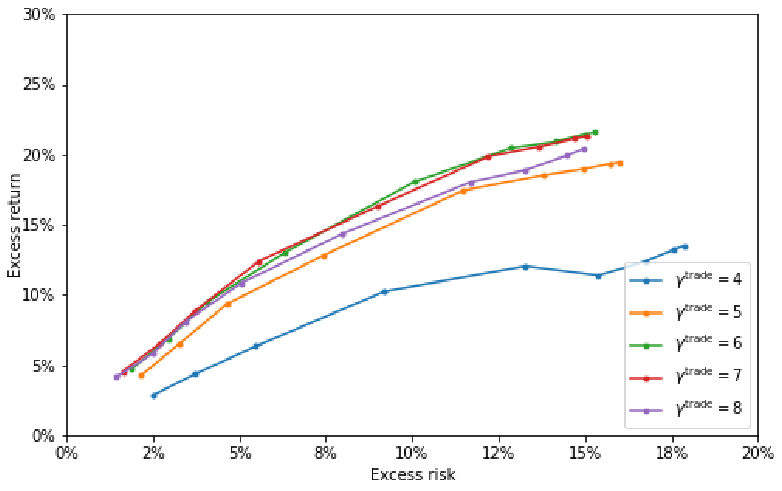

Figure 7.3: SPO example, fine hyper-parameter grid search.

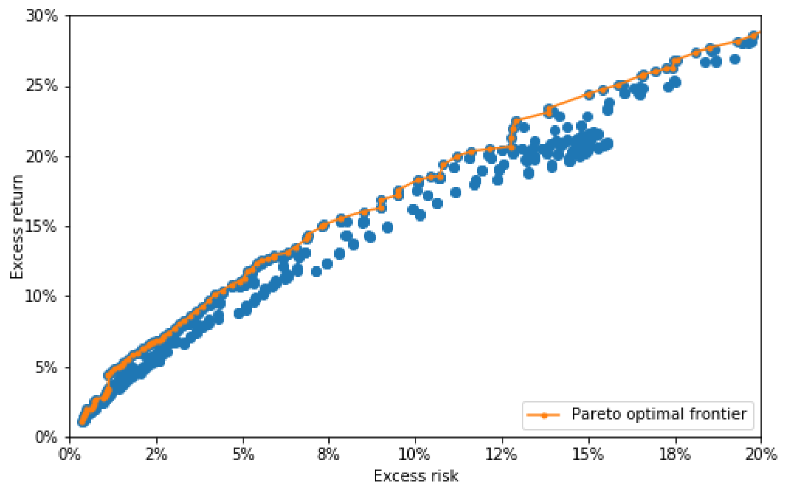

Figure 7.4: SPO example, grid search over 410 hyper-parameter combinations. The line connects the Pareto optimal points.

those with the lowest risk for a given level of return, are connected by a line. Table 7.2 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 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.

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

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.

7.4 Multi-period optimization

In this section we show the simplest possible example of the multi-period optimization model developed in chapter 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 section 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 (7.1). 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 7.5, 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.

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 7.6 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 7.7 shows the Pareto optimal frontiers for both cases. We see that the MPO method has a substantial advantage over the

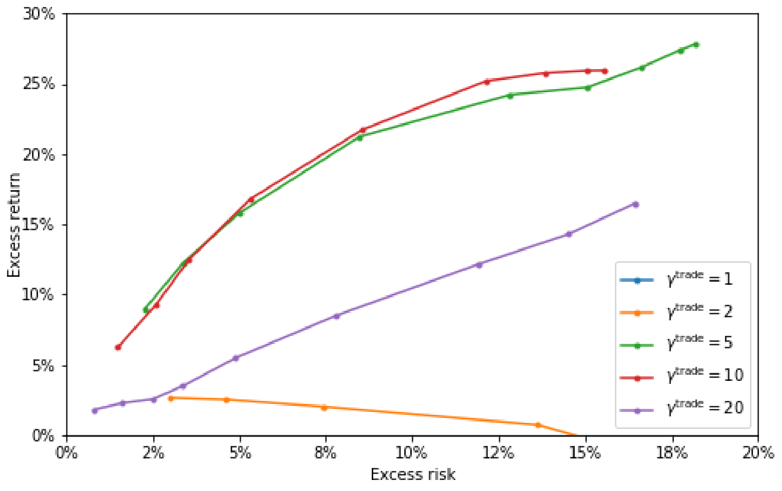

Figure 7.5: MPO example, coarse hyper-parameter grid search.

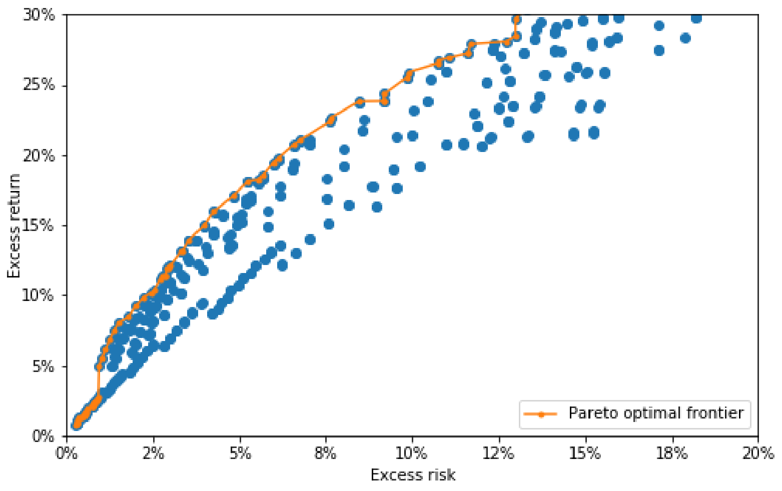

Figure 7.6: MPO example, grid search over 390 hyper-parameter combinations. The line connects the Pareto optimal points.

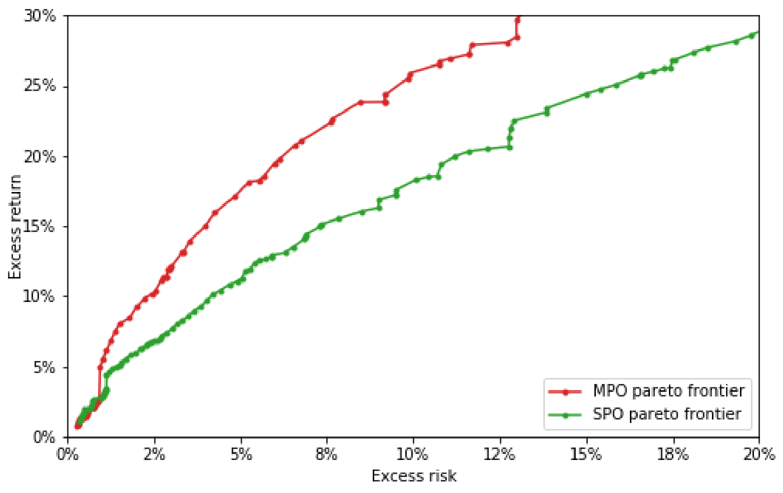

Figure 7.7: Pareto optimal frontiers for SPO and MPO.

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

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 7.8 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.

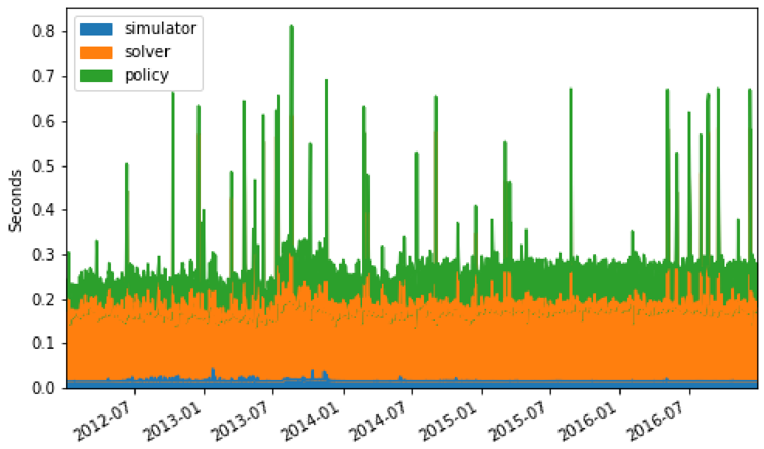

Figure 7.8: Execution time for each day for one SPO back-test.

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 [29] used in CVXPY with a custom solver, such as one based on operator-splitting methods [12], would result in an even more substantial speedup.

References

[1] R. Almgren. High frequency volatility. Available at `http://cims.nyu.edu/~almgren/timeseries/notes7.pdf`, 2009.

[2] R. Almgren and N. Chriss. Optimal execution of portfolio transactions. *Journal of Risk*, 3(2):5–39, 2001.

[3] C. Bacon. *Practical portfolio performance measurement and attribution.* Wiley, 2nd edition, 2008.

[4] D. Bailey, J. Borwein, M. Lopez de Prado, and Q. Zhu. The probability of backtest overfitting. *Journal of Computational Finance*, 20(4):39–69, 2017.

[5] R. Bellman. Dynamic programming and Lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

[6] 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.

[7] 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.

[8] 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.

[9] D. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 1995.

[10] 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.

[11] 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):1–72, 2014.

[12] 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.

[13] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[14] E. Busseti and S. Boyd. Volume weighted average price optimal execution. *arXiv preprint arXiv:1509.08503*, 2015.

[15] E. Busseti, S. Diamond, S. Boyd, and Blackrock. `CVXPortfolio`, 2017. Available at `https://github.com/cvxgrp/cvxportfolio`.

[16] E. Busseti, E. Ryu, and S. Boyd. Risk-constrained Kelly gambling. *Journal of Investing*, 25(3), 2016.

[17] J. Campbell, A. Lo, and A. MacKinlay. *The Econometrics of Financial Markets*, volume 1. Princeton University Press, 1997.

[18] J. Campbell and L. Viceira. *Strategic Asset Allocation: Portfolio Choice for Long-Term Investors.* Oxford University Press, 2002.

[19] 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.

[20] 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.

[21] 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.

[22] G. Constantinides. Multiperiod consumption and investment behavior with convex transactions costs. *Management Science*, 25(11):1127–1137, 1979.

[23] G. Cornuejols and R. Tütüncü. *Optimization Methods in Finance.* Cambridge University Press, 2006.

[24] M. Davis and A. Norman. Portfolio selection with transaction costs. *Mathematics of Operations Research*, 15(4):676–713, 1990.

[25] 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.

[26] V. DeMiguel, L. Garlappi, and R. Uppal. Optimal versus naive diversification: How inefficient is the $1/N$ portfolio strategy? *Review of Financial Studies*, 22(5):1915–1953, 2009.

[27] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[28] S. Diamond, R. Takapoui, and S. Boyd. A general system for heuristic solution of convex problems over nonconvex sets. *Optimization Methods and Software*, to appear, 2017.

[29] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pages 3071–3076, 2013.

[30] 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.

[31] E. Elton, M. Gruber, and C. Blake. Survivor bias and mutual fund performance. *The Review of Financial Studies*, 9(4):1097–1120, 1996.

[32] F. Fabozzi, D. Huang, and G. Zhou. Robust portfolios: Contributions from operations research and finance. *Annals of Operations Research*, 176(1):191–220, 2010.

[33] B. Fastrich, S. Paterlini, and P. Winker. Constructing optimal sparse portfolios using regularization methods. *Computational Management Science*, 12(3):417–434, 2015.

[34] 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. To appear.

[35] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning.* Springer, 2001.

[36] M. Frittelli and E. Gianin. Putting order in risk measures. *Journal of Banking & Finance*, 26(7):1473–1486, 2002.

[37] N. Gârleanu and L. Pedersen. Dynamic trading with predictable returns and transaction costs. *Journal of Finance*, 68(6):2309–2340, 2013.

[38] D. Goldsmith. Transactions costs and the theory of portfolio selection. *Journal of Finance*, 31(4):1127–1139, 1976.

[39] 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.

[40] M. Grant, S. Boyd, and Y. Ye. Disciplined convex programming. In *Global Optimization: From Theory to Implementation*, Nonconvex Optimization and its Applications, pages 155–210. Springer, 2006.

[41] R. Grinold. A dynamic model of portfolio management. *Journal of Investment Management*, 4(2):5–22, 2006.

[42] R. Grinold and R. Kahn. *Active Portfolio Management: A Quantitative Approach for Providing Superior Returns and Controlling Risk*. McGraw-Hill, 2nd edition, 1999.

[43] 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.

[44] 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.

[45] R. Jagannathan and T. Ma. Risk reduction in large portfolios: Why imposing the wrong constraints helps. *Journal of Finance*, 58(4):1651–1683, 2003.

[46] P. Jorion. International portfolio diversification with estimation risk. *Journal of Business*, 58(3):259–278, 1985.

[47] R. Kan and G. Zhou. Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis*, 42(3):621–656, 2007.

[48] J. Kelly, Jr. A new interpretation of information rate. *IRE Transactions on Information Theory*, 2(3):185–189, 1956.

[49] 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.

[50] W. Kwon and S. Han. *Receding Horizon Control: Model Predictive Control for State Models*. Springer-Verlag London, 2006.

[51] J. Li. Sparse and stable portfolio selection with parameter uncertainty. *Journal of Business & Economic Statistics*, 33(3):381–392, 2015.

[52] F. Lillo, J. Farmer, and R. Mantegna. Econophysics: Master curve for price-impact function. *Nature*, 421(6919):129–130, 2003.

[53] M. Lobo, M. Fazel, and S. Boyd. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research*, 152(1):341–365, 2007.

[54] H. Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[55] J. Mattingley and S. Boyd. CVXGEN: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, 2012.

[56] J. Mattingley, Y. Wang, and S. Boyd. Receding horizon control: Automatic generation of high-speed solvers. *IEEE Control Systems Magazine*, 31(3):52–65, 2011.

[57] W. McKinney. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython.* O'Reilly Media, 2012.

[58] R. Merton. Lifetime portfolio selection under uncertainty: The continuous-time case. *Review of Economics and Statistics*, 51(3):247–257, 1969.

[59] R. Merton. Optimum consumption and portfolio rules in a continuous-time model. *Journal of Economic Theory*, 3(4):373–413, 1971.

[60] A. Meucci. *Risk and Asset Allocation.* Springer, 2005.

[61] A. Meucci. Historical scenarios with fully flexible probabilities. *GARP Risk Professional*, pages 47–51, 2010.

[62] R. Michaud. The Markowitz optimization Enigma: Is 'optimized' optimal? *Financial Analysts Journal*, 45(1):31–42, 1989.

[63] C. Moallemi and M. Sağlam. Dynamic portfolio choice with linear rebalancing rules. *Journal of Financial and Quantitative Analysis*, 52:1247–1278, 2017.

[64] 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.

[65] R Narang. *Inside the Black Box: A Simple Guide to Quantitative and High Frequency Trading.* John Wiley & Sons, 2013.

[66] Y. Nesterov and A. Nemirovskii. *Interior-point Polynomial Algorithms in Convex Programming.* SIAM, 1994.

[67] P. Nystrup, H. Madsen, and E. Lindström. Dynamic portfolio optimization across hidden market regimes. Working paper, Technical University of Denmark, 2016.

[68] A. Obizhaeva and J. Wang. Optimal trading strategy and supply/demand dynamics. *Journal of Financial Markets*, 16(1):1–32, 2013.

[69] 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, June 2016.

[70] A. Perold. Large-scale portfolio optimization. *Management Science*, 30(10):1143–1160, 1984.

[71] A. Perold. The implementation shortfall: Paper versus reality. *Journal of Portfolio Management*, 31:106, 1998.

[72] W. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, 2007.

[73] Quandl. WIKI end-of-day data, 2016. Available at `https://www.quandl.com/data/WIKI`.

[74] P. Samuelson. Lifetime portfolio selection by dynamic stochastic programming. *Review of Economics and Statistics*, 51(3):239–246, 1969.

[75] W. Sharpe. Mutual fund performance. *Journal of Business*, 39(1):119–138, 1966.

[76] W. Sharpe. The arithmetic of active management. *Financial Analysts Journal*, 47(1):7–9, 1991.

[77] W. Sharpe. The Sharpe ratio. *The Journal of Portfolio Management*, 21(1):49–58, 1994.

[78] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, pages 267–288, 1996.

[79] 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.