Foundations and Trends® in Optimization
Vol. 1, No. 1 (2014) 1–72

DOI: 10.1561/2400000001

now
the essence of knowledge

Performance Bounds and Suboptimal Policies for Multi-Period Investment

Stephen Boyd
Stanford University
boyd@stanford.edu

Mark T. Mueller
Cambridge, MA
mark.t.mueller@mac.com

Brendan O'Donoghue
Stanford University
bodonoghue85@gmail.com

Yang Wang
Stanford University
yang1024@gmail.com

Contents

Abstract

We consider dynamic trading of a portfolio of assets in discrete periods over a finite time horizon, with arbitrary time-varying distribution of asset returns. The goal is to maximize the total expected revenue from the portfolio, while respecting constraints on the portfolio such as a required terminal portfolio and leverage and risk limits. The revenue takes into account the gross cash generated in trades, transaction costs, and costs associated with the positions, such as fees for holding short positions. Our model has the form of a stochastic control problem with linear dynamics and convex cost function and constraints. While this problem can be tractably solved in several special cases, such as when all costs are convex quadratic, or when there are no transaction costs, our focus is on the more general case, with nonquadratic cost terms and transaction costs.

We show how to use linear matrix inequality techniques and semidefinite programming to produce a quadratic bound on the value function, which in turn gives a bound on the optimal performance. This performance bound can be used to judge the performance obtained by any suboptimal policy. As a by-product of the performance bound computation, we obtain an approximate dynamic programming policy that requires the solution of a convex optimization problem, often a quadratic program, to determine the trades to carry out in each step. While we have no theoretical guarantee that the performance of our suboptimal policy is always near the performance bound (which would imply that it is nearly optimal) we observe that in numerical examples the two values are typically close.

S. Boyd, M. Mueller, B. O'Donoghue, Y. Wang. *Performance Bounds and Suboptimal Policies for Multi-Period Investment*. Foundations and Trends® in Optimization, vol. 1, no. 1, pp. 1–72, 2014.

DOI: 10.1561/2400000001.

1

Introduction

1.1 Overview

In this paper we formulate the discrete-time finite horizon time-varying multi-period investment problem as a stochastic control problem. By using state variables that track the *value* of the assets, instead of more traditional choices of states such as the number of shares or the fraction of total value, the stochastic control problem has *linear* (but random) dynamics. Assuming that the costs and constraints are convex, we arrive at a linear convex stochastic control problem.

This problem can be effectively solved in two broad cases. When there are no transaction costs, the multi-period investment problem can be reduced to solving a set of standard single-period investment problems; the optimal policy in this case is to simply rebalance the portfolio to a pre-computed optimal portfolio in each step. Another case in which the problem can be effectively solved is when the costs are quadratic and the only constraints are linear equality constraints. In this case standard dynamic programming (DP) techniques can be used to compute the optimal trading policies, which are affine functions of the current portfolio. We describe these special cases in more detail in §3.3 and §3.2. The problem is also tractable when the number of assets

is very small, say two or three, in which case brute force (numerical) dynamic programming can be used to compute an optimal policy.

Most problems of interest, however, include significant transaction costs, or include terms that are not well approximated by quadratic functions. In these cases, the optimal investment policy cannot be tractably computed. In such situations, several approaches can be used to find suboptimal policies, including approximate dynamic programming (ADP) and model predictive control (MPC). The performance of any suboptimal policy can be evaluated using Monte Carlo analysis, by simulation over many return trajectories. An obvious practical (and theoretical) question is, how suboptimal is the policy? In this paper we address this question.

Using linear matrix inequality (LMI) techniques widely used in control system analysis and design [18, 35, 80], we construct a (numerical) bound on the best performance that can be attained, for a given problem. The method requires the construction and solution of a semidefinite program (SDP), a convex optimization problem involving matrix inequalities. We can compare the bound on performance with the performance attained by any suboptimal policy; when they are close, we conclude that the policy is approximately optimal (and that the performance bound is nearly tight). Even when the performance bound and suboptimal policy performance are not close, we at least have a bound on how suboptimal our suboptimal policy can be.

The performance bound computation yields a quadratic approximation (in fact, underestimator) of the value functions for the stochastic control problem. These quadratic value function approximations can be used in an ADP policy, or as the terminal cost in an MPC policy. While we have no a priori guarantee that the gap between the performance bound and the performance of the ADP policy will always be small, simulations show that the ADP and MPC policies achieve performance that is often nearly optimal.

Our methods for computing the performance bound, as well as implementing the ADP and MPC suboptimal policies, rely on (numerically) solving convex optimization problems, for which there are efficient and reliable algorithms available [20, 72, 77, 73, 102]. The per-

formance bound computation requires solving SDPs [20, 95], which can be done using modern interior-point cone solvers such as SeDuMi or SDPT3 [88, 92, 94]. Parser-solvers such as CVX or YALMIP [40, 58] allow the user to specify the SDPs in a natural high-level mathematical description form, greatly reducing the time required to form and solve the SDPs. The SDPs that we solve involve T matrices of size $n \times n$, where n is the number of assets, and T is the trading period horizon. These SDPs can be challenging to solve (depending on n and T, of course), using generic methods; but this computation is done once, off-line, before trading begins.

Evaluating the ADP suboptimal policy in each period (*i.e.*, determining the trades to execute) requires solving a small and structured convex optimization problem with (on the order of) n scalar variables. Solving these problems using generic solvers might take seconds, or even minutes, depending on the problem size and types of constraints and objective terms. But recent advances have shown that if the solver is customized for the particular problem family, orders of magnitude speed up is possible [64, 65, 63, 62, 99]. This means that the ADP trading policies we design can be executed at time scales measured in milliseconds or microseconds for modest size problems (say, tens of assets), even with complex constraints. In addition, the trading policies we design can be tested and verified via Monte Carlo simulation very efficiently. For example, the simulation of the numerical examples of the ADP policies reported in this paper required the solution of around 50 million quadratic programs (QPs). These were solved in a few hours on a desktop computer using custom solvers generated by CVXGEN, a code generator for embedded convex optimization [64].

Evaluating the MPC policy also requires the solution of a structured convex optimization problem, with (on the order of) nT variables. If a custom solver is used, the computational effort required is approximately T times the effort required to evaluate the ADP policy. One major advantage of MPC is that it does not require any pre-computation; to implement the ADP policy, we must first solve a large SDP to find the approximate value functions. MPC can thus directly incorporate real-time signals such as changes in future return statistics.

1.2 Prior and related work

Portfolio optimization has been studied and used for more than 60 years. In this section our goal is to give a brief overview of some of the important research in this area, focussing on work related to our approach. Readers interested in a broader overview of the applications of stochastic control and optimization to economics and finance should refer to, *e.g.*, [1, 34, 45, 76, 90, 104].

Single-period portfolio optimization

Portfolio optimization was introduced by Markowitz in 1952 [61]. He formulated a single period portfolio investment problem as a quadratic optimization problem with an objective that trades off expected return and variance. Since this first work, many papers have extended the single period portfolio optimization framework. For example, Goldsmith [38] is one of the first papers to include an analysis of the effect of transaction costs on portfolio selection. Modern convex optimization methods, such as second-order cone programming (SOCP), are applied to portfolio problems with transaction costs in [57, 56]. Convex optimization methods have also been used to handle more sophisticated measures of risk, such as conditional value at risk (CVaR) [82, 50].

Dynamic multi-period portfolio optimization

Early attempts to extend the return-variance trade-off to multi-period portfolio optimization include [91, 71]. One of the first works on multi-period portfolio investment in a dynamic programming framework is by Merton [68]. In this seminal paper, the author considers a problem with one risky asset and one risk-free asset; at each continuous time instant, the investor chooses what proportion of his wealth to invest and what to consume, seeking to maximize the total utility of the wealth consumed over a finite time horizon. When there are no constraints or transaction costs, and under some additional assumptions on the investor utility function, Merton derived a simple closed-form expression for the optimal policy. In a companion paper [84], Samuelson derived the discrete-time analog of Merton's approach.

Constantinides [26] extended Samuelson's discrete-time formulation to problems with proportional transaction costs. In his paper, Constantinides demonstrated the presence of a convex 'no-trade cone'. When the portfolio is within the cone the optimal policy is not to trade; outside the cone, the optimal policy is to trade to the boundary of the cone. (We will see that the policies we derive in this paper have similar properties.) Davis and Norman [29] and Dumas and Lucian [33] derived similar results for the continuous-time formulation. In [28], the authors consider a specific multi-period portfolio problem in continuous time, where they derive a formula for the minimum wealth needed to hedge an arbitrary contingent claim with proportional transaction costs. More recent work includes [93, 23, 24]; in these the authors develop affine recourse policies for discrete time portfolio optimization.

Log-optimal investment

A different formulation for the multi-period problem was developed by Kelly [49], where it was shown that a log-optimal investment strategy maximizes the long-term growth rate of cumulative wealth in horse-race markets. This was extended in [21] to general asset returns and further extended to include all frictionless stationary ergodic markets in [3] and [27]. More recently, Iyengar [44] extended these problems to include proportional transaction costs.

Linear-quadratic multi-period portfolio optimization

Optimal policies for unconstrained linear-quadratic portfolio problems have been derived for continuous-time formulations by Zhou and Li [103], where the authors solve a continuous-time Riccati equation to compute the value function. In [53] this was extended to include a long-only constraint. Skaf and Boyd [87], and Gârleanu and Pederson [37], point out that the multi-period portfolio optimization problem with linear dynamics and convex quadratic objective can be solved exactly. For problems with more complex objective terms, such as proportional transaction costs, Skaf and Boyd use the value functions for an associated quadratic problem as the approximate value functions in an ADP

policy. In [43] the authors formulate a multi-period portfolio problem as a linear stochastic control problem, and propose an MPC policy.

Optimal execution

An important special case of the multi-period problem is the optimal execution problem, where we seek to execute a large block of trades while incurring as small a cost as possible. Bertsimas and Lo [16] model price impact, in which trading affects the asset prices, and derive an optimal trading policy using dynamic programming methods. Almgren and Chriss [4] address the optimal execution problem, including volatility of revenue. They show that the optimal policy can be obtained with additional restrictions on the price dynamics.

Performance bounds

In problems for which an optimal policy can be found, the optimal performance serves as a (tight) bound on performance. The present paper focuses on developing a numerical bound on the optimal performance for problems for which the optimal policy cannot be found.

Brown and Smith [22] compute a bound on optimal performance and derive a heuristic policy that achieves performance close to the bound. Their bound is given by the performance of an investor with perfect information about future returns, plus a clairvoyance penalty.

In [41], the authors construct an upper bound on a continuous time portfolio utility maximization problem with position limits. They do this by solving an unconstrained 'fictitious problem' which provides an upper bound on the value function of the original problem.

In [70], the authors describe a class of linear rebalancing policies for the discrete-time portfolio optimization problem. They develop several bounds, including a bound based on a clairvoyant investor and a bound obtained by solving an unconstrained quadratic problem.

Desai et al. [32] develop a bound for an optimal stopping problem, which is useful in a financial context for the pricing of American or Bermudan derivatives amongst other applications. The bound is derived from a dual characterization of optimal stopping problems as optimization problems over the space of martingales.

1.3 Outline

We structure our paper as follows. In chapter 2 we formulate a general multi-period investment problem as a linear convex stochastic control problem, using somewhat nontraditional state variables, and give examples of (convex) stage cost terms and portfolio constraints that arise in practical investment problems, as well as mentioning some nonconvex terms and constraints that do not fit our model. In chapter 3 we review the dynamic programming solution of the stochastic control problem, including the special case when the stage costs are convex quadratic. In chapter 4 we give our method for finding a performance bound in outline form; the full derivations are pushed to appendices A–C. We describe MPC in chapter 6. In chapter 7 we report numerical results for several examples, using both ADP and MPC trading policies.

2

Stochastic Control Formulation

2.1 Model

Portfolio. We manage a portfolio of n assets over a finite time horizon, which is divided into discrete time periods $t = 0, 1, \ldots, T$, which need not be uniformly spaced in real time. We let $x_t \in \mathbf{R}^n$ denote the portfolio (or vector of positions) at time t, where $(x_t)_i$ is the *dollar value* of asset i at the beginning of time period t, with $(x_t)_i < 0$ meaning a short position in asset i. For discussion on the relative merits of tracking the value of the assets rather than the number of units of each asset see, *e.g.*, [46, 48, 47, 85]. The dollar value is computed using the current *reference price* for each asset, which can differ from the current prices in an order book, such as the bid or ask price; a reasonable choice is the average of the bid and ask prices. We assume that the initial portfolio, x_0, is given. One of the assets can be a risk-free or cash account, as in a traditional formulation, but since we will be separately tracking cash that enters and leaves the portfolio, this is not needed.

Trading. We can buy and sell assets at the beginning of each time period. We let $u_t \in \mathbf{R}^n$ denote the dollar values of the trades: $(u_t)_i > 0$ means we buy asset i at the beginning of time period t and $(u_t)_i < 0$

means we sell asset i at the beginning of time period t. We define the post-trade portfolio as

$$x_t^+ = x_t + u_t, \quad t = 0, 1, \ldots, T,$$

which is the portfolio in time period t immediately after trading. For future reference, we note that $\mathbf{1}^T x_t$ is the total value of the portfolio (before trading), $\mathbf{1}^T u_t$ is the total cash we put into the portfolio to carry out the trades (not including transaction costs, discussed below), and $\mathbf{1}^T x_t^+$ is the total value of the post-trade portfolio, where $\mathbf{1}$ denotes the vector with all entries one.

Investment. The post-trade portfolio is held until the beginning of the next time period. The portfolio at the next time period is given by

$$x_{t+1} = R_{t+1} x_t^+, \quad t = 0, 1, \ldots, T-1, \tag{2.1}$$

where $R_{t+1} = \mathbf{diag}(r_{t+1}) \in \mathbf{R}^{n \times n}$ is the diagonal matrix of asset returns, and r_{t+1} is the vector of asset returns, from period t to period $t+1$. The return r_{t+1} is of course not known at the beginning of period t, so the choice of trades u_t must be made without knowledge of r_{t+1}. The dynamics (2.1) is *linear* (but unknown at time t); this is not the case when other state variables are chosen, such as the number of shares of each asset, or the fractional value of each asset in the portfolio.

Return model. We assume that r_t are independent random (vector) variables, with known distributions, with mean and covariance

$$\mathbf{E}\, r_t = \overline{r}_t, \qquad \mathbf{E}(r_t - \overline{r}_t)(r_t - \overline{r}_t)^T = \Sigma_t, \quad t = 1, \ldots, T.$$

Our assumption of independence of returns in different periods means that our formulation does not handle phenomena such as momentum or mean-reversion, or more sophisticated models for variance such as GARCH. The returns are typically nonnegative, but we will not use this assumption in the sequel. Time variation of the return distribution can be used to model effects such as predictable time variation in volatility, or non-uniformly spaced time periods. We note for future use that the total value of the portfolio after the investment period (or equivalently, at the beginning of the next time period) is given by $\mathbf{1}^T x_{t+1} = r_{t+1}^T x_t^+$.

Trading policy. The trades are determined in each period by the trading policy $\phi_t : \mathbf{R}^n \to \mathbf{R}^n$:

$$u_t = \phi_t(x_t), \quad t = 0, \ldots, T.$$

Thus, at time t, the trades u_t depend only on the portfolio positions x_t. (For the problem we consider, it can be shown that there is no advantage to including past state values, or past returns, in the trading policy; see, *e.g.*, [10, 13, 14].) For fixed $\phi_0, \ldots, \phi_T$, the portfolio and trades $x_0, \ldots, x_T$ and $u_0, \ldots, u_T$ become random variables. Since x_0 is given, we can assume that ϕ_0 is a constant function.

Cash in. The amount of cash we put into the portfolio at the beginning of time period t is given by $\ell_t(x_t, u_t)$, $t = 0, \ldots, T$, where $\ell_t : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R} \cup \{\infty\}$ is a closed convex function we will describe in detail in §2.2; it includes the gross cost of (or revenue from) trades, transaction costs, and other costs associated with holding the portfolio. We will refer to $\ell_t(x_t, u_t)$ as the *stage cost* for period t, and $-\ell_t(x_t, u_t)$ as the *revenue* or income from the portfolio in time period t. We refer to $\ell_T(x_T, u_T)$ as the *terminal cost*. Infinite values of $\ell_t(x_t, u_t)$ are used to encode hard constraints on the portfolio, described in detail in §2.2, such as leverage or risk limits or a required final portfolio.

Time variation in the stage cost functions can be used to model many effects. Simple examples include handling terminal costs and constraints, and including a discount factor to take into account the time value of the revenue in different periods. We can also model predictable variation in transaction cost parameters, or enforce leverage or risk limits that vary with time.

Objective. Our overall objective is the expected total cost,

$$J = \mathbf{E} \sum_{t=0}^{T} \ell_t(x_t, u_t),$$

where the expectation is over the returns sequence $r_1, \ldots, r_T$, and $u_t = \phi_t(x_t)$. (We assume the expectations exist.) Thus $-J$ is the total expected revenue from the portfolio. If a discount factor has been

incorporated into the stage cost functions, $-J$ is the expected present value of the revenue stream.

Stochastic control. The optimal investment problem is to determine a trading policy ϕ_t, $t = 0, \ldots, T$, that minimizes J, *i.e.*, maximizes the expected total revenue. We let $J^\star$ denote the optimal value of J, and we let $\phi_t^\star$, $t = 0, \ldots, T$, denote an optimal policy. This is a stochastic control problem with linear dynamics and convex stage cost. The data for the problem is the distribution of r_t, the stage cost functions ℓ_t, and the initial portfolio x_0.

Our goal here is not to address the many technical conditions arising in stochastic control (some of which can have ramifications in practical problems). For example, the problem may be unbounded below, *i.e.*, we can find policies that make the total expected revenue arbitrarily negative, in which case $J^\star = -\infty$. As another example, the optimal value $J^\star$ can be finite, but a policy that achieves the optimal value does not exist. For discussion of these and other pathologies, and more technical detail, see [10, 13, 14].

Real-time signals. Our model assumes that the return statistics (but of course not the returns) are known ahead of time, over the whole trading period. Indeed, we will see that the optimal trading policies, as well as our suboptimal trading policies, depend on all values of $\overline{r}_t$ and Σ_t. Thus, our formal model does *not* allow for the use of *real-time signals* in the return distribution model, *i.e.*, the use of real-time data, financial or non-financial, to update or predict the future return distributions during the trading period.

Signals can be formally incorporated into the model, for example, by assuming that the returns are independent, given the current signal values. While much of what we discuss in this paper can be generalized to this setting, it is far more complex, so we defer it to a future paper. As a practical matter, however, we note that the trading algorithms we describe can readily incorporate the use of signals, at least informally. We simply re-compute the policies whenever our estimates of the future return statistics change due to signals.

2.2 Stage cost function

In this section we detail some of the possible forms that the stage cost function can take. Its general form is

$$\ell_t(x,u) = \begin{cases} \mathbf{1}^T u + \psi_t(x,u) & x+u \in \mathcal{C}_t \\ \infty & \text{otherwise,} \end{cases}$$

where $\mathcal{C}_t \subseteq \mathbf{R}^n$ is the post-trade portfolio constraint set, and $\psi_t : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R}$ is a cost, with units of dollars, for period t. We assume that ψ_t and $\mathcal{C}_t$ are closed convex, with $\mathcal{C}_t$ nonempty. The term $\mathbf{1}^T u$ is the gross cash we put into the portfolio due to the trades, without transaction costs. The term $\psi_t(x,u)$ represents any additional amount we pay for the portfolio and trades. The *post-trade constraint set* $\mathcal{C}_t$ will capture constraints on the post-trade portfolio. We will refer to ψ_t as the *transaction cost* function, even though it can also include terms related to positions. The transaction cost ψ_t is typically nonnegative, but we do not need to assume this in the sequel; we intepret $-\psi_t(x_t,u_t)$ as additional revenue when $\psi_t(x_t,u_t)$ is negative.

2.3 Post-trade constraints

The post-trade constraint set (or more simply, the constraint set) $\mathcal{C}_t$ defines the set of acceptable post-trade portfolios. Since $\mathcal{C}_t$ is nonempty, it follows that for any value of x_t, we can find a u_t for which

$$x_t^+ = x_t + u_t \in \mathcal{C}_t.$$

We impose explicit constraints only on the post-trade portfolio x_t^+, and not on the portfolio itself. One reason is that we have control over the post-trade portfolio, by buying and selling (*i.e.*, through u_t); whereas the (pre-trade) portfolio x_t is determined by the (random) return r_t in the previous period, and is not directly under our control. In many cases a constraint on the post-trade portfolio implies a similar constraint on the portfolio, with some reasonable assumption about the return distribution (such as nonnegativity). For example, if we impose a nonnegativity (long-only) constraint on x_t^+ and the returns are always nonnegative, then the portfolio values x_t are also nonnegative.

We now describe examples of portfolio constraints that are useful in practice; we can of course impose any number of these, taking $\mathcal{C}_t$ to be the intersection.

Position limits. The portfolio may be subject to constraints on the post-trade positions, such as minimum and maximum allowed positions for each asset:

$$x_t^{\min} \leq x_t^+ \leq x_t^{\max},$$

where the inequalities are elementwise and $x^{\min}$ and $x^{\max}$ are given (vectors of) minimum and maximum asset holdings, given in dollars. For this constraint $\mathcal{C}_t$ is a box in $\mathbf{R}^n$. One special case is $x_t^+ \geq 0$, which means our (post-trade) portfolio can include only long positions. This corresponds to $\mathcal{C}_t = \mathbf{R}_+^n$, where $\mathbf{R}_+$ is the set of nonnegative reals.

Position limits can also be expressed relative to the total portfolio value; for example,

$$x_t^+ \leq (\mathbf{1}^T x_t^+)\alpha_t,$$

with $\alpha_t \in \mathbf{R}^n$ with positive entries, requires the value in asset i does not exceed the fraction $(\alpha_t)_i$ of the total portfolio value. This constraint is convex, with $\mathcal{C}_t$ a polyhedron.

Some simple position limits that are not convex, and so cannot be used in our model, include minimum nonzero positions, or the restriction that assets are held in integer multiples of some block size (such as 100 shares).

Total value minimum. We can require that the post-trade portfolio maintain minimum total value $v_t^{\min}$, which is the constraint $\mathbf{1}^T x_t^+ \geq v_t^{\min}$. When the pre-trade portfolio value falls $\mathbf{1}^T x_t$ below $v_t^{\min}$ (say, because of a very unfavorable return in the last period), we are required to put cash into the portfolio to bring the total value back up to $v_t^{\min}$. Several other portfolio constraints described below indirectly impose a minimum post-trade portfolio value, typically, zero. This constraint corresponds to $\mathcal{C}_t$ being a halfspace.

Terminal portfolio constraints. The simplest terminal constraint is $x_T^+ = x^{\text{term}}$, where x^{term} is a given portfolio, *i.e.*, $\mathcal{C}_T = \{x^{\text{term}}\}$. (This

constraint is the same as fixing the last trade to be $u_T = x^{\text{term}} - x_T$.) In this case our multi-period trading problem is the *optimal execution problem*: We seek the policy that starts from the given initial portfolio at $t = 0$, achieves the given final (post-trade) portfolio at $t = T$, while minimizing expected cost. When x_0 is nonzero and $x^{\text{term}} = 0$, the problem is to 'unwind' the positions x_0, *i.e.*, to cash out of the market over the given period, maximizing expected revenue. When $x_0 = 0$ and x^{term} is some given portfolio, the problem is to make investments over the period to achieve a desired portfolio, at minimum cost. A special case is $x_0 = x^{\text{term}} = 0$, which means the trading starts and ends with no holdings.

Short position and leverage limits. In the simplest case we impose a fixed limit on the total short position in the post trade portfolio:

$$\mathbf{1}^T (x_t^+)_- \leq S_t^{\max},$$

where $(x)_- = \max(-x, 0)$ and $S_t^{\max} \geq 0$ is the maximum allowed total short position. This constraint is convex (in fact, polyhedral), since the lefthand side is a piecewise linear convex function.

We can also limit the total short position relative to the total value of the post-trade portfolio:

$$\mathbf{1}^T (x_t^+)_- \leq \eta_t \mathbf{1}^T x_t^+, \tag{2.2}$$

where $\eta_t \geq 0$, sets the maximum ratio of total short position to total portfolio value. This constraint is convex (in fact, polyhedral), since the lefthand side is a piecewise linear convex function, and the righthand side is a linear function of x_t^+. This limit requires the total post-trade value to be nonnegative; for $\eta_t = 0$ it reduces to a long-only constraint.

The limit (2.2) can be written in several other ways, for example, it is equivalent to

$$\mathbf{1}^T (x_t^+)_- \leq \frac{\eta_t}{1 + \eta_t} \mathbf{1}^T (x_t^+)_+,$$

where $(x)_+ = \max(x, 0)$. In other words, we limit the ratio of the total short to total long positions.

A traditional leverage limit, which limits the ratio of the total short plus total long positions to the total assets under management, can be expressed as

$$\|x_t^+\|_1 = \mathbf{1}^T(x_t^+)_- + \mathbf{1}^T(x_t^+)_+ \leq L_t^{\max} A,$$

where $L_t > 0$ is the leverage limit, and $A > 0$ is the total value of the assets under management. (As a variation on this, we can replace A with $A + \mathbf{1}^T x_t^+$.)

Sector exposure limits. We can include the constraint that our post-trade portfolio has limited exposure to a set of economic sectors (such as manufacturing, energy, technology) or factors (determined by statistical analysis of returns). These constraints are expressed in terms of a matrix $F_t \in \mathbf{R}^{k\times n}$, called the *factor loading matrix*, that relates the portfolio to a vector of k sector exposures: $(F_t x_t^+)_j$ is the sector exposure to sector j. A sector exposure limit can be expressed as

$$s_t^{\min} \leq F_t x_t^+ \leq s_t^{\max},$$

where $s_t^{\min}$ and $s_t^{\max}$ are (vectors of) given lower and upper limits on sector exposures. A special case is sector neutrality, which is the constraint

$$(F_t x_t^+)_j = 0 \tag{2.3}$$

(for sector j neutrality). Sector exposure limits are linear inequality constraints, and sector neutrality constraints are linear equality constraints on x_t^+.

Sector limits can also be expressed relative to the total portfolio value, as in

$$F_t x_t^+ \leq (\mathbf{1}^T x_t^+)\alpha_t,$$

which limits the (positive) exposure in sector i to no more than the fraction $(\alpha_t)_i$ of the total portfolio value.

Concentration limit. A concentration limit requires that no more than a given fraction of the portfolio value can be held in some given fraction (or just a specific number p) of assets. This can be written as

$$\sum_{i=1}^{p} (x_t^+)_{[i]} \leq \beta_t \mathbf{1}^T x_t^+,$$

where $\beta_t > 0$, and the notation $a_{[i]}$ refers to the ith largest element of the vector a. The lefthand side is the sum of the p largest post-trade positions, which is a convex function of x_t^+, and the righthand side is a linear function, so this constraint is convex (in fact, polyhedral) [20, §3.2.3]. This constraint implies that the post-trade portfolio has nonnegative value.

Variance and standard deviation risk limits. We can limit the risk in the post-trade portfolio, using the traditional measure of risk based on variance of post-trade portfolio value over the next period, that is, the variance given x_t and u_t,

$$\mathbf{var}(\mathbf{1}^T x_{t+1} \mid x_t^+) = (x_t^+)^T \Sigma_{t+1} x_t^+.$$

This is a (convex) quadratic function of x_t^+. A simple risk limit can then be expressed as

$$(x_t^+)^T \Sigma_{t+1} x_t^+ \leq \gamma_t,$$

where $\gamma_t > 0$ is a given maximum variance (with units of dollars squared). This constraint is convex; in fact, the associated constraint set is an ellipsoid.

The risk limit above is fixed (for each t). By working with the standard deviation, we can develop risk limits that scale with total portfolio value. This allows us to limit the risk (measured in standard deviation, which has units of dollars) to some fraction of the post-trade portfolio value,

$$((x_t^+)^T \Sigma_{t+1} x_t^+)^{1/2} = \|\Sigma_{t+1}^{1/2} x_t^+\|_2 \leq \delta_t \mathbf{1}^T x_t^+,$$

where $\delta_t > 0$ (and is unitless, since left and righthand sides have units of dollars). These are convex constraints, in fact, second-order cone (SOC) constraints [20, §4.4.2]. They are also homogeneous constraints; that is, the allowed risk scales with the value of the post-trade portfolio. These constraints require the post-trade portfolio value to be nonnegative.

More sophisticated risk limits. There are many risk measures that are more sophisticated than variance, but are also convex in x_t^+, and therefore fit into our framework. For example, suppose that we do not

know the return covariance matrix, but are willing to assume it lies in the convex hull of a set of given covariance matrices $\Sigma^1, \ldots, \Sigma^q$. (We can think of these as the return covariance under q market regimes, or under q possible scenarios.) We can impose a constraint that limits our return variance, under all such scenarios, as

$$(x_t^+)^T \Sigma^i x_t^+ \leq \gamma_t, \quad i = 1, \ldots, q.$$

The associated constraint set is the intersection of ellipsoids (and therefore convex).

Moving beyond quadratic risk measures, we can limit the expected value of a function of period loss,

$$\mathbf{E}(\chi(\mathbf{1}^T x_{t+1} - \overline{r}_{t+1}^T x_t^+) \mid x_t^+) \leq \gamma_t,$$

where $\chi : \mathbf{R} \to \mathbf{R}$ is convex (and typically decreasing). For $\chi(v) = v^2$ we recover quadratic risk; for $\chi(v) = (v)_-^2$, this is downside quadratic risk; for $\chi(v) = (v - v_0)_-$, it is related to conditional value at risk (CVaR) [82, 5]. For such measures the constraint is convex, but not easily handled; typically, one has to resort to Monte Carlo methods to evaluate the risk measure, and stochastic optimization to handle them in an optimization setting; see, *e.g.*, [86, 9, 17, 79, 25].

2.4 Transaction and position costs

In this section we describe some of the possible forms that the transaction (and position) cost function ψ can take. Any number of the terms below can be combined by simple addition, which preserves convexity.

Broker commission. When trades are executed through a broker, we can be charged a commission for carrying out the trade on our behalf. In a simple model this cost is proportional to the total trade volume, which gives

$$\psi_t(x_t, u_t) = \kappa_t^T |u_t|,$$

where $\kappa_t \geq 0$ is the vector of commission rates, and the absolute value is elementwise. When the commission rates are equal across assets, this

reduces to $\psi_t(x_t, u_t) = \kappa_t \|u_t\|_1$, where $\kappa_t \geq 0$ is a scalar. A more general form charges different rates for buying and selling:

$$\psi_t(x_t, u_t) = (\kappa_t^{\text{buy}})^T (u_t)_+ + (\kappa_t^{\text{sell}})^T (u_t)_-,$$

where κ_t^{buy} and κ_t^{sell} are nonnegative vectors of buying and selling commission rates. These are all convex functions.

Some broker commission charges, such as charging a flat fee for any (nonzero) amount of trading in an asset, or offering a relative discount for large orders, are nonconvex; see, *e.g.*, [56] for methods for dealing with these nonconvex terms using convex optimization.

Bid-ask spread. The prices at which we buy or sell an asset are different, with the difference referred to as the bid-ask spread; the *mid-price* is the average of the buy and sell prices. If we use the mid-price for each asset as our reference price for converting shares held into our portfolio vector x_t, this means that we sell each asset for a price lower than the mid-price and buy each asset for a price higher than the mid-price. We can model this phenomenon as an additional transaction cost of the form

$$\psi_t(x_t, u_t) = \kappa_t^T |u_t|,$$

where $(\kappa_t)_i$ is one-half the bid-ask spread for asset i. (This has the same form as the broker commission described above.)

Price impact. When a large order is filled, the price moves against the trader as orders in the book are filled. This is known as price impact. A simple model for price-impact cost is quadratic,

$$\psi_t(x_t, u_t) = s_t^T u_t^2,$$

where $(s_t)_i \geq 0$, and the square above is elementwise. Many other models can be used, such as a 3/2 power transaction cost, $\psi_t(x_t, u_t) = s_t^T |u_t|^{3/2}$, or a general convex piecewise linear transaction cost, which models the depth of the order book at each price level. These are all convex functions of u_t.

We are *not* modeling multi-period price impact, which is the effect of a large order in one period affecting the price in future periods.

Borrowing/shorting fee. When going short, an asset must be borrowed from a third party, such as a broker, who will typically charge a fee for this service proportional to the value of the assets borrowed per period. This gives the (convex) position cost

$$\psi_t(x_t, u_t) = c_t^T (x_t^+)_-,$$

where $(c_t)_i \geq 0$ is the fee rate, in period t, for shorting asset i.

More generally we can pay a fee for our total short position, perhaps as a default insurance policy premium. Such a cost is an increasing convex function of $\mathbf{1}^T(x_t^+)_-$, and is therefore also convex in x_t.

Risk penalty. We have described risk limits as constraints in §2.3 above. We can also take risk into account as a real or virtual additional charge that we pay in each period. (For example, the charge might be paid to an internal risk management desk.) In this case $\ell_t(x_t, u_t)$ is the risk-adjusted cost in time period t. Indeed, traditional portfolio optimization is formulated in terms of maximizing risk-adjusted return (the negative of risk-adjusted cost).

The traditional risk adjustment charge is proportional to the variance of the next period total value, given the current post-trade position, which corresponds to

$$\psi_t(x_t, u_t) = \lambda_t \, \mathbf{var}(\mathbf{1}^T x_{t+1} \mid x_t^+) = \lambda_t (x_t^+)^T \Sigma_{t+1} x_t^+,$$

where $\lambda_t \geq 0$ is called the risk aversion parameter. This traditional charge is not easy to interpret, since λ_t has units of inverse dollars. (The main appeal of using variance is analytical tractability, which is not an issue when convex optimization beyond simple least-squares is used.)

We can levy a charge based on standard deviation rather than variance, which gives

$$\psi_t(x_t, u_t) = \lambda_t \|\Sigma_{t+1}^{1/2} x_t^+\|_2,$$

where in this case $\lambda_t \geq 0$ is dimensionless, and has the interpretation of standard deviation cost rate, in dollar cost per dollar standard deviation. More generally, the charge can be any increasing convex function of $\|\Sigma_{t+1}^{1/2} x_t^+\|_2$, which includes both the standard deviation and variance charges as special cases. All such functions are convex.

2.5 Quadratic and QP-representable stage cost

Here we describe two special forms for the stage cost, for future use.

Quadratic. An important special case occurs when ℓ_t is (convex) quadratic, possibly including linear equality constraints. This means that the transaction cost is quadratic,

$$\psi_t(x,u) = (1/2)\begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} A_t & B_t & a_t \\ B_t^T & C_t & c_t \\ a_t^T & c_t^T & d_t \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix},$$

where

$$\begin{bmatrix} A_t & B_t \\ B_t^T & C_t \end{bmatrix} \succeq 0,$$

(meaning, the matrix on the lefthand side is symmetric positive semidefinite), and the constraint set is

$$\mathcal{C}_t = \{x \mid G_t x = h_t\}.$$

When the stage cost has this form, we refer to the multi-period portfolio optimization problem as being quadratic. The quadratic problem is quite limited; it can include, for example, a terminal portfolio constraint, a quadratic risk penalty, and a quadratic transaction cost. The other constraints described above yield a problem that is not quadratic.

QP-representable. We say the stage cost is QP-representable if ℓ_t is convex quadratic plus a convex piecewise linear function, possibly including linear equality and inequality constraints. Such a function can be minimized by transformation to a QP, using standard techniques (described, *e.g.*, in [20, 72], and employed in convex optimization systems such as CVX [40], YALMIP [58], and CVXGEN [64]).

Many of the transaction cost terms and constraints described above are QP-representable, including leverage limits, sector exposure limits, concentration limits, broker fees, bid-ask spread, and quadratic risk penalty. Quadratic and second-order cone risk limits are not QP-representable, but can be represented as a second-order cone problem (SOCP) [20, §4.4], [57].

3

Optimal Policy

3.1 Dynamic programming

In this section we briefly review the dynamic programming characterization of the solution to the stochastic control problem. For more detail, see, *e.g.*, [10, 13, 6, 83, 31, 69].

The (Bellman) value functions $V_t : \mathbf{R}^n \to \mathbf{R}$, $t = 0, \ldots, T+1$ are defined by $V_{T+1} = 0$, and the backward recursion

$$V_t(x) = \inf_u \left(\ell_t(x,u) + \mathbf{E}\, V_{t+1}(R_{t+1}(x+u))\right), \quad t = T, \ldots, 0, \tag{3.1}$$

where the expectation is over the return r_{t+1}. We can write this recursion compactly as

$$V_t = \mathcal{T}_t V_{t+1}, \quad t = T, \ldots, 0, \tag{3.2}$$

where $\mathcal{T}_t$ is the Bellman operator, defined as

$$(\mathcal{T}_t h)(x) = \inf_u \left(\ell_t(x,u) + \mathbf{E}\, h(R_{t+1}(x+u))\right),$$

for $h : \mathbf{R}^n \to \mathbf{R}$.

An optimal policy can be expressed via the value functions as

$$\phi_t^\star(x) \in \operatorname*{argmin}_u \left(\ell_t(x,u) + \mathbf{E}\, V_{t+1}(R_{t+1}(x+u))\right), \tag{3.3}$$

and the optimal cost is given by

$$J^\star = V_0(x_0).$$

This general dynamic programming solution method is not a practical algorithm, since we do not in general have a method to represent V_t, let alone effectively carry out the expectation and partial minimization operations. In the quadratic case (described below), however, the Bellman iteration can be explicitly carried out, yielding an explicit form for V_t and $\phi_t^\star$.

Monotonicity. For future use we note that the Bellman operator $\mathcal{T}_t$ is monotonic: for any $f : \mathbf{R}^n \to \mathbf{R}$ and $g : \mathbf{R}^n \to \mathbf{R}$,

$$f \leq g \quad \Longrightarrow \quad \mathcal{T}_t f \leq \mathcal{T}_t g, \tag{3.4}$$

where the inequalities are interpreted pointwise [10, 13].

Convexity. The value functions $V_0, \ldots, V_{T+1}$ are convex, which we can show by a (backward) recursion. We first observe that $V_{T+1} = 0$ is convex. We will show that the Bellman operators $\mathcal{T}_t$ preserve convexity; this will show that all V_t are convex functions.

To show that the Bellman operator $\mathcal{T}_t$ preserves convexity, suppose h is convex. Then, for fixed R_{t+1}, $h(R_{t+1}(x+u))$ is a convex function of (x, u); since expectation preserves convexity, we conclude that $\mathbf{E}\, h(R_{t+1}(x+u))$ is convex in (x, u). The stage cost $\ell_t(x, u)$ is convex, so $\ell_t(x, u) + \mathbf{E}\, h(R_{t+1}(x+u))$ is convex in (x, u). Finally, partial minimization of this function (in this case, over u) preserves convexity, so we conclude that $\mathcal{T}_t h$ is convex. (For more on the convexity rules used above, see, *e.g.*, [20, Ch. 3].)

One implication of the convexity of V_t is that evaluation of the optimal policy (3.3) requires solving a convex optimization problem.

3.2 Quadratic case

When the problem is quadratic, *i.e.*, ℓ_t are quadratic functions plus (possibly) linear equality constraints, we can effectively compute V_t,

which are also (convex) quadratic functions. We give the argument in general form here, with more detail given in the appendices.

The argument is similar to that for convexity of V_t, with the attribute 'quadratic' substituted for 'convex'. We note that V_{T+1} is a quadratic function, and we will show that the Bellman operators preserve quadratic functions. It follows that all V_t are quadratic. Moreover we can explicitly compute the coefficients of the quadratic functions, so the method can be implemented.

To show that the Bellman operator $\mathcal{T}_t$ preserves convex quadratic functions, suppose h is convex quadratic. Then, for fixed R_{t+1}, $h(R_{t+1}(x+u))$ is a convex quadratic function of (x, u); since expectation preserves convex quadratic functions, we conclude that $\mathbf{E}\, h(R_{t+1}(x+u))$ is convex quadratic in (x, u). (To explicitly compute its coefficients requires knowledge of $\overline{r}_{t+1}$ and Σ_{t+1}, but no other attribute of the return distribution.) A detailed derivation, including fomulas for the coefficients, is given in appendix A.

The stage cost $\ell_t(x, u)$ is assumed convex quadratic (plus linear equality constraints), so $\ell_t(x, u) + \mathbf{E}\, h(R_{t+1}(x+u))$ is convex quadratic in (x, u) (since convex quadratic functions are closed under addition). Finally, partial minimization of a convex quadratic, possibly subject to linear equality constraints, preserves convex quadratic functions, so we conclude that $\mathcal{T}_t h$ is convex quadratic. See appendix B for a detailed derivation, and explicit formulas for the coefficients.

The optimal trading policies require the minimization of a convex quadratic function of (x, u) over u (possibly, with equality constraints). The minimizer of a convex quadratic function of (x, u), subject to linear equality constraints, can be expressed explicitly as an affine function of x (see appendix B). Thus, the optimal trading policies have the form

$$\phi_t^\star(x) = J_t x + k_t, \quad t = 0, \ldots, T, \tag{3.5}$$

where $J_t \in \mathbf{R}^{n\times n}$ and $k_t \in \mathbf{R}^n$ can be explicitly computed. (We can without loss of generality take $J_0 = 0$.) This is one of the few cases for which the value functions (and hence, the optimal policy), can be explicitly computed. The coefficients J_t and k_t in the optimal policy depend on the coefficients in the stage cost functions ℓ_τ, as well as $\overline{r}_\tau$ and Σ_τ, for $\tau = t, \ldots, T$.

3.3 No transaction cost case

Here we consider another special case in which we can solve the stochastic control problem. Suppose that $\psi_t(x,u)$ is a function of $x^+ = x+u$, which we write (with some abuse of notation) as $\psi_t(x^+)$. In other words, we exclude transaction costs (broker fees, bid-ask spread, and price impact type terms); the only stage costs we have are functions of the post-trade portfolio. (We have already assumed that the constraints have this form.)

The stage cost then has the form

$$\ell_t(x,u) = \mathbf{1}^T x^+ - \mathbf{1}^T x + \psi_t(x^+) + I_t(x^+),$$

where I_t is the indicator function of $\mathcal{C}_t$. The objective can then be written as

$$\begin{aligned} J &= \mathbf{E}\sum_{t=0}^{T} \ell_t(x_t,u_t) \\ &= \sum_{t=0}^{T} \mathbf{E}\left(\mathbf{1}^T x_t^+ + \psi_t(x_t^+) + I_t(x_t^+)\right) - \mathbf{1}^T x_0 - \mathbf{E}\sum_{t=1}^{T} r_t^T x_{t-1}^+ \\ &= -\mathbf{1}^T x_0 + \sum_{t=0}^{T} \mathbf{E}\left((\mathbf{1}-r_{t+1})^T x_t^+ + \psi_t(x_t^+) + I_t(x_t^+)\right), \end{aligned}$$

where we take $r_{T+1} = 0$. Thus, our problem is equivalent to a stochastic control problem with the modified stage cost

$$\tilde{\ell}_t(x,u) = (\mathbf{1}-r_{t+1})^T x^+ + \psi_t(x^+) + I_t(x^+),$$

which is a function only of $x^+ = x+u$. This stochastic control problem has an associated value function, which we denote by $\tilde{V}$, which satisfies (3.1) with the modified stage cost function.

Using the modified stage cost, the minimization in the optimal policy (3.3) is over a function of x^+. To minimize a function of $x^+ = x+u$ over u, we simply minimize the function over x^+ to find $x^{+\star}$, and then take $u = x^{+\star} - x$. For the optimal policy (3.3), we conclude that $\phi_t^\star(x_t) = x_t^{+\star} - x_t$, where $x_t^{+\star}$ minimizes

$$(\mathbf{1}-\overline{r}_{t+1})^T x_t^+ + \psi_t(x_t^+) + I_t(x_t^+) + \mathbf{E}\,\tilde{V}_{t+1}(R_{t+1}x_t^+)$$

over x_t^+. Moreover, the minimum value of this expression is independent of x, which, together with (3.1), implies $\tilde{V}_t$ are constant functions. It follows that we can find $x_t^{+\star}$ as the minimizers of

$$(\mathbf{1} - \overline{r}_{t+1})^T x_t^+ + \psi_t(x_t^+) + I_t(x_t^+).$$

Thus, an optimal policy can be found as follows. For $t = 0, \ldots, T$, we solve the problems

$$\begin{array}{ll} \text{minimize} & (\mathbf{1} - \overline{r}_{t+1})^T x_t^+ + \psi_t(x_t^+) \\ \text{subject to} & x_t^+ \in \mathcal{C}_t, \end{array} \tag{3.6}$$

over variables x_t^+, to determine optimal post-trade portfolios $x_t^{+\star}$. An optimal policy simply rebalances to these optimal post-trade portfolios:

$$\phi_t^\star(x) = x_t^{+\star} - x$$

(which is an affine policy, of a special form). The optimal objective $J^\star$ is the sum of the optimal values of the problems (3.6) less the total wealth of the initial portfolio, $\mathbf{1}^T x_0$.

The optimization problems (3.6) are single-stage portfolio problems, which can be solved independently (in parallel). When the stage-cost function is QP-representable, they reduce to QPs; for standard deviation risk limits, they reduce to SOCPs.

In summary, when all stage costs are functions only of the post-trade portfolio, the multi-period investment problem is readily solved using standard single-period portfolio optimization. However, when cost terms that depend on u are included in the model, such as broker commission, bid-ask spread, or price-impact, the full stochastic control formulation is needed. These are significant terms in most practical multi-period investment problems, which provides the motivation for developing a method for handling them.

4

Performance Bounds

Here we describe methods for computing a lower bound on the optimal value $J^\star$, using techniques described in a recent series of papers [100, 97, 96, 75]. We describe the development of these bounds starting from the high level ideas, and then proceed to lower level details.

These methods are based on finding a set of convex quadratic functions $V_t^{\rm lb}$ that are underestimators of V_t, *i.e.*, satisfy $V_t^{\rm lb} \leq V_t$, where the inequality means pointwise. It follows that

$$J^{\rm lb} = V_0^{\rm lb}(x_0) \leq V_0(x_0) = J^\star,$$

i.e., $J^{\rm lb}$ is a lower bound on $J^\star$. We let $V_t^{\rm lb}$ take the form

$$V_t^{\rm lb}(x) = (1/2) \left[\begin{array}{c} x \\ 1 \end{array} \right]^T \left[\begin{array}{cc} P_t & p_t \\ p_t^T & q_t \end{array} \right] \left[\begin{array}{c} x \\ 1 \end{array} \right],$$

where $P_t \succeq 0$.

4.1 Bellman inequalities

The quadratic underestimators are found as follows. We construct a set of quadratic functions that satisfy the Bellman *inequalities*, [60, 42, 30],

$$V_t^{\rm lb} \leq \mathcal{T}_t V_{t+1}^{\rm lb}, \quad t = T, \ldots, 0, \tag{4.1}$$

with $V_{T+1}^{\rm lb} = 0$ (*cf.* the Bellman recursion *equalities*, given in (3.2)). By monotonicity of the Bellman operator we get

$$V_T^{\rm lb} \leq \mathcal{T}_T V_{T+1}^{\rm lb} \leq \mathcal{T}_T V_{T+1} = V_T.$$

Continuing this argument recursively we see that

$$V_t^{\rm lb} \leq V_t, \quad t = 0, \ldots, T+1,$$

i.e., $V_t^{\rm lb}$ are underestimators of V_t.

4.2 LMI conditions

Now we explain how to obtain quadratic functions that satisfy the Bellman inequalities (4.1). For simplicity, we describe the method when ψ_t is convex quadratic and $\mathcal{C}_t$ has the representation

$$\mathcal{C}_t = \{x \mid g_1(x) \leq 0, \ldots, g_s(x) \leq 0,\ g_{s+1}(x) = \cdots = g_M(x) = 0\}, \quad (4.2)$$

where $g_i : \mathbf{R}^n \to \mathbf{R}$ are convex quadratic constraint functions. (In fact, after appropriate problem transformations, this includes all of the transaction cost terms and constraints described in §2.2, except for the more sophisticated risk bounds.)

The Bellman inequality (4.1) can be written as

$$V_t^{\rm lb}(x) \leq \mathbf{1}^T u + \psi_t(x,u) + \mathbf{E}\, V_{t+1}^{\rm lb}(R_{t+1}(x+u)), \quad \forall (x+u) \in \mathcal{C}_t.$$

From appendix A we know that when $V_{t+1}^{\rm lb}$ is convex quadratic, $\mathbf{E}\, V_{t+1}^{\rm lb}(R_{t+1}(x+u))$ is also convex quadratic, so

$$\mathbf{1}^T u + \psi_t(x,u) + \mathbf{E}\, V_{t+1}^{\rm lb}(R_{t+1}(x+u)) - V_t^{\rm lb}(x) \quad (4.3)$$

is quadratic in (x,u). Thus, the Bellman inequality (4.1) requires that the quadratic function (4.3) be nonnegative on the set $\mathcal{C}_t$, which is described by the set of quadratic inequalities (4.2).

A sufficient condition for a quadratic function to be nonnegative on a set described by a set of quadratic inequalities can be obtained via the $\mathcal{S}$-procedure [18, §2.6.3], [20, §B.2], which results in a set of linear matrix inequalities (LMIs) in the coefficients of $V_t^{\rm lb}$ (see appendix C for details). If P_t, p_t, q_t, $t = 0, \ldots, T+1$ satisfy these LMIs, then $V_t^{\rm lb}$,

$t = 0, \ldots, T+1$ must satisfy the Bellman inequalities (4.1), and hence are value function underestimators. We can then maximize our lower bound on $J^\star$,

$$J^{\mathrm{lb}} = V_0^{\mathrm{lb}}(x_0) = (1/2)x_0^T P_0 x_0 + p_0^T x_0 + q_0$$

(which is a linear function of the coefficients of V_t^{lb}), subject to these LMIs. This is an SDP with variables P_t, p_t, q_t, $t = 0, \ldots, T+1$.

The details of this method, including a full derivation of the SDPs involved, can be found in appendix D and [100, 97].

4.3 Summary

The method described in this chapter produces a (numerical) lower bound on $J^\star$, the optimal value of the stochastic control problem, given the problem data, *i.e.*, the stage cost function and the first and second moments of the return distributions. We note that while the performance of any given policy can depend on higher moments of the returns, our performance bound only depends on the first and second moments. The method requires forming and solving an SDP, with on the order of T matrix variables of size $n \times n$.

5

Approximate Dynamic Programming

5.1 Basic idea

Except for the special case when the problem is quadratic, described in §3.2 above, it is usually impossible to compute (or even represent) the value functions V_t. A common alternative is to replace the value functions appearing in (3.3) with approximate (or surrogate) value functions $\hat{V}_t : \mathbf{R}^n \to \mathbf{R}$, which gives the ADP policy

$$\phi_t^{\text{adp}}(x) \in \operatorname*{argmin}_u \left(\ell_t(x,u) + \mathbf{E}\,\hat{V}_t(R_{t+1}(x+u))\right), \quad t = 0, \ldots, T. \tag{5.1}$$

When $\hat{V}_t = V_t$, the ADP policy is optimal; when $\hat{V}_t = 0$, the ADP policy is the greedy policy, in which trades are chosen to minimize the current stage cost, without regard for any impact on the future (other than those contained in the current stage cost). When $\hat{V}_t$ is convex, evaluating the ADP policy requires solving a convex optimization problem. We let J^{adp} denote the cost achieved by the ADP policy.

A variety of methods, which we do not survey here, can be used to choose approximate value functions (see, *e.g.*, [15, 78, 54, 30]). The goals in choosing approximate value functions $\hat{V}_t$ are the following:

- *Ease of evaluation.* The ADP policy should be easily evaluated, *i.e.*, the minimization over u in (5.1) should be easy to carry out. For example, when the stage cost is QP-representable, and $\hat{V}_t$ are convex quadratic, evaluation of the ADP policy reduces to solving a QP, for which there are very efficient methods.

- *Performance.* The ADP policy should attain near-optimal performance, *i.e.*, $J^{\mathrm{adp}} \approx J^{\star}$. (A more modest goal is for the ADP policy to obtain good performance, meaning J^{adp} is not too much larger than $J^{\star}$.) This would be the case if J^{adp} is not much larger than J^{lb}, the lower bound computed using the methods of §4.

One simple and general method for obtaining approximate value functions is to find the (exact) value functions for a quadratic problem that approximates in some sense or relaxes the original problem, for example, by ignoring portfolio constraints other than equality constraints, as well as any part of the transaction cost that is not quadratic (such as bid-ask spread). Then we use these value functions as approximate value functions for the original problem.

5.2 Quadratic approximate dynamic programming

When the approximate value functions are convex quadratic, $\mathbf{E}\,\hat{V}_t(R_{t+1}(x+u))$ is a convex quadratic function, whose coefficients can be explicitly computed, using $\overline{r}_{t+1}$ and Σ_{t+1}. (This is shown in appendix A.) Thus, evaluating the ADP policy in this case reduces to solving a convex optimization problem, with an explicit objective function that does not involve expectation over returns.

A further simplification occurs when the stage cost is QP-representable. In this case, evaluation of the ADP policy reduces to solving a QP, for which there are a number of effective methods, already mentioned above. It can be shown in this case that the ADP policy is a piecewise affine function of the current portfolio x [8], *i.e.*, it has the form

$$\phi_t^{\mathrm{adp}}(x) = J_t^i x + k_t^i, \quad x \in \mathcal{R}_t^i,$$

where $\{\mathcal{R}_t^1, \ldots, \mathcal{R}_t^N\}$ is a polyhedral partition of $\mathbf{R}^n$ (*cf.* the optimal

policy for the quadratic case, given in (3.5)). This observation leads to another method for implementing the ADP policy, called explicit MPC or multi-parametric QP [51, 7]: We compute J_t^i, k_t^i, and (the inequalities that define) $\mathcal{R}_t^i$ explicitly, off-line. On-line, policy evaluation requires two steps. First, the region i containing the current portfolio x_t is determined; then the corresponding affine trading policy is applied. This method works well when the number of assets is small (say, no more than around 5), but quickly becomes intractable for larger portfolios, since N grows exponentially with n.

5.3 ADP based on quadratic underestimators

The quadratic underestimators V_t^{lb} found using the performance bounding method described in chapter 4 are natural candidates for approximate value functions in ADP. Indeed, ADP policies using $\hat{V}_t = V_t^{\mathrm{lb}}$ have been observed to perform well in many practical applications, including variations on the multi-period portfolio optimization problem considered here [97, 100].

Let us summarize what is required to evaluate this ADP policy.

- *Performance bound computation.* We solve an SDP to obtain J^{lb}, a lower bound on $J^\star$, as well as the (coefficients of) quadratic approximate value functions V_t^{lb}. This SDP involves around T matrix variables of size $n \times n$. This can be done off-line, before the trading begins.

- *On-line policy evaluation.* In each period we solve a small QP, with around n variables, which includes the current portfolio x_t in its data, to determine which trades to carry out.

The first task can involve substantial computation, but is carried out off-line; the second task, which is carried out on-line in each trading period, can be done very quickly.

6

Model Predictive Control

In this chapter we describe certainty-equivalent model predictive control (MPC), another suboptimal policy for the multi-period investment problem. MPC is a widely used and extensively studied suboptimal policy; see, *e.g.*, [59, 67, 101, 39, 52, 36]. It can also be interpreted as an ADP policy, with an appropriately defined approximate value function.

6.1 Policy

MPC is based on a simple idea. To determine u_t, we replace all future (unknown) returns with their mean values, $\overline{r}_\tau$, $\tau = t+1, \ldots, T$. This turns the stochastic control problem into a standard optimization problem,

$$\begin{array}{ll} \text{minimize} & \sum_{\tau=t}^{T} l_\tau(z_\tau, v_\tau) \\ \text{subject to} & z_{\tau+1} = \mathbf{diag}(\overline{r}_{\tau+1})(z_\tau + v_\tau), \quad \tau = t, \ldots, T-1 \\ & z_t = x_t \end{array} \tag{6.1}$$

with variables z_τ, v_τ, $\tau = t, \ldots, T$. We solve this convex optimization problem to obtain an optimal sequence of trades $v_t^\star, \ldots, v_{T-1}^\star$. This sequence is a plan for future trades over the remaining trading horizon, under the (highly unrealistic) assumption that future returns will be

equal to their mean values. The MPC policy takes $u_t = \phi^{\mathrm{mpc}}(x_t) = v_t^\star$. In other words, we execute the first trade in our planned sequence of trades. At the next step, we repeat the process, starting from the new portfolio x_{t+1}.

6.2 Implementation

Evaluating $\phi^{\mathrm{mpc}}(x_t)$ requires solving the convex optimization problem (6.1), which is an optimization problem with $2(T-t)n$ variables. Using a generic optimization solver that does not exploit sparsity structure, the computational effort required is order $(T-t)^3n^3$ flops (floating-point operations). The MPC problem (6.1) can be more efficiently solved by exploiting its structure. When the variables are appropriately ordered, the linear equations that are solved in each step of an interior-point method are block tri-diagonal, and can be solved with a computational effort that grows linearly in $T-t$. In this case, the overall computational effort is order $(T-t)n^3$ flops. For details, see, *e.g.*, [98, 66, 2, 81], which describe implementations of the MPC policy for similar control problems. Note that the cost of evaluating the MPC policy decreases with increasing t, since later in the trading interval our planning is done over a shorter horizon. The total cost of the T evaluations of the MPC policy, for $t = 1, \ldots, T$, is order T^2n^3 flops. In contrast, the total cost of T evaluations of the ADP policy is order Tn^3 flops.

Comparison with quadratic ADP policy. We can compare the computational cost of the MPC and ADP policies. The main difference is that the ADP policy requires significantly more off-line computation, while MPC is more expensive to evaluate on-line.

- *Off-line pre-computation.* The ADP policy requires solving a large SDP to find the quadratic approximate value functions. This is done off-line, before the trading policy is implemented on-line. In contrast, MPC does not require any pre-computation. The only computation that is required is solving (6.1), which is solved on-line at every time step.

- *On-line policy evaluation.* In ADP, after the quadratic approximate value functions have been computed, on-line evaluation requires solving a QP with n variables. The computational effort required is order n^3 flops. In contrast, evaluating the MPC policy requires order $(T-t)n^3$ flops (assuming we exploit the sparsity structure).

Signals. There are computational advantages to using MPC in cases when (estimates of) future return statistics are updated in real-time, using signals. In this case, the expected returns $\overline{r}_t$ are simply replaced with the most recent return estimates. In contrast, for our quadratic ADP policies, when return statistics are updated, the quadratic approximate value function must be re-computed, which requires solving a large SDP. Our ADP policies can be extended to handle some real-time signals without re-solving the SDP, but the method is more complicated, and is beyond the scope of this paper.

6.3 Interpretation as an ADP policy

Model predictive control is itself an ADP policy, and can be interpreted as a special case of a method called *rollout*, a popular method in approximate dynamic programming [10, §6.4] [11, 12, 89]. In rollout, the approximate value function is taken to be the cost achieved by a heuristic policy called the *base policy*. In most cases, the cost achieved by the base policy can only be evaluated via Monte Carlo simulation, *i.e.*, by 'rolling out' possible sample paths.

In MPC, the base heuristic is an open loop policy, where we compute an optimal sequence of trades starting from an initial portfolio, assuming that the true returns are equal to their mean values $\overline{r}_t$. In this base policy, the planned sequence of trades is executed without recourse, *i.e.*, the trades depend only on the initial portfolio. Evaluating the cost achieved by this policy would require Monte Carlo simulation, which is computationally intensive (We would need to evaluate the cost achieved over a large number of return trajectories and average.)

Thus, in MPC a further simplification is to roll out only one return

trajectory—the trajectory of mean returns. We can write the MPC policy as

$$\phi_t^{\mathrm{mpc}}(x) = \operatorname*{argmin}_u \left(\ell_t(x,u) + V_{t+1}^{\mathrm{mpc}}(\overline{R}_{t+1}(x+u))\right), \quad t = 0, \ldots, T,$$

where $V_t^{\mathrm{mpc}}(z)$ is the optimal value of the problem (6.1), starting from $x_t = z$ at period t. We can see that MPC can be interpreted as an ADP policy, with a relatively complex approximate value function (given as the optimal cost of a convex optimization problem).

For more details on interpretations of MPC, and connections to approximate dynamic programming and rollout, see [11, 12, 10, 13, 8].

6.4 Truncated MPC

The MPC policy described above plans a sequence of trades for the full time interval $t, \ldots, T$. A common variation is to look ahead a limited number of steps, M, into the future. At each time t we solve

$$\begin{array}{ll} \text{minimize} & \sum_{\tau=t}^{t+M-1} l_\tau(z_\tau, v_\tau) + V_{t+M}^{\mathrm{term}}(z_{t+M}) \\ \text{subject to} & z_{\tau+1} = \mathbf{diag}(\overline{r}_{\tau+1})(z_\tau + v_\tau), \quad \tau = t, \ldots, t+M-1 \\ & z_t = x_t \end{array}$$

with variables $v_t, \ldots, v_{t+M-1}$, and $z_t, \ldots, z_{t+M}$. Here, M is the number of steps of look-ahead, and V_{t+M}^{term} is the terminal cost (to be chosen). As with full look-ahead MPC, we take $\phi_t^{\mathrm{mpc}}(x_t) = v_t^\star$ and repeat the process at the next time step, starting from the portfolio x_{t+1}. For $t > T - M$, we use the basic MPC policy (6.1).

An important parameter in this policy is the terminal cost V_{t+M}^{term}. If this cost is appropriately chosen, the truncated MPC policy is exactly the same as the full look-ahead policy [8]. Common choices for terminal costs are approximate value functions obtained via any approximate dynamic programming method.

The idea in truncated MPC is to trade off on-line and off-line computation. If the number of steps of look-ahead M is large, we need a less accurate approximate value function as our terminal cost, so less off-line computation is needed. On the other hand, if we are willing to spend a lot of time off-line computing a good approximate value

function, we can set $M \ll T$ and save significant computational effort on-line. Thus, truncated MPC gives a family of policies that sit in between (full look-ahead) MPC and ADP.

7

Numerical Examples

In this chapter, we present several numerical examples. We first describe the five problems we consider.

7.1 Problem data

All five examples use zero initial portfolio $x_0 = 0$, and impose a zero terminal portfolio constraint: $x_T^+ = 0$. The examples use the same return distribution; they differ only in the choice of stage cost function. Our first example is a quadratic problem for which we can compute the optimal policy exactly. The others use the same transaction cost terms and differ only in the choice of constraints. All examples involve $n = 30$ assets over a time horizon of 100 time steps, *i.e.*, $T = 99$.

Return distribution. The returns r_t are identically distributed with log-normal distribution,

$$\log r_t \sim \mathcal{N}(\mu, \tilde{\Sigma}),$$

where μ and $\tilde{\Sigma}$ are the mean and covariance of the log return. The return mean and covariance are then given by

$$\overline{r} = \exp(\mu + (1/2)\,\mathbf{diag}(\tilde{\Sigma})), \qquad \Sigma_{ij} = \overline{r}_i \overline{r}_j (\exp \tilde{\Sigma}_{ij} - 1), \quad i, j = 1, \ldots, n.$$

The log return mean and covariance were chosen as follows. First we chose the log return standard deviations $(\tilde{\Sigma}_{ii})^{1/2}$ from a uniform distribution on $[0, 0.01]$. We then formed $\tilde{\Sigma}$ using

$$\tilde{\Sigma}_{ij} = C_{ij}(\tilde{\Sigma}_{ii}\tilde{\Sigma}_{jj})^{1/2}, \quad i, j = 1, \ldots, n,$$

where C is a matrix of correlation coefficients chosen at random, with entries varying from about -0.3 to $+0.9$. To form C we generate a matrix $Z \in \mathbf{R}^{n\times n}$ with all entries drawn from a standard Gaussian distribution, then form $Y = ZZ^T + \zeta \mathbf{1}\mathbf{1}^T$, where $\zeta > 0$, and finally set

$$C = \mathbf{diag}(Y_{11}^{-1/2}, \ldots, Y_{nn}^{-1/2}) Y \, \mathbf{diag}(Y_{11}^{-1/2}, \ldots, Y_{nn}^{-1/2}).$$

We chose ζ so that the entries of C are in the range we desire. The log return means μ_i were chosen from a $\mathcal{N}(0, 0.03^2)$ distribution. The resulting mean returns $\overline{r}_i$ ranged from 0.95 to 1.08; the asset standard deviations $(\Sigma_{ii})^{1/2}$ ranged from 0.03 to 0.10.

Transaction cost. For simplicity we consider a transaction cost that does not vary over time, *i.e.*, $\psi_t = \psi$ for $t = 0, \ldots, T$. For the quadratic example, the transaction cost is

$$\psi(x_t, u_t) = s^T u_t^2 + \lambda (x_t^+)^T \Sigma x_t^+,$$

which includes a quadratic transaction cost and a quadratic risk penalty. For the other examples, the transaction cost is

$$\psi(x_t, u_t) = c^T (x_t^+)_- + \kappa^T |u_t| + s^T u_t^2 + \lambda (x_t^+)^T \Sigma x_t^+,$$

which includes a quadratic risk penalty, a quadratic transaction cost, a shorting fee and an absolute term to model bid-ask spread and broker commissions. The stage cost for the quadratic example is smaller than the stage cost for the other examples, since the additional terms are nonnegative. It follows that the optimal cost for the quadratic example is a lower bound on the optimal cost for the other examples.

The parameter λ was set to 0.5; s_i, κ_i, and c_i were sampled from uniform distributions on $[0, 1]$, $[0, 0.1]$ and $[0, 0.05]$, respectively. These choices resulted in each term giving a significant contribution to the over all objective.

Constraint set. For all five examples we have a zero terminal portfolio constraint, *i.e.*, $\mathcal{C}_T = \{0\}$. The quadratic case has no other constraints, *i.e.*, $\mathcal{C}_t = \mathbf{R}^n$ for $t = 0, \ldots, T-1$. For the other four examples we take one of the following constraints for $t = 0, \ldots, T-1$:

- Unconstrained: $\mathcal{C}_t = \mathbf{R}^n$.
- Long-only: $\mathcal{C}_t = \mathbf{R}^n_+$.
- Leverage limit: $\mathcal{C}_t = \{x \mid \mathbf{1}^T(x)_- \leq \eta \mathbf{1}^T x\}$, with $\eta = 0.3$.
- Sector neutral: $\mathcal{C}_t = \{x \mid Fx = 0\}$, with $F \in \mathbf{R}^{2\times n}$; the rows of F are the eigenvectors associated with the two largest eigenvalues of Σ.

7.2 Computation

7.2.1 Cost evaluation and simulation

For the quadratic example, we can evaluate the optimal cost exactly, as $V_0(0)$, as well as the (affine) optimal policy. For the other four problems we use Monte Carlo simulation to compute (approximately) the objective value obtained by the ADP and MPC policies. For the quadratic example, the estimate of objective obtained from Monte Carlo serves as a consistency check. To evaluate the performance of the ADP policy we ran 50000 Monte Carlo simulations for each example. Thus, Monte Carlo simulation of each example required solving around 5 million small QPs. To evaluate the performance of the MPC policy we ran 5000 Monte Carlo simulations for each example, which required solving 500000 larger MPC QPs. (Of course we could have reduced the number of Monte Carlo samples we needed to take by using one of the many variance reduction techniques, such as control variates or importance sampling.)

Our sampling was sufficient to obtain around three significant figures of accuracy in our performance evaluations, as estimated by the empirical variance in our estimates (reported below), and also verified experimentally by re-running the simulations with different initial

random number generator seed, which yielded numerical results that agreed to three significant figures.

7.2.2 Convex optimization solvers

All computation was carried out on an Intel Xeon processor, with clock speed 3.4GHz, running Linux.

We used CVX [40] to formulate the SDPs required to compute the performance bounds, which in turn uses SeDuMi [88] to solve the transformed SDPs. For reference, the SDP required to compute the performance bound for the long-only example had 474100 variables and 59096 constraints after transformation to a standard form SDP and took about 24 minutes to solve. (An optimized solver for this type of problem could have solved it much more quickly.)

To solve the QPs needed to evaluate ADP policies, we used CVXGEN [64] to generate fast custom solvers. The QP solved in each time step to evaluate the ADP trading policy has 30 variables and 30 constraints; transformed into a standard form QP it has 90 variables and 122 constraints. The CVXGEN generated solver (which is single thread) solves such a problem in around 300μs. (This computation time could easily have been reduced by a factor of 3 or more using various tricks for fast embedded solvers [64, 63, 62].) Thus, a simulation run of 100 time periods requires about 30ms; running 50000 simulations in series can be carried out in around 25 minutes. On the 8-core Xeon, the 5 million QPs are solved in slightly more than 3 minutes. (Using CVX, this would have taken days.) Of course, the Monte Carlo simulations are trivially parallelizable, and could easily have been carried out on k processors with a speed-up of around k.

Evaluating the MPC policy required solving much larger QPs, beyond the size limits of CVXGEN. To solve these QPs quickly we used an operator splitting technique known as the alternating direction method of multipliers (ADMM) [19], specialized to the case of convex optimal control, and described in full in [74]. The QPs to be solved to evaluate the MPC policy are larger at earlier time periods and get progressively smaller as we approach the final period. For example, the QP solved at the first period for the long-only example has 18243 variables and

Example	Lower bound	ADP	MPC
quadratic	-450.1	-450.0	-444.3
unconstrained	-132.6	-131.9	-130.6
long-only	-41.3	-41.0	-40.6
leverage limit	-87.5	-85.6	-84.7
sector neutral	-121.3	-118.9	-117.5

Table 7.1: Lower bound on performance, and ADP and MPC policy performance.

9052 constraints (when converted to standard form). It requires about 88ms to solve this QP using ADMM; solving the same problem using CVX takes more than 3 minutes.

7.3 Performance bounds and policy performance

The results are summarized in Table 7.1. The first column gives the lower bound on performance computed using our method. The second and third columns give the performance attained by the ADP and MPC policies, respectively, estimated via Monte Carlo simulation. For the quadratic example, the lower bound is the optimal performance; the ADP policy is in fact the optimal policy, so the ADP performance entry for the quadratic example serves as a check on our Monte Carlo simulation (which is consistent with our estimate of Monte Carlo error given below). In all cases we see that the policies are very nearly optimal, since the lower bound and performance obtained are very close. In any practical sense, our ADP and MPC trading policies are optimal.

7.3.1 Monte Carlo error estimation

The empirical standard deviations for the ADP and MPC Monte Carlo simulations are in Table 7.2. For the quadratic problem the positions typically involve higher leverage than for the other problems, resulting in total cost with higher variance. For the quadratic problem, the Monte Carlo error standard deviation is estimated to be around $30/\sqrt{50000} \approx 0.13$ for the ADP simulations, and around $40/\sqrt{5000} \approx 0.6$ for the

Example	ADP	MPC
quadratic	30.9	40.3
unconstrained	9.6	9.9
long-only	7.1	7.2
leverage limit	9.8	9.9
sector neutral	8.9	9.9

Table 7.2: Empirical standard deviations of total cost for ADP and MPC Monte Carlo simulations.

MPC simulations. For the others, it is estimated to be around 0.04 for the ADP simulations and 0.14 for the MPC simulations. Re-running the simulations with different random number generator seed yields numerical results consistent with these error estimates.

7.3.2 Simpler methods

We also evaluated simpler policies for comparison. Our first simple method was to ignore the transaction cost terms, which results in problems that can be solved exactly, as described in §3.3. Since the ignored transaction cost terms are nonnegative, this gives a lower bound on the performance, as well as a performance value (evaluated including the transaction costs). The other simple method is to use the (exact) value function for the quadratic example as an approximate value function. Here too we get a lower bound, which is simply the optimal performance for the quadratic example. Table 7.3 lists the lower bound obtained by ignoring the transaction cost, the performance obtained when the transaction cost is ignored, the lower bound attained by ignoring nonquadratic terms, and the performance obtained when the quadratic value function is used in ADP. We can see that the bounds obtained are quite poor, with the exception of the long-only example, which is merely bad. Note in particular that these simpler policies result in positive objective value for all examples, meaning that they lose money money on average; they are worse than not trading at all (*i.e.*, taking $u_t = 0$ for all t).

	No transaction costs		No nonquadratic terms	
Example	**bound**	**performance**	**bound**	**performance**
unconstrained	-4405	2.1×10^6	-450.1	59.2
long-only	-70.6	864.5	-450.1	162.3
leverage limit	-241.4	4525	-450.1	219.5
sector neutral	-4391	2.1×10^6	-450.1	75.6

Table 7.3: Lower bounds and performance obtained by simpler policies, obtained by ignoring transaction costs (first two columns), and by ignoring nonquadratic terms (last two columns).

7.4 Simulation results and trajectories

In this section we give some more detail of the simulations, showing typical and average trajectories of various quantities over the investment period. These simulations show that the problems are not simple, that none of the stage cost terms is negligible, that the constraints are indeed active, and that our ADP and MPC policies are not obvious.

Table 7.4 summarizes average values for various quantities over the 100 time steps and 50000 simulations under the ADP policy, for each example. The quantities are: gross cash in $\mathbf{1}^T u_t$, risk cost $\lambda(x_t^+)^T\Sigma(x_t^+)$, quadratic transaction cost $s^T u_t^2$, linear transaction cost $\kappa^T|u_t|$, and total long, short, and long-short positions. These numbers show that each of the cost terms contributes to the overall objective (except in the quadratic example). For the quadratic example, the stage cost does not include a shorting fee or an absolute value transaction cost, so those numbers are in parentheses; they are what we would have been charged, had these stage cost terms been present.

To get an idea of the trajectories over the time horizon, Figure 7.3 shows several time trajectories for the leverage limit example under the ADP policy. The lighter lines show 5 sample traces from the 50000 Monte Carlo simulations. The darker lines are the average (or in the case of the leverage, the median) over all the runs. The upper left plot shows the total value of the long and short positions (in blue and red, respectively), and the upper right plot shows the leverage ratio. We see that the leverage ratio saturates at the leverage limit, η, which

Example	$\mathbf{1}^T u_t$	$\lambda(x_t^+)^T \Sigma x_t^+$	$s^T u_t^2$	$c^T (x_t^+)_-$
quadratic	-9.00	0.77	3.73	(4.93)
unconstrained	-3.82	0.29	1.03	0.96
long-only	-0.86	0.32	0.08	0
leverage limit	-2.03	0.54	0.35	0.18
sector neutral	-3.71	0.19	1.02	1.07

Example	$\kappa^T \lvert u_t \rvert$	$\mathbf{1}^T (x_t^+)_+$	$\mathbf{1}^T (x_t^+)_-$	$\lVert x_t^+ \rVert_1$
quadratic	(0.70)	164.16	168.76	332.92
unconstrained	0.22	80.93	47.76	128 .7
long-only	0.04	25.7	0	25.7
leverage limit	0.11	52.66	12.03	64.7
sector neutral	0.23	77.2	53.5	130.7

Table 7.4: Average of various quantities over all simulations under the ADP policy.

is 0.3 in this case. The middle left plot shows the gross cash in, and the middle right plot shows cumulative net cash in. We can see that (on average) we put money into the portfolio over the first 12 or so steps; after that we derive income from the portfolio, and (of course) cash out on the last step. The cumulative net cash at time T is our performance, for which we have the lower bound of -87.5; this bound is shown with a red dashed line. The bottom left plot shows the total of the quadratic, absolute value, and shorting costs, which is relatively large in the beginning (while we set up our portfolio) and the end (when we cash out). The risk cost is shown in the bottom right plot. This naturally grows as we invest in the portfolio, and then shrinks as we cash out.

Figure 7.1 shows the distribution of total cost for the leverage limit example under the ADP policy over the Monte Carlo simulations. The mean of this distribution is the performance of the policy, shown as a solid blue line; the lower bound is shown in red.

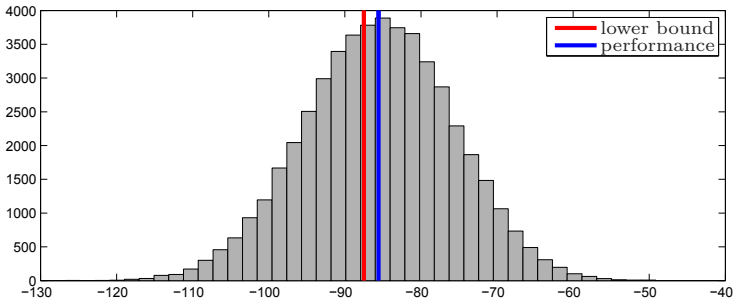

Figure 7.1: Histogram of total cost for the leverage limit example, with ADP policy.

7.5 Robustness to model parameters

Here we show that the performance of our policies is not very sensitive to the return distribution parameters, even though our formulation does not include explicit robustness requirements. We carry out 10 additional sets of 5000 Monte Carlo simulations for each example, drawing the returns in each simulation from a log-normal distribution with perturbed mean return $\overline{r}_t^{\text{pert}}$. (We can think of $\overline{r}_t$ as our estimate of the returns, and $\overline{r}_t^{\text{pert}}$ as the true mean return.)

The 10 perturbations were found as follows. We used log return mean $\mu^{\text{pert}} = \mu + \delta$, where μ is the log return mean used in development of the ADP policy, and δ_i were sampled from a uniform distribution on $[-0.003, +0.003]$. Since $\mu_i \sim \mathcal{N}(0, 0.03^2)$, this perturbation amounts to a change of log return means that ranges from around 1% to a factor of 5, including cases in which the signs of μ_i and μ_i^{pert} are different. (A few assets that are 'winners' under the assumed distribution become 'losers' under the perturbed return statistics, and vice versa.) Thus, the perturbation of the return distribution is not small.

Table 7.5 gives the *worst* performance values over the 10 perturbed cases. These values are a bit worse than the nominal values, as expected, but quite respectable given the fairly large perturbation of the return distribution. (The average performance values over the 10 perturbations are very close to the nominal performance values.)

Example	Nominal performance	Worst-case performance
quadratic	-450.0	-425.1
unconstrained	-131.9	-123.4
long-only	-41.0	-38.2
leverage limit	-85.6	-80.3
sector neutral	-118.9	-110.8

Table 7.5: ADP policy performance with perturbed return distribution.

Example	Original	Heavy-tailed
quadratic	-450.0	-450.6
unconstrained	-131.9	-132.0
long-only	-41.0	-41.0
leverage limit	-85.6	-85.5
sector neutral	-118.9	-118.8

Table 7.6: ADP policy performance with heavy-tailed return distribution.

7.6 Robustness to return distribution

Our performance bounds and policies depend only on the first and second moments of the return distribution, whereas the performance obtained by our policies can depend on higher order moments of the return distribution. In this section we demonstrate that the performance obtained by our policies is not particularly sensitive to these higher order moments, in particular, when the returns come from a distribution with substantially larger tails than a log-normal.

We evaluated the performance of the ADP policy, with the returns sampled from a heavy-tailed distribution with identical first and second moments to the original log-normal distribution (so our numerical bounds are unchanged). We generated the new returns as

$$\log \hat{r}_t \sim \begin{cases} \mathcal{N}(\mu_1, \hat{\Sigma}) & \text{w.p. } 0.8 \\ \mathcal{N}(\mu_2, 10\hat{\Sigma}) & \text{w.p. } 0.2 \end{cases}$$

where μ_1, μ_2 and $\hat{\Sigma}$ were chosen so that the first and second moments of $\hat{r}_t$ matched those of the original distribution. Intuitively, with prob-

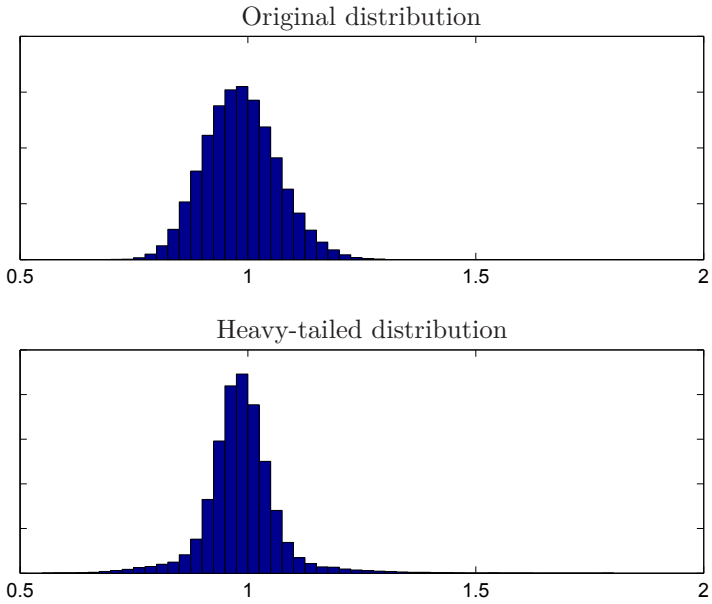

Figure 7.2: A comparison of return distributions for the first asset.

ability 0.2 the system experiences a significant 'event' that typically moves the asset prices much more drastically than usual.

Figure 7.2 compares the original and the heavy-tailed distribution of the returns of the first asset. The average kurtosis, or fourth standardized moment, of the returns increased from 3.1 for the original distribution to 8.8 under our new distribution, indicating a distribution with heavier tails. Table 7.6 summarizes the performance of the ADP policy under the original and new heavy-tailed distribution for our five examples. The performance of the policy under the new distribution was evaluated via Monte Carlo with 5000 samples. The performance under the heavy-tailed distribution was almost identical to, and in some cases better than, the original performance.

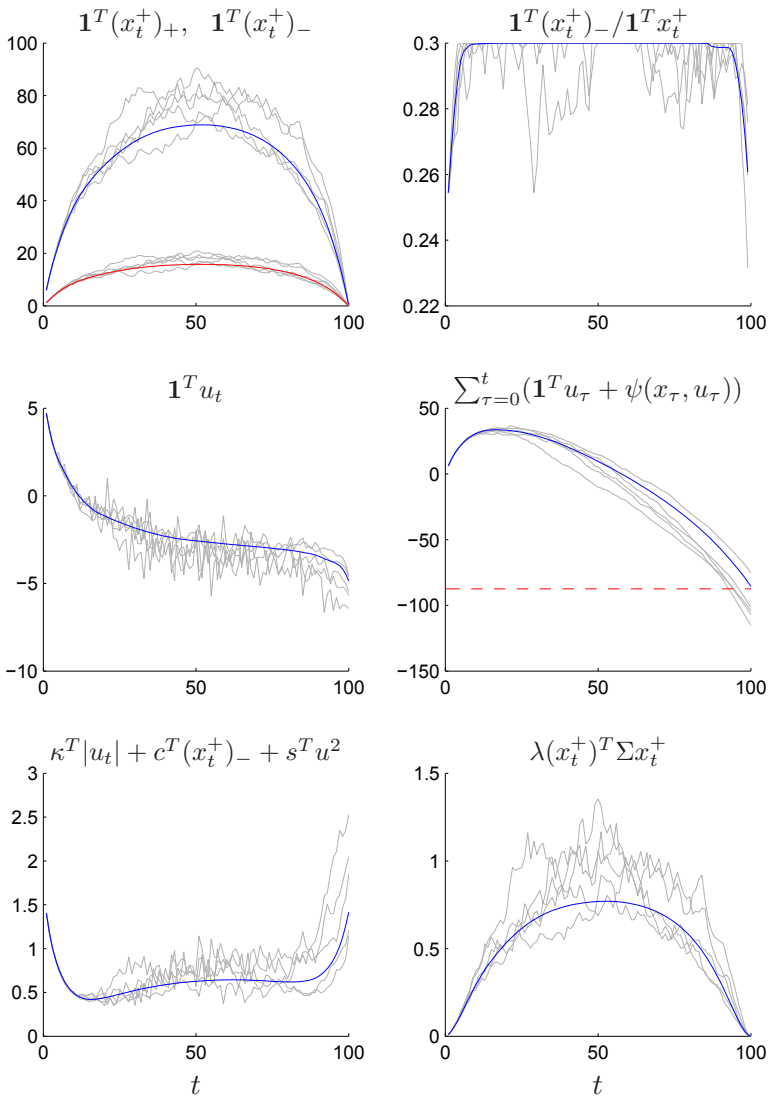

Figure 7.3: Trajectories of various quantities for the leverage limit example under the ADP policy.

8

Conclusions

In this paper we formulated the multi-period portfolio optimization problem as a convex stochastic control problem with linear dynamics. Our model captures many features of real multi-period portfolio optimization problems, including limits on leverage, complex transaction costs, and time-varying return statistics. (Features that are not captured by our model include minimum allowed nonzero positions, nonconvex transaction costs, price momentum, and multi-period price impact.) Our formal model does not include the use of signals, *i.e.*, on-line updating of the problem parameters during the trading period; we discuss this below.

We used recently developed methods based on LMIs to compute a performance bound—a specific number—for any particular multi-period portfolio problem. This performance bound can be used as a yardstick against which the performance of any policy can be judged. As a by-product of the performance bound computation, we obtain a quadratic approximation of the value functions, which can be used as the approximate value functions in an ADP policy, or as a terminal cost in an MPC policy. Both the ADP and MPC policies require the solution of a convex optimization problem, typically a QP, to compute

the trades to be executed in each step.

We do not currently have an upper bound on how far our performance bound will be from the performance of the ADP or MPC policies. At the moment, all we can do is evaluate the bound, and the performance of the ADP or MPC policies, for any particular multi-period portfolio optimization problem, and subtract them to obtain a gap. We would welcome a result that upper bounds the gap, based on general features of the problem (such as dimensions, or type of stage cost). Such a result would likely require further assumptions about the stage costs and constraints; moreover, it would be unlikely to be sharp enough to be of practical use.

In numerical examples, however, we have seen that the gap is very small, which means that our ADP and MPC policies (and bounds) are nearly optimal for the particular problem instances we consider. We certainly do not claim that the gap will always be small; we merely observe that it often is. Even for problem instances with a larger gap (such as, say, a factor of two), the method seems to us to be very useful: It provides a good (if not known to be nearly optimal) trading policy, along with an upper bound on how suboptimal it can be (in this case, a factor of two).

Our methods are easily implementable, thanks to recent advances in convex optimization parser-solvers for solving complex SDPs, and code generation tools for generating extremely fast solvers for policy evaluation. In particular we rely on CVX, which calls SDP solvers SeDuMi or SDPT3, and CVXGEN, which is a code generation system for QPs. Without these tools, a user would have to manually transform the SDPs into standard form, and manually write custom solvers for policy implementation (which would take days, if not months). Instead, CVX and CVXGEN allow us to formulate, solve, and implement the policies with relatively little coding, and then carry out extensive Monte Carlo simulation in no more than a few hours, using a standard desktop computer.

We have focused specifically on QP representable problems in this paper, but the same methods extend easily to general convex stage costs and constraints. In fact, even when the problem is nonconvex,

we can still obtain suboptimal policies and performance bounds, via simple relaxations and approximations. For the single period problem, [56] outlines several approaches for handling nonconvex costs.

In our numerical examples, we have seen that our method is quite robust to uncertainties; for instance, when we do not have an accurate model for the return distribution. Robustness can also be directly incorporated into the stochastic control problem, in the computation of the bounds and approximate value functions.

There are more sophisticated methods for both finding performance bounds, as well as approximate policies, which we have not mentioned in this paper [97, 75]. But these more advanced methods are not needed for the examples considered, since the gap between policy performance and lower bound is small.

We close by addressing again the issue of signals, which are not part of our formal model. Incorporating signals in a formal model is complicated, since it involves the joint distribution of returns and signals. Even if we did incorporate signals into our formal model, it is not clear how much practical significance the optimal performance $J^\star$ would have, since in practice we would certainly not know very much about the joint distribution of signals and returns. Incorporating signals into our formal model would break much of our analysis, including our lower bound calculations based on Bellman inequalities.

On the other hand, it is easy to think of ways in which updated estimates of future problem data (*i.e.*, stage costs and return statistics) can be incorporated. For an MPC policy it is trivial to incorporate changing estimates of future return statistics: We simply use the most recently future return statistics estimates in the policy, as it is evaluated. This approach has no additional computational cost. For an ADP policy, we would need to re-do the policy synthesis whenever future estimates change, solving a new SDP to find new quadratic approximate value functions.

Acknowledgments

We are very grateful to David Brown, Matt Kraning, and Neal Parikh for helpful comments and suggestions, and to Joëlle Skaf, whose unpublished manuscript [87] inspired some of the ideas in this paper. We also thank an anonymous reviewer for suggesting that we address the special case of no transaction costs.

Appendices

A

Expectation of Quadratic Function

In this appendix we show that when $h : \mathbf{R}^n \to \mathbf{R}$ is a convex quadratic function, say,

$$h(x) = (1/2) \begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} P & p \\ p^T & q \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix},$$

with $P \succeq 0$, so is $g(x,u) = \mathbf{E}\, h(R_{t+1}(x+u))$. In fact, we will derive explicit formulas for the coefficients of g,

$$g(x,u) = (1/2) \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} A & B & a \\ B^T & C & c \\ a^T & c^T & d \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}, \tag{A.1}$$

that only require knowledge of the mean and covariance of r_{t+1}. To show that g has this form, we first observe that $h(R_{t+1}(x+u))$ is

$$\begin{aligned} & (1/2) \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} R_{t+1} & 0 \\ R_{t+1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P & p \\ p^T & q \end{bmatrix} \begin{bmatrix} R_{t+1} & R_{t+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \\ = \; & (1/2) \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} R_{t+1}PR_{t+1} & R_{t+1}PR_{t+1} & R_{t+1}p \\ R_{t+1}PR_{t+1} & R_{t+1}PR_{t+1} & R_{t+1}p \\ p^T R_{t+1} & p^T R_{t+1} & q \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}. \end{aligned}$$

Taking expectation over R_{t+1} we see that g has the form (A.1), with

$$A = B = C = \mathbf{E}\, R_{t+1} P R_{t+1} = P \circ (\Sigma_{t+1} + \overline{r}_{t+1}\overline{r}_{t+1}^T),$$

$$a = c = \mathbf{E}\, R_{t+1} p = p \circ \overline{r}_{t+1}, \quad d = q,$$

where $\circ$ denotes the Hadamard (elementwise) product. This quadratic function is convex, since $P \circ (\Sigma_{t+1} + \overline{r}_{t+1}\overline{r}_{t+1}^T) \succeq 0$ (which follows from the fact that the Hadamard product of symmetric positive semidefinite matrices is positive semidefinite).

B

Partial Minimization of Quadratic Function

In this appendix we show that partial minimization of a quadratic function over some of its variables, subject to linear equality constraints, results in a quadratic function in the remaining variables, whose coefficients are readily computed. In addition, we will show that the minimizer can be expressed as an affine function of the remaining variables, whose coefficients are readily computed.

Suppose $g : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R}$ is a convex quadratic function,

$$g(x,u) = (1/2)\begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} A & B & a \\ B^T & C & c \\ a^T & c^T & d \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix}.$$

Let us consider minimization of g over u, subject to $Fx+Gu = h$, with $h \in \mathbf{R}^k$. (We assume that such a minimizer exists.) Let $h(x)$ denote the minimum value, as a function of x.

Necessary and sufficient optimality (KKT) conditions are

$$\begin{bmatrix} C & G^T \\ G & 0 \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = -\left(\begin{bmatrix} c \\ h \end{bmatrix} + \begin{bmatrix} B^T \\ F \end{bmatrix} x \right),$$

where $y \in \mathbf{R}^k$ is a dual variable associated with the equality constraint.

It follows that

$$\left[\begin{array}{c} u \\ y \end{array}\right] = -\left[\begin{array}{cc} C & G^T \\ G & 0 \end{array}\right]^\dagger \left(\left[\begin{array}{c} c \\ h \end{array}\right] + \left[\begin{array}{c} B^T \\ F \end{array}\right] x\right)$$

is a minimizer, where $(\cdot)^\dagger$ is the Moore-Penrose pseudo-inverse. (When the KKT matrix is nonsingular, the pseudo-inverse becomes the inverse, and the minimizer is unique.) This shows that u (a minimizer, or the minimizer when it is unique) is an affine function of x, which we can write as

$$u = Jx + k.$$

Substituting this expression into g we find that h is (convex) quadratic,

$$h(x) = (1/2)\left[\begin{array}{c} x \\ 1 \end{array}\right]^T \left[\begin{array}{cc} I & 0 \\ J & k \\ 0 & 1 \end{array}\right]^T \left[\begin{array}{ccc} A & B & a \\ B^T & C & c \\ a^T & c^T & d \end{array}\right] \left[\begin{array}{cc} I & 0 \\ J & k \\ 0 & 1 \end{array}\right] \left[\begin{array}{c} x \\ 1 \end{array}\right].$$

C

$\mathcal{S}$-Procedure

Let f_i, $i = 0, \ldots, M$ be (not necessarily convex) quadratic functions in the variable $x \in \mathbf{R}^n$,

$$f_i(x) = (1/2) \begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} P_i & p_i \\ p_i^T & q_i \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}.$$

We seek a condition on the coefficients P_0, p_0, q_0 under which f_0 is nonnegative on the set defined by the quadratic inequalities and equalities

$$\mathcal{C} = \{x \mid f_1(x) \geq 0, \ldots, f_r(x) \geq 0,\ f_{r+1}(x) = \cdots = f_M(x) = 0\},$$

that is,

$$f_0(x) \geq 0, \quad \forall x \in \mathcal{C}. \tag{C.1}$$

Condition (C.1) can be thought of as an infinite number of inequalities in the coefficients P, p and q (one for each $x \in \mathcal{C}$).

An obvious sufficient condition for (C.1) is the existence of $\lambda_1, \ldots, \lambda_r \in \mathbf{R}_+$, $\lambda_{r+1}, \ldots, \lambda_M \in \mathbf{R}$, for which

$$f_0(x) \geq \sum_{i=1}^{M} \lambda_i f_i(x), \quad \forall x \in \mathbf{R}^n.$$

(This follows immediately, since the righthand side is nonnegative for $x \in \mathcal{C}$.) This condition can be written as a *linear matrix inequality*

$$\left[\begin{array}{cc} P_0 & p_0 \\ p_0^T & q_0 \end{array} \right] - \sum_{i=1}^{M} \lambda_i \left[\begin{array}{cc} P_i & p_i \\ p_i^T & q_i \end{array} \right] \succeq 0,$$

in the variables P_0, p_0, q_0, and $\lambda_1, \ldots, \lambda_M$. (We also have nonnegativity constraints on $\lambda_1, \ldots, \lambda_r$.) Thus, the $\mathcal{S}$-procedure gives a sufficient condition for (C.1) that involves a single LMI in the coefficients P_0, p_0, and q_0, as well as the (multiplier) variables $\lambda_1, \ldots, \lambda_M$.

D

LMI Sufficient Condition for Bellman Inequality

We can write a single Bellman inequality in the form

$$V \leq \mathcal{T}V^+,$$

which equivalent to

$$V(x) \leq \mathbf{1}^T u + \psi(x,u) + \mathbf{E}\, V^+(R(x+u)), \quad (x+u) \in \mathcal{C}.$$

(Here we drop all time indices for notational simplicity.) We assume that ψ is convex quadratic,

$$\psi(x,u) = (1/2) \left[\begin{array}{c} x \\ u \\ 1 \end{array} \right]^T \left[\begin{array}{ccc} A & B & a \\ B^T & C & c \\ a^T & c^T & d \end{array} \right] \left[\begin{array}{c} x \\ u \\ 1 \end{array} \right],$$

with

$$\left[\begin{array}{cc} A & B \\ B^T & C \end{array} \right] \succeq 0.$$

The constraint set $\mathcal{C}$ defined by quadratic equalities and inequalities,

$$\mathcal{C} = \{x \mid g_1(x) \leq 0, \ldots, g_r(x) \leq 0,\ g_{r+1}(x) = \cdots = g_M(x) = 0\},$$

where

$$g_i(x) = (1/2) \left[\begin{array}{c} x \\ 1 \end{array} \right]^T \left[\begin{array}{cc} G_i & f_i \\ f_i^T & h_i \end{array} \right] \left[\begin{array}{c} x \\ 1 \end{array} \right], \quad i = 1, \ldots, M.$$

We assume V and V^+ are convex quadratic,

$$\begin{aligned} V(x) &= (1/2)\begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} P & p \\ p^T & q \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} \\ V^+(x) &= (1/2)\begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} P^+ & p^+ \\ p^{+T} & q^+ \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}, \end{aligned}$$

with

$$P \succeq 0, \qquad P^+ \succeq 0.$$

Using the result in appendix A, we can write the Bellman inequality as

$$(1/2)\begin{bmatrix} x \\ u \\ 1 \end{bmatrix}^T \begin{bmatrix} \tilde{A} & \tilde{B} & \tilde{a} \\ \tilde{B}^T & \tilde{C} & \tilde{c} \\ \tilde{a}^T & \tilde{c}^T & \tilde{d} \end{bmatrix} \begin{bmatrix} x \\ u \\ 1 \end{bmatrix} \geq 0, \quad (x+u) \in \mathcal{C},$$

where

$$\begin{aligned} \tilde{A} &= A + P^+ \circ (\Sigma + \overline{r}\overline{r}^T) - P \\ \tilde{B} &= B + P^+ \circ (\Sigma + \overline{r}\overline{r}^T) \\ \tilde{C} &= C + P^+ \circ (\Sigma + \overline{r}\overline{r}^T) \\ \tilde{a} &= a + p^+ \circ \overline{r} - p \\ \tilde{c} &= c + \mathbf{1} + p^+ \circ \overline{r} \\ \tilde{d} &= q^+ + d - q. \end{aligned}$$

We note that (the coefficients of) $\tilde{A}$, $\tilde{B}$, $\tilde{C}$, $\tilde{a}$, $\tilde{c}$ and $\tilde{d}$ are affine functions of (the coefficients of) V and V^+.

Applying the $\mathcal{S}$-procedure (appendix C), a sufficient condition for the Bellman inequality is the existence of $\lambda_1, \ldots, \lambda_r \in \mathbf{R}_+$, and arbitrary $\lambda_{r+1}, \ldots, \lambda_M \in \mathbf{R}$ for which

$$\begin{bmatrix} \tilde{A} & \tilde{B} & \tilde{a} \\ \tilde{B}^T & \tilde{C} & \tilde{c} \\ \tilde{a}^T & \tilde{c}^T & \tilde{d} \end{bmatrix} \succeq \sum_{i=1}^{M} \lambda_i \begin{bmatrix} G_i & G_i & f_i \\ G_i & G_i & f_i \\ f_i^T & f_i^T & h_i \end{bmatrix}. \tag{D.1}$$

This is an LMI in the coefficients P, p, q, P^+, p^+, q^+, and $\lambda_1, \ldots, \lambda_M$.

E

No-Trade Region

We define the *no-trade region* for a policy ϕ_t in period t as the set of portfolio values in which the policy does not trade:

$$\mathcal{N}_t = \{x \mid \phi_t(x) = 0\}.$$

In this appendix we observe that for the quadratic ADP policy, the presence of linear transaction cost terms in the stage cost lead to $\mathcal{N}_t$ typically having nonempty interior.

For the ADP policy (5.1), $\mathcal{N}_t$ is the set of portfolios x for which $u = 0$ minimizes

$$\ell_t(x, u) + \mathbf{E}\, \hat{V}_t(R_{t+1}(x + u)).$$

When $\hat{V}_t$ is quadratic, the second term is convex quadratic, so we can write it as

$$(1/2)(x + u)^T P_t(x + u) + p_t^T(x + u)$$

where $P_t \succeq 0$. The necessary and sufficient condition for $u = 0$ to minimize the function above is then

$$0 \in \partial_u \ell_t(x, 0) + P_t x + p_t,$$

where ∂_u denotes the subdifferential with respect to u.

Now suppose that

$$\ell_t(x,u) = \tilde{\ell}_t(x,u) + \kappa_+^T(u)_+ + \kappa_-^T(u)_-,$$

where $\tilde{\ell}_t$ is differentiable in u for fixed x, $\kappa_+ > 0$, $\kappa_- > 0$, and (for simplicity) $\mathcal{C}_t = \mathbf{R}^n$. In other words, the stage cost contains a linear transaction cost term (with nonzero cost rates). The subdifferential of the linear transaction cost term at $u = 0$ is the rectangle in $\mathbf{R}^n$ given by $[-\kappa_-, \kappa_+]$, so the no-trade region is characterized by

$$-\left(\nabla_u \tilde{\ell}_t(x,0) + P_t x + p_t\right) \in [-\kappa_-, \kappa_+]. \tag{E.1}$$

In this case, there is a simple interpretation for the no-trade region. We interpret the lefthand side as the vector of marginal values of trading the assets in period t; we do not trade when the marginal values are smaller than the linear transaction cost rates.

When the lefthand side above (*i.e.*, the marginal utility) is a continuous function of x, and its range intersects the open rectangle $(-\kappa_-, \kappa_+)$, we conclude that $\mathcal{N}_t$ has non-empty interior.

Short-circuiting. We can use the characterization (E.1) of the no-trade region to (sometimes) speed up the evaluation of $\phi_t(x_t)$. We first check if (E.1) holds; if so, we return $u_t = 0$ as the optimal trade vector. If not, we solve the convex optimization problem required to determine the (nonzero) value of u_t. This trick does not speed up the worst-case time required to evaluate the policy, but it can reduce the average time to evaluate it, if in many periods the policy does not trade. This can be useful in simulations, for example.

References

[1] J. Adda and R. Cooper. *Dynamic economics: Quantitative methods and applications*. MIT Press, 2003.

[2] M. Åkerblad and A. Hansson. Efficient solution of second order cone program for model predictive control. *International Journal of Control*, 77(1):55–77, January 2004.

[3] P. Algoet and T. Cover. Asymptotic optimality and asymptotic equipartition properties of log-optimum investment. *The Annals of Probability*, 16(2):876–898, April 1988.

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

[5] S. Basak and A. Shapiro. Value-at-risk-based risk management: Optimal policies and asset prices. *Review of Financial Studies*, 14(2):371–405, February 2001.

[6] R. Bellman. *Dynamic Programming*. Dover Publications, 1957.

[7] A. Bemporad and C. Filippi. Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming. *Journal of Optimization Theory and Applications*, 117(1):9–38, November 2004.

[8] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[9] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[10] D. Bertsekas. *Dynamic Programming and Optimal Control: Volume 1*. Athena Scientific, 2005.

[11] D. Bertsekas. Dynamic programming and suboptimal control: A survey from ADP to MPC. *European Journal of Control*, 11(4–5):310–334, October 2005.

[12] D. Bertsekas. Rollout algorithms for constrained dynamic programming. Technical report, Laboratory for Information and Decision Systems, MIT, 2005.

[13] D. Bertsekas. *Dynamic Programming and Optimal Control: Volume 2*. Athena Scientific, 2007.

[14] D. Bertsekas and S. Shreve. *Stochastic optimal control: The discrete-time case*. Athena Scientific, 1996.

[15] D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[16] D. Bertsimas and A. Lo. Optimal control of execution costs. *Journal of Financial Markets*, 1(1):1–50, April 1998.

[17] J. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research and Financial Engineering. Springer, 1997.

[18] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994.

[19] 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.

[20] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[21] L. Breiman. Optimal gambling systems for favorable games. In *Proc. 4th Berkeley Symp. Math. Statist. Probab.*, volume 1, pages 65–78, 1961.

[22] D. Brown and J. Smith. Dynamic portfolio optimization with transaction costs: Heuristics and dual bounds. *Management Science*, 57(10):1752–1770, October 2011.

[23] G. Calafiore. Multi-period portfolio optimization with linear control policies. *Automatica*, 44(10):2463–2473, October 2008.

[24] G. Calafiore. An affine control method for optimal dynamic asset allocation with transaction costs. *SIAM J. Control Optim.*, 48(4):2254–2274, June 2009.

[25] G. Calafiore. Random convex programs. *SIAM J. Optim.*, 20(6):3427–3464, December 2010.

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

[27] T. Cover. Universal portfolios. *Mathematical Finance*, 1(1):1–29, January 1991.

[28] J. Cvitanić and I. Karatzas. Hedging and portfolio optimization under transaction costs: A Martingale approach. *Mathematical Finance*, 6(2):133–165, April 1996.

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

[30] D. De Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, November 2003.

[31] E. Denardo. *Dynamic Programming: Models and Applications*. Prentice-Hall, 1982.

[32] V. Desai, V. Farias, and C. Moallemi. Pathwise optimization for optimal stopping problems. *Management Science*, June 2012.

[33] B. Dumas and E. Luciano. An exact solution to a dynamic portfolio choice problem under transaction costs. *The Journal of Finance*, 46(2):577–595, June 1991.

[34] J. Dupačová, J. Hurt, and J. Štėpán. *Stochastic modeling in economics and finance*. Applied optimization. Kluwer Academic Publishers, 2002.

[35] L. El Ghaoui and S. Niculescu. *Advances in linear matrix inequality methods in control*. Advances in design and control. SIAM, 2000.

[36] C. Garcia, D. Prett, and M. Morari. Model predictive control: Theory and practice. *Automatica*, 25(3):335–348, May 1989.

[37] N. Gârleanu and L. Pedersen. Dynamic trading with predictable returns and transaction costs. Manuscript, available at `http://pages.stern.nyu.edu/~lpederse/papers/DynamicTrading.pdf`, 2011.

[38] D. Goldsmith. Transactions costs and the theory of portfolio selection. *Journal of Finance*, 31(4):1127–39, September 1976.

[39] G. Goodwin, M. Seron, and J. De Doná. *Constrained control and estimation*. Springer, 2005.

[40] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, April 2011.

[41] M. Haugh, L. Kogan, and J. Wang. Evaluating portfolio policies: A duality approach. *Operations Research*, 54(3):405–418, June 2006.

[42] O. Hernandez-Lerma and J. Lasserre. Linear programming approximations for Markov control processes in metric spaces. *Acta Applicandae Mathematicae*, 51:123–139, 1998.

[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] G. Iyengar and T. Cover. Growth optimal investment in horse race markets with costs. *IEEE Transactions on Information Theory*, 46(7):2675–2683, November 2000.

[45] K. Judd. *Numerical Methods in Economics*. The MIT Press, 1998.

[46] Y. Kabanov. Hedging and liquidation under transaction costs in currency markets. *Finance and Stochastics*, 3(2):237–248, 1999.

[47] Y. Kabanov, M. Rásonyi, and C. Stricker. On the closedness of sums of convex cones in L^0 and the robust no-arbitrage property. *Finance and Stochastics*, 7(3):403–411, 2003.

[48] Y. Kabanov and C. Stricker. The Harrison–Pliska arbitrage pricing theorem under transaction costs. *Journal of Mathematical Economics*, 35(2):185–196, 2001.

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

[50] P. Krokhmal, J. Palmquist, and S. Uryasev. Portfolio optimization with conditional value-at-risk objective and constraints. *Journal of Risk*, 4(2):11–27, 2002.

[51] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004.

[52] W. Kwon and S. Han. *Receding Horizon Control*. Springer-Verlag, 2005.

[53] X. Li, X. Zhou, and A. Lim. Dynamic mean-variance portfolio selection with no-shorting constraints. *SIAM J. Control Optim.*, 40(5):1540–1555, January 2002.

[54] B. Lincoln and A. Rantzer. Relaxing dynamic programming. *IEEE Transactions on Automatic Control*, 51(8):1249–1260, August 2006.

[55] A. Lo and M. Mueller. Warning: Physics envy may be hazardous to your wealth!, 2010.

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

[57] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, November 1998.

[58] J. Löfberg. YALMIP: A toolbox for modeling and optimization in Matlab. In *Proceedings of the CACSD Conference*, 2004.

[59] J. Maciejowski. *Predictive Control with Constraints*. Prentice-Hall, 2002.

[60] A. Manne. Linear programming and sequential decisions. *Management Science*, 6(3):259–267, April 1960.

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

[62] J. Mattingley and S. Boyd. Real-time convex optimization in signal processing. *IEEE Signal Processing Magazine*, 23(3):50–61, June 2009.

[63] J. Mattingley and S. Boyd. Automatic code generation for real-time convex optimization. In D. P. Palomar and Y. C. Eldar, editors, *Convex optimization in signal processing and communications*, pages 1–41. Cambridge University Press, 2010.

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

[65] J. Mattingley, Y. Wang, and S. Boyd. Code generation for receding horizon control. In *IEEE Multi-Conference on Systems and Control*, pages 985–992, 2010.

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

[67] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, June 2000.

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

[69] S. Meyn and R. Tweedie. *Markov chains and stochastic stability*. Communications and control engineering. Cambridge University Press, 2009.

[70] C. Moallemi and M. Sağlam. Dynamic portfolio choice with linear rebalancing rules. Available at `http://www.columbia.edu/~ms3760/linear.pdf`, 2012.

[71] J. Mossin. Optimal multiperiod portfolio policies. *The Journal of Business*, 41(2):215–229, April 1968.

[72] Y. Nesterov and A. Nemirovsky. *Interior-Point Polynomial Methods in Convex Programming*. SIAM, 1994.

[73] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 1999.

[74] B. O'Donoghue, G. Stathopoulos, and S. Boyd. A splitting method for optimal control. *IEEE Trans. Autom. Control*, 2013. To appear.

[75] B. O'Donoghue, Y. Wang, and S. Boyd. Min-max approximate dynamic programming. In *Proceedings IEEE Multi-Conference on Systems and Control*, pages 424–431, September 2011.

[76] H. Pham. *Continuous-time stochastic control and optimization with financial applications*. Spinger Series in Stochastic modelling and applied probability. Springer, 2009.

[77] F. Potra and S. Wright. Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1–2):281–302, 2000.

[78] W. Powell. *Approximate dynamic programming: Solving the curses of dimensionality*. John Wiley & Sons, 2007.

[79] A. Prékopa. *Stochastic programming*. Mathematics and its applications. Kluwer Academic Publishers, 1995.

[80] M. Rami and L. El Ghaoui. LMI optimization for nonstandard Riccati equations arising in stochastic control. *IEEE Trans. Autom. Control*, 41(11):1666–1671, November 1996.

[81] C. V. Rao, S. J. Wright, and J. B. Rawlings. Application of interior point methods to model predictive control. *Journal of optimization theory and applications*, 99(3):723–757, November 2004.

[82] R. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *The Journal of Risk*, 2(3):21–42, 2000.

[83] S. Ross. *Introduction to Stochastic Dynamic Programming: Probability and Mathematical*. Academic Press, 1983.

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

[85] W. Schachermayer. The fundamental theorem of asset pricing under proportional transaction costs in finite discrete time. *Mathematical Finance*, 14(1):19–48, January 2004.

[86] A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on stochastic programming: Modeling and theory*. MPS-SIAM series on optimization. Society for Industrial and Applied Mathematics, 2009.

[87] J. Skaf and S. Boyd. Multi-period portfolio optimization with constraints and transaction costs. `http://www.stanford.edu/~boyd/papers/dyn_port_opt.html`, 2008. Manuscript.

[88] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11:625–653, 1999. Software available at `http://sedumi.ie.lehigh.edu/`.

[89] M. Sznaier, R. Suarez, and J. Cloutier. Suboptimal control of constrained nonlinear systems via receding horizon constrained control Lyapunov functions. *International Journal on Robust and Nonlinear Control*, 13(3–4):247–259, March 2003.

[90] C. Tapiero. *Applied stochastic models and control for finance and insurance*. Kluwer, 1998.

[91] J. Tobin. The theory of portfolio selection. In F. Hahn and F. Brechling, editors, *The Theory of Interest Rates*. Macmillan, 1965.

[92] K. Toh, M. Todd, and R. Tütüncü. SDPT3—A Matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1):545–581, 1999.

[93] U. Topcu, G. Calafiore, and L. El Ghaoui. Multistage investments with recourse: A single-asset case with transaction costs. In *Proceedings of the 47th Conference on Decision and Control*, pages 2398–2403, Cancun, Mexico, 2008.

[94] R. Tütüncü, K. Toh, and M. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming*, 95(2):189–217, 2003.

[95] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[96] Y. Wang and S. Boyd. Performance bounds for linear stochastic control. *Systems & Control Letters*, 58(3):178–182, March 2009.

[97] Y. Wang and S. Boyd. Approximate dynamic programming via iterated Bellman inequalities. `http://www.stanford.edu/~boyd/papers/adp_iter_bellman.html`, 2010. Manuscript.

[98] Y. Wang and S. Boyd. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, March 2010.

[99] Y. Wang and S. Boyd. Fast evaluation of quadratic control-Lyapunov policy. *IEEE Transactions on Control Systems Technology*, 19(4):939–946, July 2011.

[100] Y. Wang and S. Boyd. Performance bounds and suboptimal policies for linear stochastic control via LMIs. *International Journal of Robust and Nonlinear Control*, 21(14):1710–1728, September 2011.

[101] P. Whittle. *Optimization Over Time: Dynamic Programming and Stochastic Control*. John Wiley & Sons, 1982.

[102] S. Wright. *Primal-Dual Interior-Point Methods*. SIAM, 1997.

[103] X. Zhou and D. Li. Continuous-time mean-variance portfolio selection: A stochastic LQ framework. *Applied Mathematics & Optimization*, 42(1):19–33, 2000.

[104] W. Ziemba and R. Vickson. *Stochastic Optimization Models in Finance*. World Scientific, 2006.