Successive Linear Approximation Solution of Infinite Horizon Dynamic Stochastic Programs

John R. Birge * and Gongyun Zhao †

September 10, 2003

*Department of Industrial Engineering and Management Sciences, Robert R. McCormick School of Engineering and Applied Science, Northwestern University, Evanston, IL, 60208, USA (jrbirge@northwestern.edu). His work was supported in part by the National Science Foundation under Grant DMI-0200429.

†Department of Mathematics, National University of Singapore, Singapore. (matzgy@nus.edu.sg).

Successive Linear Approximation Solution of Infinite Horizon Dynamic Stochastic Programs

Abstract

Models for long-term planning often lead to infinite horizon stochastic programs that offer significant challenges for computation. Finite-horizon approximations are often used in these cases but they may also become computationally difficult. In this paper, we directly solve for stationary solutions of infinite horizon stochastic programs. We show that a successive linear approximation method converges to an optimal stationary solution for the case with convex objective, linear dynamics, and feasible continuation.

Keywords: stochastic programming, dynamic programming, infinite horizon, linear approximation, cutting planes

AMS Classification: 65K05, 90C15, 90C39, 91B28

1 Introduction

Many long-term planning problems can be expressed as infinite horizon stochastic programs. The infinite horizon often arises because of uncertainty about any specific endpoint (e.g., the lifetime of an individual or organization). Solving such problems with multiple decision variables and random parameters presents obvious computational difficulties. A common technique is to use a finite-horizon approximation, but even these problems become quite difficult for practical size.

The approach in this paper is to assume stationary data and to solve for the infinite horizon value function directly. A motivating example is an infinite horizon portfolio problem, which involves decisions on amounts to invest in different assets and amounts to consume over time. For simple cases, such as those in Samuelson [13] for discrete-time and Merton [10] for continuous-time, optimality conditions can be solved directly but general transaction costs and constraints on consumption or investment require more complex versions of the form in the infinite horizon problems considered here.

The problems considered here also relate to the dynamic programming literature (see, for example, Bertsekas [2]) and particularly to methods for solving partially observed Markov decision processes (see the survey in Lovejoy [8]). Our method is most similar to the piecewise linear construction in Smallwood and Sondik [14] for finite horizons and the bounding approximations used in Lovejoy [9] for both finite and infinite horizons. The main differences in our model are that we do not assume a finite action space and do not use finite horizon approximations. Our method also does not explicitly find a policy or approximate the state space with a discrete grid; instead, we use the convexity of the value function and the contraction properties of the dynamic programming operator in a form of approximate value iteration.

In the next section, we describe the general problem setting. Section 3 describes the algorithm, while Section 4 describes the convergence properties. Section 5 discusses the construction of the value function domain as required for algorithm convergence. Section 6 describes a portfolio example and implementation of the algorithm. Section 7 concludes with a discussion of further issues.

2 Problem Setting

We seek to find the value function V^* of the infinite horizon problem

$$\begin{aligned} V^*(x) = \quad & \min_{y_1,y_2,\ldots} \quad E_{\xi_0,\xi_1,\ldots} \sum_{t=0}^{\infty} \delta^t c_t(x_t, y_t) && (2.1) \\ & s.t. \quad x_{t+1} = A_t x_t + B_t y_t + b_t, \qquad \text{for } t = 0,1,2,\ldots, \\ & \qquad\quad x_0 = x, \end{aligned}$$

where $\xi_t = (A_t, B_t, b_t)$, $t = 0,1,2,\ldots$, are random vectors and $0 < \delta < 1$ is a discount factor. The above problem can be represented as

$$\begin{aligned} \min_{y_0} \quad & \{c_0(x_0,y_0) + \delta E_{\xi_0} \min_{y_1}\{c_1(x_1,y_1) + \delta E_{\xi_1} \min_{y_2}\{c_2(x_2,y_2) + \ldots\}\}\} \\ s.t. \quad & x_{t+1} = A_t x_t + B_t y_t + b_t, \qquad \text{for } t = 0,1,2,\ldots, \\ & x_0 = x. \end{aligned}$$

Here E_{ξ_t} has the same meaning as $E_{x_{t+1}}$.

In this paper we consider a simple version of (2.1), namely, $c_t = c$ and $(A_t, B_t, b_t) = (A, B, b)$, for all $t = 0,1,2,\ldots$. For the presentation below, we assume that $\xi = (A, B, b)$ is a discrete random vector with $p_i = Prob(\xi = (A_i, B_i, b_i))$, $i = 1,\ldots,L$. (The general algorithm does not require finite realizations but practical implementations make this assumption necessary.)

The value function V^* defined by (2.1) is a solution of $V = M(V)$, where the map M (often called the *dynamic programming operator*) is defined by

$$M(V)(x) \quad = \quad \min_{y}\{c(x,y) + \delta E_{\xi} V(Ax + By + b)\}$$

$$= \min_y \{c(x,y) + \delta \sum_{i=1}^{L} p_i V(A_i x + B_i y + b_i)\}. \quad (2.2)$$

Note: The problem of finding a solution of $V = M(V)$ and the infinite horizon problem (2.1) are different. The value function V^* defined by the infinite horizon problem is a solution to $V = M(V)$; however, the equation $V = M(V)$ may have many solutions. We will discuss this later. For the time being, we only need to know that a solution of $V = M(V)$ is equal to V^* if its effective domain coincides with $dom(V^*)$.

More precisely, let $D^* = dom(V^*)$ be compact and convex. Let $\mathcal{B}(D^*)$ be the Banach space of all functions finite on D^* and equipped with the norm $\|f\|_{D^*} = \sup_{x\in D^*}\{|f(x)|\}$; then, the equation $V = M(V)$ has a unique solution on $\mathcal{B}(D^*)$, which is V^*.

We will propose a cutting plane method to construct a piecewise linear value function V^k which approximately solves the problem $V = M(V)$. The cutting plane method can only be applied to convex problems; thus, we assume the cost function c be convex throughout the paper.

For any functions f and $g : R^n \to R \cup \{+\infty\}$, we say $f \geq g$ if $f(z) \geq g(z)$ for all $z \in R^n$.

3 A cutting plane method

In this and the next sections we assume that the domain $D^* = dom(V^*)$ is known and is compact and convex. All functions are regarded as elements in $\mathcal{B}(D^*)$; thus, we will only define values of functions on D^*.

Algorithm 1

1. Initialization: Find a piecewise linear convex function $V^0 \in \mathcal{B}(D^*)$ satisfying $V^0 \leq V^*$. Set $k \leftarrow 0$.
2. If $V^k \geq M(V^k)$, stop, V^k is the solution. Otherwise, find a point $\bar{x} \in D^*$ with $V^k(\bar{x}) < M(V^k)(\bar{x})$.
3. Find a supporting hyperplane of $M(V^k)$ at $\bar{x}$, say $t = Q^{k+1}x + q^{k+1}$. Define $V^{k+1}(x) = \max\{V^k(x), Q^{k+1}x + q^{k+1}\}$.
   $k \leftarrow k + 1$. Go to Step 2.

Details of the algorithm

Step 1: Usually, we can find V^0 easily. For instance, if $c(x,y) \geq c_0$ for all (x,y) in its domain, then we can choose $V^0(x) = c_0/(1-\delta)$, a constant function on D^*.

It is clear that

$$V^*(x) \geq \sum_{i=0}^{\infty} \delta^i c_0 = V^0(x), \quad \forall\, x \in D^*.$$

Step 2 consists of two parts. Part 1 is the valuation of $M(V^k)(x)$ and Part 2 describes a method for finding a point $\bar{x} \in D^k$ with $V^k(\bar{x}) < M(V^k)(\bar{x})$.

Part 1. Assume that V^k is defined by k linear cuts, i.e., for any $x \in D^*$,

$$\begin{aligned} V^k(x) &= \max\{Q^i x + q^i : i = 1, \ldots, k\} \\ &= \min\{\theta \mid \theta \geq Q^i x + q^i,\, i = 1, \ldots, k\}. \end{aligned}$$

Then

$$\begin{aligned} M(V^k)(x) &= \min_y \{c(x,y) + \delta \sum_{j=1}^{L} p_j V^k(A_j x + B_j y + b_j)\} \\ &= \min_{y,\theta} \{c(x,y) + \delta \sum_{j=1}^{L} p_j \theta^i \mid \theta^i \geq Q^i z^j + q^i,\, z^j = A_j x + B_j y + b_j \in D^*, \\ &\qquad\qquad i = 1, \ldots, k;\, j = 1, \ldots, L\}. \end{aligned} \tag{3.1}$$

Part 2. We seek to find $\bar{x}$ by approximately minimizing $V^k(x) - M(V^k)(x)$ on D^*. Notice that $V^k - M(V^k)$ is a d.c. function (difference of two convex functions) on D^*. There are rich results on solving d.c. programs, originated by Horst and Tuy [5]. We will propose a method described in [4].

$$\min_{x \in D^*} \qquad V^k(x) - M(V^k)(x)$$

is equivalent to

$$\begin{array}{rl} \min & x_{n+1} \\ x, x_{n+1}: & V^k(x) - M(V^k)(x) - x_{n+1} \leq 0, \\ & x \in D^*, \end{array}$$

which is equivalent to

$$\begin{array}{rl} \min & x_{n+1} \\ x, x_{n+1}, x_{n+2}: & V^k(x) - x_{n+1} - x_{n+2} \leq 0, \\ & M(V^k)(x) - x_{n+2} \geq 0 \\ & x \in D^*, \end{array}$$

which is equivalent to

$$\begin{array}{ll} \min & x_{n+1} \\ x, x_{n+1}, x_{n+2}: & Q^i x + q^i - x_{n+1} - x_{n+2} \le 0, \qquad i = 1, \dots, k, \\ & x \in D^*, \\ & M(V^k)(x) - x_{n+2} \ge 0. \end{array}$$

The first $k+1$ constraints define a polyhedral set, denoted by D. (In order to use the algorithm described in [4], D should be bounded. This can be done by adding appropriate lower and upper bounds on x_{n+1} and x_{n+2}, without changing the solution of the minimization problem.) The function in the $k+2$-nd constraint is convex, denoted by g. Such a program can be solved by Algorithm 4.1 in [4]. Here we describe it briefly. The algorithm solves

$$\begin{array}{ll} \min & c^T x \\ s.t. & x \in D,\, g(x) \ge 0. \end{array} \tag{3.2}$$

Initialization:

Solve $\min\{c^T x : x \in D\}$ to obtain $x^0 \in D$. Assume $g(x^0) < 0$ (otherwise, x^0 is optimal solution to (3.2)). Construct a simple polytope S_0, e.g., a simplex, containing D, such that x^0 is a vertex of S_0; compute the vertex set $V(S_0)$ of S_0. $k \leftarrow 0$.

Iterations:

Choose $z \in V(S_k)$ satisfying $g(z) = \max\{g(x) : x \in V(S_k)\}$.

If $g(z) = 0$ and $z \in D$, then stop; z is an optimal solution.

Otherwise, apply the simplex algorithm with starting vertex z to solve $\min\{c^T x : x \in S_k\}$ until an edge $[u, v]$ of S_k is found such that $g(u) \ge 0$ and $g(v) < 0$ and $c^T v < c^T u$; compute the intersection point s of $[u, v]$ with $\{x : g(x) = 0\}$.

If $s \in D$ then $S_{k+1} \leftarrow S_k \cap \{x : c^T x \le x^T s\}$.

If $s \notin D$ then $S_{k+1} \leftarrow S_k \cap \{x : l(x) \le 0\}$, where $l(x) \le 0$ is one of the linear constraints defining D satisfying $l(s) > 0$.

$k \leftarrow k + 1$, repeat.

Step 3: Let $D^* = \{x \mid \mathbf{F}x \le \mathbf{f}\}$ and

$$V^k(x) = \begin{cases} \min\{\theta \mid \mathbf{Q}x + \mathbf{q} \le \theta e\} & \text{if } \mathbf{F}x \le \mathbf{f}, \\ +\infty & \text{otherwise,} \end{cases}$$

where $e = (1, \dots, 1)^T$.

Let $(\bar{y}, \bar{\theta})$ and $\{(\bar{\lambda}_j, \bar{\mu}_j) : j = 1, \dots, L\}$ be an optimal solution and optimal multipliers of the problem:

$$\min_{y,\theta}\{c(\bar{x}, y) + \delta \sum_{j=1}^{L} p_j \theta^j \mid \mathbf{Q}z^j + \mathbf{q} \le \theta^j e,\, \mathbf{F}z^j \le \mathbf{f},\, z^j = A_j \bar{x} + B_j y + b_j, j = 1, \dots, L\}.$$

Then

$$\xi = \nabla_x c(\bar{x}, \bar{y}) + \sum_{j=1}^{L} (\bar{\lambda}_j^T \mathbf{Q} A_j + \bar{\mu}_j^T \mathbf{F} A_j)$$

is a subgradient of $M(V^k)$ at $\bar{x}$. (See Appendix for the proof.)

A supporting hyperplane of $M(V^k)$ at $\bar{x}$ is

$$t \;=\; M(V^k)(\bar{x}) + \xi(x - \bar{x})$$

That is,

$$Q^{k+1} = \xi, \qquad q^{k+1} = M(V^k)(\bar{x}) - \xi\bar{x}.$$

4 Convergence

In this section we assume D^* is a polytope.

Theorem 4.1 *M is a contraction on $\mathcal{B}(D^*)$.*

Proof. The proof can be found, e.g., in Theorem 3.8 of [7]. □

The unique solution of $V = M(V)$ on $\mathcal{B}(D^*)$ is V^*. We also denote $\mathcal{B} = \mathcal{B}(D^*)$.

Lemma 4.2 *$M(V)$ is convex if V is convex.*

Proof. (see Corollary 3.11 in [7].)

The following theorem is crucial for the convergence.

Theorem 4.3 *Let $V \in \mathcal{B}$. If $M(V) \leq V \leq V^*$, then $V = V^*$.*

Proof. It follows from $M(V) \leq V$ that for all $x \in D^*$

$$\begin{aligned} M(M(V))(x) &= \min_y \{c(x,y) + EM(V)(Ax + By + b)\} \\ &\leq \min_y \{c(x,y) + EV(Ax + By + b)\} \\ &= M(V)(x). \end{aligned}$$

This implies $V \geq M(V) \geq M^2(V) \geq \ldots \geq M^k(V) \geq \ldots$. Since M is a contraction on $\mathcal{B}$ by Theorem 4.1, $M^k(V) \to V^*$. This shows that $V \geq V^*$. Now we have $V^* \geq V \geq V^*$ which implies $V = V^*$. □

If Algorithm 1 stops at an iteration with $V^K \geq M(V^K)$, then we have $V^K = V^*$ by the above theorem.

If the algorithm does not terminate in a finite number of iterations, then the convergence of V^k to V^* is not so obvious. We notice that each cut is related to a testing point $\bar{x}$. Such a process can lead to pointwise convergence, but not necessarily uniform convergence. A catch is that the limit of a pointwise convergent sequence $\{V^k\}$ may not be V^*, because V^k may only be updated in some area of the domain but not over the full domain. Our convergence analysis shall answer the following two questions. Under what conditions can the sequence $\{V^k\}$ converge uniformly? If $\{V^k\}$ converges uniformly, is the limit function equal to V^*? Our main condition for uniform convergence is that D^* is a polytope. Indeed, if D^* is an arbitrary convex compact set, one can construct a monotone increasing sequence of convex functions $\{V^k\}$ which converges pointwise but not uniformly. For the second question, we will give a positive answer. The result is presented in Theorem 4.6.

Lemma 4.4 *$V^k \leq V^{k+1} \leq V^*$.*

Proof. From Step 2, it is obvious that $V^k \leq V^{k+1}$.

The initial function $V^0 \leq V^*$. Suppose $V^k \leq V^*$, then $M(V^k) \leq M(V^*) = V^*$, thus for any $x \in D^*$

$$V^{k+1}(x) \leq \max\{V^k(x),\, M(V^k)(x)\} \leq V^*(x).$$

□

Lemma 4.5 *If $f^k \leq f^{k+1}$, $f^k \to \bar{f}$ pointwise on a compact set $D \subset R^n$, and f^k and $\bar{f}$ are all continuous on D, then $f^k \to \bar{f}$ uniformly on D.*

Proof. If uniform convergence is not true, then there exists a constant $\epsilon_0 > 0$, and, for each k, there exists $x^k \in D$ such that

$$\bar{f}(x^k) - f^k(x^k) \geq \epsilon_0.$$

Let $x^k \to \bar{x} \in D$ (otherwise, use a convergent subsequence). For any l and any $k > l$,

$$\bar{f}(x^k) - f^l(x^k) \geq \bar{f}(x^k) - f^k(x^k) \geq \epsilon_0.$$

By continuity, $\bar{f}(\bar{x}) - f^l(\bar{x}) \geq \epsilon_0 > 0$, which contradicts that $f^l(\bar{x}) \to \bar{f}(\bar{x})$. □

Theorem 4.6 *Suppose that D^* is a polytope and suppose that, at the k-th iteration, a point $x^k \in D^*$ is selected such that*

$$M(V^k)(x^k) - V^k(x^k) \geq \alpha \max\{M(V^k)(x) - V^k(x) \mid x \in D^*\}$$

for some constant $\alpha > 0$. Then $V^k \to V^*$ uniformly.

Proof. First, suppose that Algorithm 1 stops at a finite iteration K with $V^K \geq M(V^K)$. By Lemma 4.4 $V^K \leq V^*$. Thus by Theorem 4.3, $V^K = V^*$.

Now suppose Algorithm 1 generates a sequence $\{V^k\}$ which converges to $\tilde{V}$ pointwise.

Because $V^k \leq V^{k+1} \to \tilde{V}$, $epi(\tilde{V}) = \cap_k epi(V^k)$, which is a closed set since every V^k is closed. Thus $\tilde{V}$ is a closed convex function on D^*. By our assumption, D^* is a polytope; thus, by Theorem 10.2 in [11], $\tilde{V}$ is continuous on D^*. By Lemma 4.5, $V^k \to \tilde{V}$ uniformly on D^*.

Assume that there exists an $\hat{x} \in D^*$ such that

$$M(\tilde{V})(\hat{x}) - \tilde{V}(\hat{x}) = 2\sigma > 0.$$

By Theorem 4.1, $M(V^k) \to M(\tilde{V})$ uniformly on D^* since $V^k \to \tilde{V}$ uniformly. Thus there exists a $\hat{k}$ such that $M(V^k)(\hat{x}) \geq M(\tilde{V})(\hat{x}) - \sigma$, for all $k \geq \hat{k}$. This yields

$$M(V^k)(\hat{x}) - V^k(\hat{x}) \geq M(\tilde{V})(\hat{x}) - \sigma - \tilde{V}(\hat{x}) \geq \sigma.$$

Since the supporting hyperplane at iteration k satisfies $Q^{k+1}x^k + q^{k+1} = M(V^k)(x^k)$, we have

$$\begin{aligned} V^{k+1}(x^k) - V^k(x^k) &= M(V^k)(x^k) - V^k(x^k) \\ &\geq \alpha[M(V^k)(\hat{x}) - V^k(\hat{x})] \\ &\geq \alpha\sigma. \end{aligned}$$

Thus

$$\tilde{V}(x^k) - V^k(x^k) \geq \alpha\sigma, \qquad \forall\, k \geq \hat{k}.$$

The above contradicts the uniform convergence of $V^k \to \tilde{V}$ on D^*.

The contradiction implies that

$$M(\tilde{V})(x) \leq \tilde{V}(x), \qquad \forall\, x \in D^*.$$

Then, by Theorem 4.3, $\tilde{V} = V^*$. This means that V^k converges to V^* uniformly on D^*. □

5 Construction of the domain D^*

This section discusses how to find D^*. We do not have a complete answer for this problem; nevertheless, the method proposed can find D^* in finite iterations for many cases.

Approximating D^*

Although the domain of a solution of $V = M(V)$ does not necessarily coincide with $D^* = dom(V^*)$ where V^* is the value function defined by (2.1), it does provide useful information for finding D^*. We first represent the domain of $M(V)$.

From (2.2) one can see that $x \in dom(M(V))$ if and only if there exists a y such that $c(x, y) < +\infty$ and $A_i x + B_i y + b_i \in dom(V)$ for all $i = 1, \ldots, L$. Denote

$$\begin{aligned} D_c(x) &= \{y \mid c(x, y) < +\infty\} \\ G(x, D) &= \{y \mid A_i x + B_i y + b_i \in D, \forall i = 1, \ldots, L\} \end{aligned}$$

Then, $x \in dom(M(V))$ if and only if $D_c(x) \cap G(x, dom(V)) \neq \emptyset$.

Denote

$$\Gamma(D) = \{x \mid D_c(x) \cap G(x, D) \neq \emptyset\}.$$

Then

$$dom(M(V)) = \Gamma(dom(V)).$$

The following lemma shows that if $dom(V) \subseteq dom(M(V))$ then $dom(V) \subseteq dom(V^*)$.

Lemma 5.1 *Suppose c is bounded on its domain. If $D \subseteq \Gamma(D)$ then $D \subseteq D^*$.*

Proof. For any $x_t \in D$, we have $x_t \in \Gamma(D)$, thus

$$\exists y_t \in D_c(x_t) : A(\omega_t)x_t + B(\omega_t)y_t + b(\omega_t) \in D, \forall \omega_t \in \Omega. \tag{5.1}$$

For any $\bar{x} \in D$, let $x_0 = \bar{x}$. There exists y_0 satisfying (5.1). For each $\omega_0 \in \Omega$, let $x_1(\omega_0) = A(\omega_0)x_0 + B(\omega_0)y_0 + b(\omega_0)$. Since $x_1(\omega_0) \in D \subseteq \Gamma(D)$, there exists $y_1(\omega_0)$ satisfying (5.1), and so on. So we obtain a sequence $\{(x_t, y_t) : t = 0, 1, \ldots\}$ (here (x_t, y_t) are random vectors) satisfying $(x_t, y_t) \in dom(c)$ because $y_t \in D_c(x_t)$. Since c is bounded on its domain, $E \sum_{t=0}^{\infty} \delta^t c(x_t, y_t) < \infty$, thus $V^*(\bar{x})$, defined by (2.1), is finite, i.e., $\bar{x} \in D^*$. Therefore, $D \subseteq D^*$. □

Suppose we use a cutting plane method to construct D^* and start with a set $D \supseteq D^*$. The above lemma suggests that one should cut off a portion of D if $D \not\subseteq \Gamma(D)$.

Because

$$D^* \subseteq D_{cx} := \{x \mid D_c(x) \neq \emptyset\} = \{x \mid \exists y \text{ such that } c(x, y) < +\infty\},$$

we start with D_{cx} to find D^*. A generic cutting plane method which constructs D^* with this idea is as follows:

Algorithm 2

(0) Let $D^0 = D_{cx}$. $k = 0$.

(1) If $D^k \subseteq \Gamma(D^k)$, stop. Otherwise, find a cut $F^{k+1}x \leq f^{k+1}$ which cuts off a portion of $D^k \setminus D^*$.

(2) Let $D^{k+1} = D^k \cap \{x \mid F^{k+1}x \leq f^{k+1}\}$. $k \leftarrow k+1$. Repeat. □

The algorithm stops when a D^k with $D^k \subseteq \Gamma(D^k)$ is found. Question: is $D^k = D^*$?

Theorem 5.2 *If the algorithm terminates at a finite iteration K, then $D^K = D^*$.*

Proof. The algorithm starts with D_{cx} which contains D^*. No cut cuts off any point of D^*, thus $D^K \supseteq D^*$. When the algorithm stops with $D^K \subseteq \Gamma(D^K)$, Lemma 5.1 yields $D^K \subseteq D^*$. Thus, $D^K = D^*$. □

Generating cuts

Now we discuss Step (1) in detail.

Let $dom(c) = \{(x,y) : Tx + Wy \leq r\}$ and $D^k = \{x : F^i x \leq f^i : i = 1, \ldots, k\}$.

$\bar{x} \notin \Gamma(D^k)$ if and only if

$$\begin{aligned} T\bar{x} + Wy &\leq \ , r \\ F^i(A_j\bar{x} + B_j y + b_j) &\leq f^i, \quad i = 1, \ldots, k;\ j = 1, \ldots, L. \end{aligned}$$

has no feasible solution; then, by Farkas' Theorem, if and only if

$$\begin{aligned} \sum_{i=1}^{k}\sum_{j=1}^{L} \pi_{ij} F^i B_j + \lambda^T W = 0, \\ \sum_{i=1}^{k}\sum_{j=1}^{L} \pi_{ij}[F^i(A_j\bar{x} + b_j) - f^i] + \lambda^T(T\bar{x} - r) > 0, \\ \pi \geq 0, \lambda \geq 0, \end{aligned} \tag{5.2}$$

has a solution.

Thus, to find a point $\bar{x} \in D^k$ such that $\bar{x} \notin \Gamma(D^k)$ is equivalent to finding a triple $(\bar{x}, \lambda, \pi)$ satisfying $\bar{x} \in D^k$ and (5.2). (Note that finding a solution to (4.2) is equivalent to determining the sign of the supremum of an indefinite quadratic function subject to linear constraints, which would also require global optimization methods as in Horst et al. [4].)

Once a solution $(\bar{x}, \lambda, \pi)$ is found, one can construct a feasibility cut:

$$\sum_{i=1}^{k}\sum_{j=1}^{L} \pi_{ij}[F^i(A_j x + b_j) - f^i] + \lambda^T(Tx - r) \leq 0,$$

i.e.,

$$F^{k+1} = \sum_{i=1}^{k}\sum_{j=1}^{L} \pi_{ij} F^i A_j, \quad f^{k+1} = \sum_{i=1}^{k}\sum_{j=1}^{L} \pi_{ij}[f^i - F^i b_j] + \lambda^T r.$$

We can add an objective to find the "best" cut, e.g., in the sense that F^{k+1} is the normal direction of a facet of D^* (or, perhaps more realistically, a facet of $\Gamma(D^k)$).

The cut generated above cuts off a portion of D^k, but does not cut off any point in $\Gamma(D^k)$. A cutting plane method with such cuts may fail if $\Gamma^k(D_{cx}) = D^*$ does not occur in a finite number of steps.

Let us look at a simple example to see how $\Gamma^k(D_{cx})$ approximates D^*.

Example 1

Suppose

$$\begin{aligned} dom(c) &= \{(x,y) \mid x \in [-1,1]^2,\ y \in [-\beta,\beta]\} \\ Ax + By + b &= \alpha x + e_1 y, \qquad \text{(deterministic)} \end{aligned}$$

Here $e_1 = (1,0)^T$. Then

$$\begin{aligned} D_c(x) &= \begin{cases} [-\beta,\beta] & \text{if } x \in [-1,1]^2, \\ \emptyset & \text{otherwise.} \end{cases} \\ G(x,D) &= \{y \mid \alpha x + e_1 y \in D\}. \end{aligned}$$

Thus,

$$\begin{aligned} D_c(x) \cap G(x,D) \neq \emptyset \quad &\Longleftrightarrow \quad x \in [-1,1]^2,\ [-\beta,\beta] \cap \{y \mid \alpha x + e_1 y \in D\} \neq \emptyset, \\ &\Longleftrightarrow \quad x \in [-1,1]^2,\ (\alpha x + e_1[-\beta,\beta]) \cap D \neq \emptyset. \end{aligned} \tag{5.3}$$

For $0 < \alpha \leq 1$,

$$\begin{aligned} D_{cx} &= \{x : D_c(x) \neq \emptyset\} = [-1,1]^2, \\ \Gamma(D_{cx}) &= \{x \in [-1,1]^2 : (\alpha x + e_1[-\beta,\beta]) \cap [-1,1]^2 \neq \emptyset\}, \\ &\supseteq \{x \in [-1,1]^2 : \{\alpha x\} \cap [-1,1]^2 \neq \emptyset\}, \\ &= [-1,1]^2 = D_{cx}, \end{aligned}$$

thus, by Lemma 5.1, $D^* = [-1,1]^2 = D_{cx}$.

Note: For the case $\alpha = 1$, any subset D of $[-1,1]^2$ of the form $\{|x_1| \leq 1,\ |x_2| \leq t\}$ for some $t \leq 1$ satisfies $D = \Gamma(D)$. Thus M is a contraction on the Banach space $\mathcal{B}(D)$ and $V = M(V)$ has a solution (fixed point) V_D in $\mathcal{B}(D)$. This shows that the equation $V = M(V)$ may have many solutions.

For $1 < \alpha \leq 1 + \beta$,

$$\begin{aligned} D_{cx} &= [-1,1]^2, \\ \Gamma(D_{cx}) &= \{|x_1| \leq 1,\ |x_2| \leq 1/\alpha\}, \\ &\vdots \\ \Gamma^k(D_{cx}) &= \{|x_1| \leq 1,\ |x_2| \leq 1/\alpha^k\}, \end{aligned}$$

thus $D^* = \lim_{k\to\infty} \Gamma^k(D_{cx}) = \{|x_1| \leq 1,\ x_2 = 0\} \neq D_{cx}$.

For $\alpha > 1 + \beta$,

$$\begin{aligned} D_{cx} &= [-1,1]^2 \\ \Gamma(D_{cx}) &= \{|x_1| \leq \frac{1+\beta}{\alpha},\ |x_2| \leq 1/\alpha\} \\ \Gamma^2(D_{cx}) &= \{|x_1| \leq \frac{1+\beta+\alpha\beta}{\alpha^2},\ |x_2| \leq 1/\alpha^2\} \\ &\vdots \\ \Gamma^k(D_{cx}) &= \{|x_1| \leq \frac{1+\beta+\alpha\beta+\ldots+\alpha^{k-1}\beta}{\alpha^k},\ |x_2| \leq 1/\alpha^k\} \end{aligned}$$

thus $D^* = \lim_{k\to\infty} \Gamma^k(D_{cx}) = \{|x_1| \leq \frac{\beta}{\alpha-1},\ x_2 = 0\} \neq D_{cx}$. □

Remarks:

(i) In some case ($\alpha \leq 1$), $D^* = D_{cx}$, which can be obtained directly.

(ii) In some case ($\alpha > 1$), infinitely many iterations are required to reach D^*.

(iii) Suppose the cutting plane algorithm generates only one-side-cuts in the case of $\alpha > 1$, e.g., only the cuts $x_2 \geq -1/\alpha^k$. Then $D^k \not\to D^*$.

(iv) If we can directly generate the cut $x_2 \geq 0$ instead of infinitely many cuts $\{x_2 \geq -1/\alpha^k : k = 1, 2, \ldots\}$, then we can construct D^* in 2 (4) iterations for the problem with $\alpha \leq (>) 1 + \beta$. □

Generating the deepest cut

If a cut does not cut off any point in $\Gamma(D^k)$, then, as shown in Example 1, the cutting plane method may fail to approximate D^* even with infinitely many iterations. In order to fulfill the goal in Remark (iv), we must find a deeper cut (a smaller f^{k+1}), which cuts into $\Gamma(D^k)$, hopefully, reaching the boundary of D^*.

Given a D, suppose that we have obtained a cut $d^T x \leq t_0$ which cuts off a portion of $D \setminus \Gamma(D)$ (but does not cut into $\Gamma(D)$). Denote $D_t := D \cap \{d^T x \leq t\}$. If the plane $d^T x = t_0$ has touched the boundary of $\Gamma(D)$, then no point of $D_{t_0} \setminus \Gamma(D)$ can be cut off by any cut of the form $d^T x \leq t$ for arbitrary t. However, since $\Gamma(D_{t_0})$ is smaller than $\Gamma(D)$, a portion of $D_{t_0} \setminus \Gamma(D_{t_0})$ may be cut off by some cut $d^T x \leq t$. We wish to find $\bar{t} < t_0$ such that no point of $D_{\bar{t}} \setminus \Gamma(D_{\bar{t}})$ can be cut off by any cut of the form $d^T x \leq t$. In other words, the plane $d^T x = \bar{t}$ touches the boundary of $\Gamma(D_{\bar{t}})$ (a supporting plane of $\Gamma(D_{\bar{t}})$). Thus, for any $t > \bar{t}$, the plane $d^T x = t$ does not touch $\Gamma(D_t)$, and for any $t \leq \bar{t}$, the half-space $d^T x \leq t$ intersects with $\Gamma(D_t)$. The latter means that there exists $x \in \Gamma(D_t)$ satisfying $d^T x \geq t$. The latter interpretation suggests to determine $\bar{t}$ by the following linear program:

$$\bar{t} = \quad \max \quad \{t \mid d^T x \geq t,\, x \in \Gamma(D_t)\}. \tag{5.4}$$

Because $d^T x \leq t_0$ does not cut into $\Gamma(D)$ (then does not cut into $\Gamma(D_t)$), there exists no x satisfying $d^T x \geq t$ and $x \in \Gamma(D_t)$ if $t > t_0$. This implies that $\bar{t} \leq t_0$. Therefore, the cut $d^T x \leq \bar{t}$ is deeper than the cut $d^T x \leq t_0$.

On the other hand, the following lemma guarantees that the cut $d^T x \leq \bar{t}$ will not cut off any point in D^*.

Lemma 5.3 *Suppose that $D^* \subset D$. Let $\bar{t}$ be the optimal objective value of problem (5.4). Then $d^T x \leq \bar{t}$ is satisfied by all $x \in D^*$.*

Proof. Let

$$x^* = \mathrm{argmax}\{d^T x \mid x \in D^*\}, \qquad t^* = d^T x^*.$$

Because $x^* \in D^*$, there exist $\{(x_t, y_t) : t = 0, 1, 2, \ldots\}$ such that

$$\begin{aligned} V^*(x^*) &= E[\sum_{t=0}^{\infty} \delta^t c(x_t, y_t)] < \infty, \\ x_{t+1} &= A x_t + B y_t + b, \qquad x_0 = x^*. \end{aligned}$$

For any integer $K \geq 0$, let $\tilde{x}_l = x_{l+K}$ and $\tilde{y}_l = y_{l+K}$. Because $E[\sum_{t=K}^{\infty} \delta^{t-K} c(x_t, y_t)] < \infty$, we have

$$V^*(x_K) \leq E[\sum_{l=0}^{\infty} \delta^l c(\tilde{x}_l, \tilde{y}_l)] < \infty.$$

Therefore, $x_K \in D^*$.

Now $y_0 \in D_c(x^*)$ follows from $c(x^*, y_0) < \infty$ and $y_0 \in G(x^*, D^*)$ follows from $x_1 = A_i x^* + B_i y_0 + b_i \in D^*$ for every $i = 1, \ldots, L$. Thus $x^* \in \Gamma(D^*)$. Because $D^* \subseteq D$ and $D^* \subseteq \{d^T x \leq t^*\}$, we have $D^* \subseteq D_{t^*}$. Therefore, $x^* \in \Gamma(D_{t^*})$. This, together with $d^T x^* = t^*$, shows that (x^*, t^*) is a feasible point of (5.4); thus $t^* \leq \bar{t}$, from which the claim of the lemma follows. □

The following example shows the effect of the deepest cut.

Example 1

(Continued)

Consider $\alpha > 1$. Consider the cut of the form $-x_2 \leq t$ for some $t \in R$ to be determined. So, $d = (0, -1)^T$ in (5.4). Start with $D = D_{cx} = [-1, 1]^2$. Then

$$D_t = \{x \in [-1,1]^2 : -x_2 \leq t\}.$$

Linear program (5.4) is

$$\begin{aligned} \bar{t} = \quad & \max \quad t \\ & s.t. \quad -x_2 \geq t, \\ & \qquad -1 \leq \alpha x_1 + y \leq 1, \\ & \qquad -t \leq \alpha x_2 \leq 1, \\ & \qquad -\beta \leq y \leq \beta. \end{aligned}$$

A feasible solution must satisfy

$$-t/\alpha \leq x_2 \leq -t.$$

This can only be satisfied when $t \leq 0$ since $\alpha > 1$. Thus we have $\bar{t} = 0$. The cut $-x_2 \leq 0$ reaches the bottom of D^*. With one more cut from above ($d = (0, 1)^T$), D^* will be completely determined for the case of $1 < \alpha \leq 1 + \beta$.

For the case of $\alpha > 1 + \beta$, suppose we have $d = (1, 0)^T$. Then

$$D_t = \{x \in [-1,1]^2 : x_1 \leq t\}$$

Linear program (5.4) is

$$\begin{aligned} \bar{t} = \quad & \max \quad t, \\ & s.t. \quad x_1 \geq t, \\ & \qquad -1 \leq \alpha x_1 + y \leq t, \\ & \qquad -1 \leq \alpha x_2 \leq 1, \\ & \qquad -\beta \leq y \leq \beta. \end{aligned}$$

Feasible solutions must satisfy

$$t \leq x_1 \leq \frac{t + \beta}{\alpha}.$$

This implies

$$t \leq \frac{\beta}{\alpha - 1}.$$

Thus $\bar{t} = \frac{\beta}{\alpha-1}$. So we obtain a cut $x_1 \leq \frac{\beta}{\alpha-1}$ which cuts exactly to the boundary of D^* on the right. One more cut from the left ($d = (-1,0)^T$) will completely determine D^*. So, in total, we need only 4 cuts.

Under what conditions does $D^* = D_{cx}$ hold true?

The set D_{cx} is easy to determine. If $D_{cx} = D^*$, then Algorithm 2 is not needed. For what (A_i, B_i, b_i), $i = 1, \ldots, L$, does $D^* = D_{cx}$ hold true?

Lemma 5.4 *If $D_{cx} \subseteq \Gamma(D_{cx})$ then $D_{cx} = D^*$.*

Proof. $D_{cx} \subseteq \Gamma(D_{cx})$ implies $D_{cx} \subseteq D^*$ by Lemma 5.1. On the other hand, $D^* \subseteq D_{cx}$ always holds. Thus, $D_{cx} = D^*$. □

The condition is spelled out as

$$\forall x \in D_{cx}, \exists y \in D_c(x) \text{ such that } A_i x + B_i y + b_i \in D_{cx}, i = 1, \ldots, L.$$

This condition is not easy to check in general. For the simple problem as Example 1, we have a sufficient condition.

Corollary 5.5 *Consider*

$$M(V)(x) = \min_y \{c(x,y) + \delta \sum_{i=1}^{L} p_i V(\alpha_i x + B_i y + b_i)\}.$$

Assume that there exits an $\bar{x} \in D_{cx}$ such that for any $x \in D_{cx}$ there is a $y \in D_c(x)$ such that $\alpha_i \bar{x} + B_i y + b_i - \bar{x} = 0$ for all $i = 1, \ldots, L$. If $\alpha_i \leq 1$ for all $i = 1, \ldots, L$, then $D^* = D_{cx}$.

Proof. For any $x \in D_{cx}$ we have $y \in D_c(x)$ satisfying

$$\alpha_i x + B_i y + b_i \quad = \quad \alpha_i(x - \bar{x}) + \bar{x} \in D_{cx}.$$

Thus, $x \in \Gamma(D_{cx})$. This implies $D_{cx} \subseteq \Gamma(D_{cx})$. Thus, $D_{cx} = D^*$. □

Example 1 satisfies the condition in the lemma, letting $\bar{x} = 0$ and $y = 0$ for all x. Thus, for $\alpha \leq 1$ we have $D^* = D_{cx} = [-1,1]^2$.

6 Infinite horizon portfolio example

As noted in the introduction, this work was motivated by solving infinite horizon investment problems that face long-enduring institutions. We will demonstrate how the algorithm performs on a small example where an infinite horizon optimum can be found analytically (as done, for example, in [13] and [10]). The goal is to maximize the discounted expected utility of consumption over an infinite horizon.

The decisions in each period are how much to consume and how much to invest in a risky asset (or in a variety of assets).

The state variable x in this case corresponds to wealth or the current market value of all assets. The control variable y has two components, y_1, which corresponds to consumption, and y_2, which corresponds to the amount invested in a risky asset with random return ξ. The assumption in this model is that any remaining funds, after consuming y_1 and investing y_2 in the risky asset, are invested in a riskfree asset (e.g., U.S. Treasury bills) with known rate of return r. For this model, $c(x,y)$ is either ∞ if $x<0$ or $-y_1^\gamma/\gamma$, for some non-zero parameter $\gamma<1$, giving (the negative of) the common utility function with constant relative risk aversion (i.e., such that risk preferences do not depend on the level of wealth).

With these assumptions, $M(V)$ takes the following form (for $x \geq 0$):

$$M(V)(x)=\min_y\{-y_1^\gamma/\gamma+\delta\sum_{i=1}^{L}p_iV((1+r)x-(1+r)y_1+(\xi_i-r)y_2)\}. \tag{6.5}$$

where ξ_i is the ith realization of the random return with probability p_i. The solution V^* of $M(V)=V$ can be found analytically by observing that the optimal value function is proportional to x^γ (by, for example, considering the limiting case of a finite horizon problem). We then have that

$$\begin{aligned}
&\sum_{i=1}^{L}p_iV((1+r)x-(1+r)y_1+(\xi_i-r)y_2)\\
&=-K\sum_{i=1}^{L}p_i((1+r)x-(1+r)y_1+(\xi_i-r)y_2)^\gamma\\
&=-K(x-y_1)^\gamma\sum_{i=1}^{L}p_i((1+r)+(\xi_i-r)[y_2/(x-y_1)])^\gamma\\
&=-K(x-y_1)^\gamma\sum_{i=1}^{L}p_i((1+r)+(\xi_i-r)z)^\gamma,
\end{aligned}$$

where $z=\frac{y_2}{x-y_1}$ is the fractional risky investment after consuming y_1 and K is some positive constant. The optimal z^* then must solve

$$\sum_{i=1}^{L}p_i\gamma(\xi_i-r)((1+r)+(\xi_i-r)z)^{\gamma-1}=0, \tag{6.6}$$

which is independent of y_1. With $\bar{V}^*=-\sum_{i=1}^{L}p_i((1+r)+(\xi_i-r)z^*)^\gamma$, optimal y_1^* now must solve

$$-y_1^{\gamma-1}+\delta\gamma K\bar{V}^*(x-y_1)^{\gamma-1}=0 \tag{6.7}$$

or

$$y_1^* = x\frac{(\delta\gamma K\bar{V}^*)^{1/(\gamma-1)}}{1+(\delta\gamma K\bar{V}^*)^{1/(\gamma-1)}} = xw^*, \tag{6.8}$$

for an optimal consumption fraction w^*. The last step is to find K from $M(V) = V$, using

$$-x^\gamma((w^*)^\gamma/\gamma + \delta K\bar{V}^*(1-w^*)^\gamma) = M(V(x)) = V(x) = -Kx^\gamma \tag{6.9}$$

to obtain $K = \frac{((\delta\bar{V}^*)^{1/(\gamma-1)}-1)^{\gamma-1}}{\delta\gamma\bar{V}^*}$.

While this function can be found explicitly, various other constraints on investment (such as transaction costs and limits on consumption changes from period to period) make analytical solutions impossible. We use the analytical solution here to observe Algorithm 1's performance and convergence behavior. We also use the analytical solution to derive initial upper bounding approximations. For our test, we use the linear supports of V^* at $x = 0.1$ and $x = 10$ as initial cuts and restrict our search in x to the interval $[0.1, 10]$, although the feasible region is unbounded.

For our test, we used $\gamma = 0.03$, $r = 0.05$, and ξ_i chosen as a discrete approximation of the lognormal return distribution with mean return of 0.08 and standard deviation of 0.4. Algorithm 1 was implemented in MATLAB using `fmincon` to solve the optimization subproblems and a linesearch to find $\bar{x}$ in Step 2. We tried different values for the discount factor, δ. The results for $\delta = \frac{1}{1.25}$ appear in Figure 1, which includes V^0, V^5, V^{10}, V^{20}, V^{50}, V^{100}, and V^*. In this case, after 100 iterations, the approximation almost perfectly matches the true infinite-horizon value function.

With a larger δ, the value of K increases rapidly as $(\delta\bar{V}^*)^{1/(\gamma-1)}$ approaches one. For $\delta = \frac{1}{1.25}$, K is 155.6, while, for $\delta = \frac{1}{1.07}$, K is 466.3. The result is that larger δ values (corresponding to lower discount rates) require additional iterations of Algorithm 1 to approach V^*. The results for the same data as in Figure 1 except with $\delta = \frac{1}{1.07}$ appear in Figure 2. After 500 iterations, the approximating V^{500} agrees relatively well with V^* as shown in the figure but has not converged to nearly the same accuracy as the approximations (with fewer iterations) in Figure 1.

Understanding the numerical behavior of Algorithm 1 and finding mechanisms to speed convergence should be topics for further investigation. Comparing V^k to V^* in the figures shows how the algorithm forces closer approach in some areas of the curve over others. The behavior of the algorithm is generally to move along the curve V^k to create tighter cuts and then to repeat that process with less improvement on a new sequence of iterations. These observations suggest that procedures with multiple cut generation and tightening tolerances should be considered for accelerating convergence.

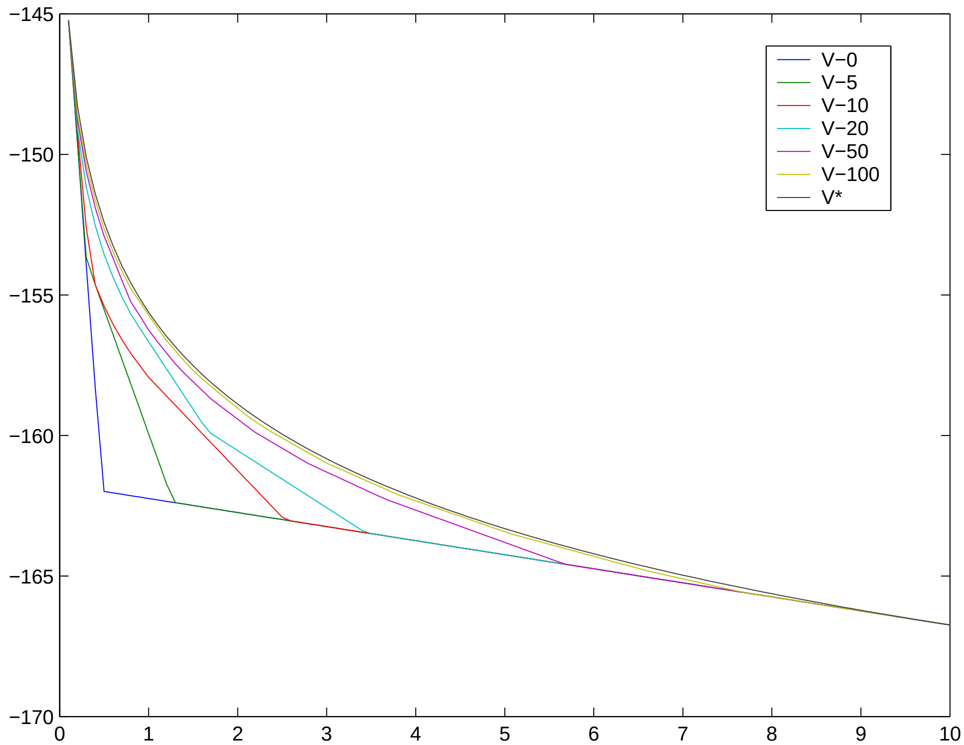

Figure 1: Value function approximations for portfolio example, $\delta = 1/1.25$.

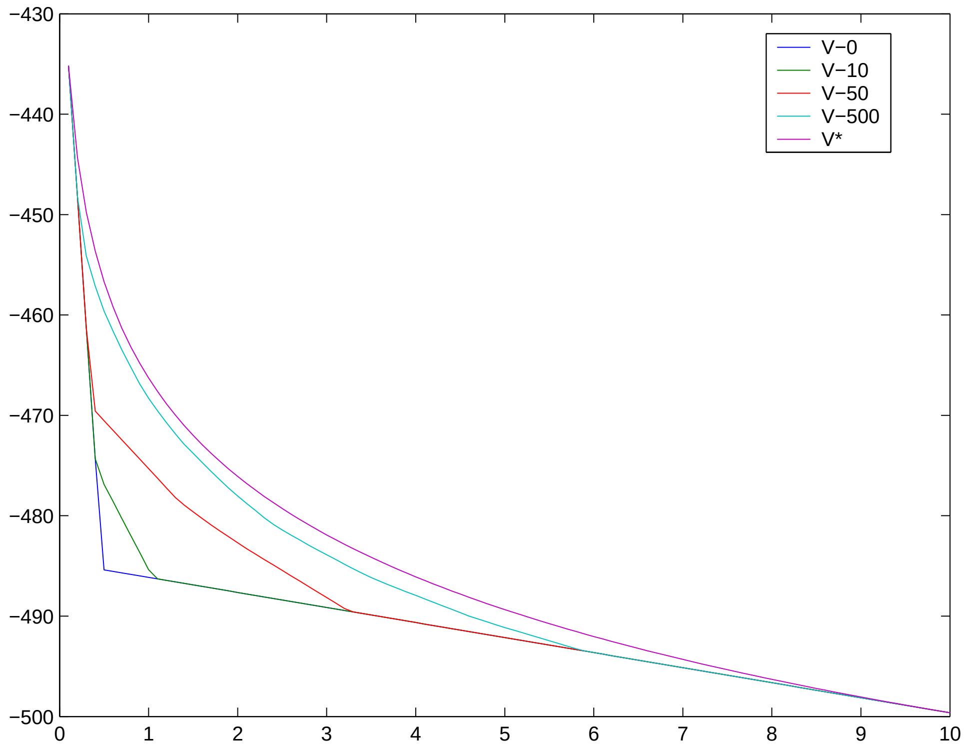

Figure 2: Value function approximations for portfolio example, $\delta = 1/1.07$.

7 Observations and future issues

We have described an algorithm for solving a general class of discrete-time convex infinite horizon optimization problems and demonstrated the method on a simple example. As the example demonstrates, many iterations may be required to achieve accuracy for highly nonlinear value functions. Since each iteration involves additional optimization steps, selection of $\bar{x}$ (or potentially multiple points on each iteration) has a critical effect on performance. We mentioned the d.c. methods as one possibility for finding good points but other methods that require fewer function evaluations may also be useful. These options require further study.

Our method also relies on identification of the feasible domain, D^*. In that case, we are left with the following questions:

(i) Under what condition is D^* a polytope?

(ii) Can Algorithm 2 terminate in a finite number of iterations if D^* is a polytope?

(iii) How can one modify the algorithm if D^* is not a polytope?

These questions and further implementation issues are subjects for future research.

8 Appendix

In this appendix, we will show

$$\xi = \nabla_x c(\bar{x}, \bar{y}) + \sum_{j=1}^{L} (\bar{\lambda}_j^T \mathbf{Q} A_j + \bar{\mu}_j^T \mathbf{F} A_j)$$

is a subgradient of $M(V)$ at $\bar{x}$, where

$$V(x) = \begin{cases} \min\{\theta \mid \mathbf{Q}x + \mathbf{q} \le \theta e\} & \text{if } \mathbf{F}x \le \mathbf{f}, \\ +\infty & \text{otherwise.} \end{cases}$$

Denote $\rho(x) = M(V)(x)$. Then

$$\begin{aligned} \rho(x) = \quad & \min \quad && c(x,y) + \delta \sum_{j=1}^{L} p_j \theta^j \\ & y, \theta : && \mathbf{Q}(A_j x + B_j y + b_j) + \mathbf{q} \le \theta^j e, \\ & && \mathbf{F}(A_j x + B_j y + b_j) \le \mathbf{f}, \qquad j = 1, \ldots, L, \\ = \quad & \max && h(\lambda, \mu; x) \\ & s.t. && \lambda_j^T e = \delta p_j, \qquad j = 1, \ldots, L, \\ & && \lambda \ge 0,\ \mu \ge 0, \end{aligned}$$

where $\lambda = (\lambda_1; \ldots; \lambda_L)$, $\mu = (\mu_1; \ldots; \mu_L)$, and

$$\begin{aligned} h(\lambda, \mu; x) = \quad \min_y \quad & c(x,y) + \sum_{j=1}^{L} [\lambda_j^T (\mathbf{Q}(A_j x + B_j y + b_j) + \mathbf{q}) \\ & + \mu_j^T (\mathbf{F}(A_j x + B_j y + b_j) - \mathbf{f})]. \end{aligned} \tag{8.1}$$

Let $(\bar\lambda, \bar\mu)$ be the optimal solution of the problem

$$\begin{aligned} \max \quad & h(\lambda, \mu; \bar x) \\ s.t. \quad & \lambda_j^T e = \delta p_j, \qquad j = 1, \ldots, L, \\ & \lambda \geq 0,\ \mu \geq 0. \end{aligned}$$

Then $\rho(\bar x) = h(\bar\lambda, \bar\mu; \bar x)$.

The necessary and sufficient conditions for the optimal solution $\bar y$ of the problem (8.1) (given $(\bar\lambda, \bar\mu; \bar x)$) are

$$\nabla_y c(\bar x, \bar y) + \sum_{j=1}^{L} [\bar\lambda_j^T \mathbf{Q} B_j + \bar\mu_j^T \mathbf{F} B_j] \quad = \quad 0. \tag{8.2}$$

For fixed $(\lambda, \mu) = (\bar\lambda, \bar\mu)$ and for any x, denote by y_x the optimal solution of (8.1). Then

$$\begin{aligned} \rho(x) &= \max_{\lambda,\mu} h(\lambda, \mu; x), \\ &\geq h(\bar\lambda, \bar\mu; x), \\ &= c(x, y_x) + \sum_{j=1}^{L} [\bar\lambda_j^T (\mathbf{Q}(A_j x + B_j y_x + b_j) + \mathbf{q}) + \bar\mu_j^T (\mathbf{F}(A_j x + B_j y_x + b_j) - \mathbf{f})]. \end{aligned}$$

Because c is convex,

$$c(x,y) \geq c(\bar x, \bar y) + \nabla_x c(\bar x, \bar y)(x - \bar x) + \nabla_y c(\bar x, \bar y)(y - \bar y).$$

Note that $y_{\bar x} = \bar y$ and

$$\begin{aligned} \rho(\bar x) &= h(\bar\lambda, \bar\mu; \bar x), \\ &= c(\bar x, \bar y) + \sum_{j=1}^{L} [\bar\lambda_j^T (\mathbf{Q}(A_j \bar x + B_j \bar y + b_j) + \mathbf{q}) + \bar\mu_j^T (\mathbf{F}(A_j \bar x + B_j \bar y + b_j) - \mathbf{f})]. \end{aligned}$$

Thus,

$$\begin{aligned} \rho(x) &\geq \rho(\bar x) + \nabla_x c(\bar x, \bar y)(x - \bar x) + \nabla_y c(\bar x, \bar y)(y_x - \bar y), \\ &\quad + \sum_{j=1}^{L} [\bar\lambda_j^T (\mathbf{Q} A_j (x - \bar x) + \mathbf{Q} B_j (y_x - \bar y)) + \bar\mu_j^T (\mathbf{F} A_j (x - \bar x) + \mathbf{F} B_j (y_x - \bar y))], \\ &= \rho(\bar x) + \{\nabla_x c(\bar x, \bar y) + \sum_{j=1}^{L} (\bar\lambda_j^T \mathbf{Q} A_j + \bar\mu_j^T \mathbf{F} A_j)\}(x - \bar x). \end{aligned}$$

where the last equation uses (8.2). The above inequality shows that

$$\xi = \nabla_x c(\bar{x}, \bar{y}) + \sum_{j=1}^{L} (\bar{\lambda}_j^T \mathbf{Q} A_j + \bar{\mu}_j^T \mathbf{F} A_j)$$

is a subgradient of $M(V)$ at $\bar{x}$.

References

[1] H. Benitez-Silva, G. Hall, G.J. Hitsch, G. Pauletto, and J. Rust, A comparison of discrete and parametrix approximation methods for continuous-state dynamix programming problems, Reseach report, Department of Economics, Yale University, New Haven, CT, 2000.

[2] D.P. Bertsekas,*Dynamic Programming and Optimal Control, Volume II*, Athena Scientific, Belmont, MA, 1995.

[3] S.D. Flåm and R.J-B Wets, Existence results and finite horizon approximates for infinite horizon optimization problems, *Econometrica*, 55 (1987), pp. 1187–1209.

[4] R. Horst, P.M. Pardalos, and N.V. Thoai, *Introduction to Global Optimization*, 2nd edition, Kluwer Academic Publishers, Dordrecht, 2000.

[5] R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches*, 2nd revised edition. Springer Verlag, Heidelberg, 1993.

[6] R.C. Grinold, Finite horizon approximations of infinite horizon linear programs, *Math. Program.*, 12 (1977), pp. 1–17.

[7] L.A. Korf, An approximation framework for infinite horizon stochastic dynamic optimization problems with discounted costs, Research report, Department of Mathematics, Washington University, Seattle, WA, 2000.

[8] W.S. Lovejoy, A survey of algorithmic methods for partially observed Markov decision processes," *Ann. Oper. Res.*, 28 (1991), pp. 47–66.

[9] W.S. Lovejoy, Computationally feasible bound for partially observed Markov decision processes, *Oper. Res.*, 39 (1991), pp. 162–175.

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

[11] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[12] R.T. Rockafellar and R.J-B Wets, *Variational Analysis*, Springer-Verlag, Berlin, 1998.

[13] P.A. Samuelson, Lifetime portfolio selection by dynamic stochastic programming, *The Review of Economics and Statistics*, 51 (1969), pp. 239–246.

[14] R.D. Smallwood and E.J. Sondik, The optimal control of partially observable Markov processes over a finite horizon, *Oper. Res.*, 21 (1973), pp. 1071–1088.