INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
Int. J. Robust. Nonlinear Control 0000; **00**:1–25
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/rnc

Approximate Dynamic Programming via Iterated Bellman Inequalities

Yang Wang*, Brendan O'Donoghue, Stephen Boyd[1]

[1] *Packard Electrical Engineering, 350 Serra Mall, Stanford, CA, 94305*

SUMMARY

In this paper we introduce new methods for finding functions that lower bound the value function of a stochastic control problem, using an iterated form of the Bellman inequality. Our method is based on solving linear or semidefinite programs, and produces both a bound on the optimal objective, as well as a suboptimal policy that appears to work very well. These results extend and improve bounds obtained in a previous paper using a single Bellman inequality condition. We describe the methods in a general setting, and show how they can be applied in specific cases including the finite state case, constrained linear quadratic control, switched affine control, and multi-period portfolio investment. Copyright © 0000 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Convex Optimization; Dynamic Programming; Stochastic Control

1. INTRODUCTION

In this paper we consider stochastic control problems with arbitrary dynamics, objective, and constraints. In some special cases, these problems can be solved analytically. One famous example is when the dynamics are linear, and the objective function is quadratic (with no constraints), in which case the optimal control is linear state feedback [1, 2, 3]. Another example where the optimal policy can be computed exactly is when the state and action spaces are finite, in which case methods such as value iteration or policy iteration can be used [2, 3]. When the state and action spaces are infinite, but low dimensional, the optimal control problem can be solved by gridding or other discretization methods.

In general however, the optimal control policy cannot be tractably computed. In such situations, there are many methods for finding good suboptimal controllers that can often achieve a small objective value. One particular method we will discuss in detail is approximate dynamic programming (ADP) [2, 3, 4, 5], which relies on an expression for the optimal policy in terms of the value function for the problem. In ADP, the true value function is replaced with an approximation. These control policies often achieve surprisingly good performance, even when the approximation of the value function is not particularly good. For problems with linear dynamics and convex objective and constraints, we can evaluate such policies in tens of microseconds, which makes them entirely practical for fast real-time applications [6, 7, 8].

In this paper, we present a method for finding an approximate value function that globally underestimates (and approximates) the true value function. This yields both a numerical lower bound on the optimal objective value, as well as an ADP policy based on our underestimator.

*Correspondence to: Yang Wang. Email: yang1024@gmail.com

Our underestimator/bound is non-generic, in the sense that it does not simply depend on problem dimensions and some basic assumptions about the problem data. Instead, they are computed (numerically) for each specific problem instance. We will see that for many different problem families, our method is based on solving a convex optimization problem, thus avoiding the 'curses of dimensionality' usually associated with dynamic programming [5].

The bound we compute can be compared to the objective achieved by any suboptimal policy, which can be found via Monte-Carlo simulation. If the gap between the two is small, we can conclude that the suboptimal policy is nearly optimal, and our bound is nearly tight. If the gap is large, then one or both of the bound and the policy is poor. Under certain assumptions, we can also provide generic guarantees on the tightness of our bound. Our results extend and improve similar guarantees found in [9].

In previous works, the authors have considered bounds and underestimators based on the Bellman inequality [10, 11]. In this paper we present a more general condition based on an iterated form of the Bellman inequality, which significantly improves our results. Indeed, in numerical examples we find that the bound we compute is often extremely close to the objective achieved by the ADP policy.

1.1. Prior and related work

One work closely related to ours is by De Farias and Van Roy [9], who consider a similar stochastic control problem with a finite number of states and inputs. In their paper, the authors obtain a value function underestimator by relaxing the Bellman equation to an inequality. This results in a set of linear constraints, so the underestimators can be found by solving a linear programming problem (LP). The authors show that as long as the basis functions are 'well chosen', the underestimator will be a good approximation. (We will use similar methods to derive tightness guarantees for our iterated Bellman condition.) In [10, 11], Wang and Boyd extended these ideas to problems with an infinite number of states and inputs, obtaining a tractable sufficient condition for the Bellman inequality via the $\mathcal{S}$-procedure [12, §2.6.3]. Similar ideas and methods can also be found in papers by Savorgnan, Lasserre and Diehl [13], Bertsimas and Caramanis [14], and Lincoln and Rantzer [15, 16].

We should point out that this approach is popular and widely used in approximate dynamic programming. The original characterization of the true value function via linear programming is due to Manne [17]. The LP approach to ADP was introduced by Schweitzer and Seidmann [18] and De Farias and Van Roy [9]. There are many applications of this method, for example in optimal scheduling problems, revenue and portfolio management, inventory management, stochastic games, decentralized control and many others [19, 20, 21, 22, 23, 24, 25, 26, 27, 28].

While these methods typically work well, *i.e.*, the bound we get is often close to the objective achieved by the suboptimal policy, there are also situations in which the gap is large. This is partly due to the fact that the Bellman inequality is a sufficient, but not necessary condition for a lower bound on the value function. As a result the condition can often be overly conservative, as was pointed out in [9, 29]. In [29] Desai, Farias and Moallemi address this problem by adding slack variables to relax the Bellman inequality condition. This produces much better approximate value functions, but these may not be underestimators in general. In this paper we present a method that both relaxes the Bellman inequality, and also retains the lower bound property. We will see that this produces much better results compared with a single Bellman inequality condition.

There is a vast literature on computing lower bounds for stochastic control. In [30] Cogill and Lall derive a method for average cost-per-stage problems for finding both an upper bound on the cost incurred by a given policy, as well as a lower bound on the optimal objective value. One advantage of this method is that it does not require a restrictive lower bound condition, such as the Bellman inequality (but it still requires searching for good candidate functions, as in approximate dynamic programming). Using this method they analytically derive suboptimality gaps for queueing problems [30] as well as event-based sampling [31].

In [32], Brown, Smith and Sun take a different approach, where they relax the nonanticipativity constraint that decisions can only depend on information available at the current time. Instead, a

Prepared using rncauth.cls
DOI: 10.1002/rnc

penalty is imposed that punishes violations of the constraint. In one extreme case, the penalty is infinitely hard, which corresponds to the original stochastic control problem. The other extreme is full prescience, *i.e.*, there is no penalty on knowing the future, which clearly gives a lower bound on the original problem. Their framework comes with corresponding weak duality, strong duality, and complementary slackness results.

For specific problem families it is often possible to derive generic bounds that depend on some basic assumptions about the problem data. For example, Kumar and Kumar [19] derive bounds for queueing networks and scheduling policies. Bertsimas, Gamarnik and Tsitsiklis [33] consider a similar class of problems, but uses a different method based on piecewise linear Lyapunov functions. In a different application, Castañón [34] derives bounds for controlling a sensor network to minimize estimation error, subject to a resource constraint. To get a lower bound, the resource constraint is 'dualized' by adding the constraint into the objective weighted by a nonnegative Lagrange multiplier. The lower bound is then optimized over the dual variable. In fact, in certain special cases, the Bellman inequality approach can also be interpreted as a simple application of Lagrange duality [35].

Performance bounds have also been studied for more traditional control applications. For example, in [36], Peters, Salgado and Silva-Vera derive bounds for linear control with frequency domain constraints. Vuthandam, Genceli and Nikolau [37] derive bounds on robust model predictive control with terminal constraints.

Throughout this paper we assume that the set of basis functions used to parameterize the approximate value function has already been selected. We do not address the question of how to select such a set. This is large topic and an active area of research; we direct the interested reader to [38, 39, 40, 41, 42, 43, 44, 45] and the references therein. There are also many works that outline general methods for solving stochastic control problems and dealing with the 'curses of dimensionality' [5, 4, 46, 47, 48, 15]. Many of the ideas we will use appear in these and will be pointed out.

1.2. Outline

The structure of the paper is as follows. In §2 we define the stochastic control problem and give the dynamic programming characterization of the solution. In §3 we describe the main ideas behind our bounds in a general, abstract setting. In §4 we derive tightness guarantees for our bound. Then, in §5–§8 we outline how to compute these bounds for several problem families. For each problem family, we present numerical examples where we compute our bounds and compare them to the performance achieved by suboptimal control policies. Finally, in §9 we briefly outline several straightforward extensions/variations of our method.

2. STOCHASTIC CONTROL

We consider a discrete-time time-invariant dynamical system, with dynamics

$$x_{t+1} = f(x_t, u_t, w_t), \quad t = 0, 1, \ldots, \tag{1}$$

where $x_t \in \mathcal{X}$ is the state, $u_t \in \mathcal{U}$ is the input, $w_t \in \mathcal{W}$ is the process noise, all at time (or epoch) t, and $f : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$ is the dynamics function. We assume that $x_0, w_0, w_1, \ldots$, are independent random variables, with $w_0, w_1, \ldots$ identically distributed.

We consider causal state feedback control policies, where the input u_t is determined from the current and previous states $x_0, \ldots, x_t$. For the problem we consider it can be shown that there is a time-invariant optimal policy that depends only on the current state, *i.e.*,

$$u_t = \psi(x_t), \quad t = 0, 1, \ldots, \tag{2}$$

where $\psi : \mathcal{X} \to \mathcal{U}$ is the state feedback function or policy. With fixed state feedback function (2) and dynamics (1), the state and input trajectories are stochastic processes.

Prepared using rncauth.cls
DOI: 10.1002/rnc

We will consider an infinite horizon discounted objective with the form

$$J = \mathbf{E} \sum_{t=0}^{\infty} \gamma^t \ell(x_t, u_t), \tag{3}$$

where $\ell : \mathcal{X} \times \mathcal{U} \to \mathbf{R} \cup \{+\infty\}$ is the stage cost function, and $\gamma \in (0, 1)$ is a discount factor. We use infinite values of ℓ to encode constraints on the state and input. For example, unless

$$(x_t, u_t) \in \mathcal{C} = \{(x, u) \mid \ell(x, u) < \infty\} \text{ a.s.},$$

we have $J = \infty$. We assume that for each $z \in \mathcal{X}$, there is a $v \in \mathcal{U}$ with $\ell(z, v) < \infty$; in other words, for each state there is at least one feasible input. We assume that the expectations and sum in (3) exist; this is the case if ℓ is bounded below, for example, nonnegative.

The stochastic control problem is to find a state feedback function ψ that minimizes the objective J. We let $J^\star$ denote the optimal value of J (which we assume is finite), and $\psi^\star$ denote an optimal state feedback function. The problem data are the dynamics function f, the stage cost function ℓ, the discount factor γ, the distribution of the initial state x_0, and the distribution of the noise w_0 (which is the same as the distribution of w_t).

For more on the formulation of the stochastic control problem, including technical details, see, *e.g.*, [2, 3, 4, 49].

2.1. Dynamic programming

In this section we give the well known characterization of a solution of the stochastic control problem using dynamic programming. These results (and the notation) will be used later in the development of our performance bounds.

Let $V^\star : \mathcal{X} \to \mathbf{R}$ be the value function, *i.e.*, the optimal value of the objective, conditioned on starting from state $x_0 = z$:

$$V^\star(z) = \inf_{\psi} \mathbf{E} \left(\sum_{t=0}^{\infty} \gamma^t \ell(x_t, u_t) \right),$$

subject to the dynamics (1), with policy (2), and $x_0 = z$; the infimum here is over all policies ψ. We have $J^\star = \mathbf{E}\, V^\star(x_0)$ (with the expectation over x_0); we assume that $V^\star(x_0) < \infty$ a.s.

The function $V^\star$ is the unique solution of the Bellman equation,

$$V^\star(z) = \inf_{v \in \mathcal{U}} \{\ell(z, v) + \gamma\, \mathbf{E}\, V^\star(f(z, v, w_t))\} \quad \forall z \in \mathcal{X}, \tag{4}$$

which we can write in abstract form as

$$V^\star = \mathcal{T} V^\star,$$

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

$$(\mathcal{T}h)(z) = \inf_{v \in \mathcal{U}} \{\ell(z, v) + \gamma\, \mathbf{E}\, h(f(z, v, w_t))\} \tag{5}$$

for any $h : \mathcal{X} \to \mathbf{R}$. We can express an optimal policy in terms of $V^\star$ as

$$\psi^\star(z) = \operatorname*{argmin}_{v \in \mathcal{U}} \{\ell(z, v) + \gamma\, \mathbf{E}\, V^\star(f(z, v, w_t)\}. \tag{6}$$

Computing the optimal policy. The value function and associated optimal policy can be effectively computed in several special cases. When $\mathcal{X}$, $\mathcal{U}$, and $\mathcal{W}$ are finite, it can be solved by several methods, including value iteration (described below; [2, 3]). (This is practical when the product of the cardinality of these sets is not too large, say, under 10^8.) Another famous special case is when $\mathcal{X} = \mathbf{R}^n$, $\mathcal{U} = \mathbf{R}^m$, and ℓ is a convex quadratic function [1]. In this case, the value function is convex quadratic, and the optimal policy is affine, with coefficients that are readily computed from the problem data. In many cases, however, it is not practical to compute the value function $V^\star$, the optimal value of the stochastic control problem $J^\star$, or an optimal policy $\psi^\star$.

Prepared using rncauth.cls
DOI: 10.1002/rnc

2.2. Properties of the Bellman operator

The Bellman operator $\mathcal{T}$ has several interesting properties which we will use later in developing our bounds. Here, we state these properties without justification; for details and proofs, see *e.g.*, [2, 3, 4, 49, 47].

Monotonicity. For functions $f, g : \mathcal{X} \to \mathbf{R}$,

$$f \leq g \quad \Longrightarrow \quad \mathcal{T}f \leq \mathcal{T}g, \tag{7}$$

where the inequality between functions means elementwise, *i.e.*, $f(x) \leq g(x)$ for all $x \in \mathcal{X}$.

Value iteration convergence. For any function $f : \mathcal{X} \to \mathbf{R}$,

$$V^\star(x) = \lim_{k\to\infty} (\mathcal{T}^k f)(x), \tag{8}$$

for any $x \in \mathcal{X}$. In other words, iteratively applying the Bellman operator to any initial function results in pointwise convergence to the value function. (Much stronger statements can be made about the convergence; but we will only need pointwise convergence.) Computing $V^\star$ by iterative application of the Bellman operator is called *value iteration*.

2.3. Suboptimal policies and performance bounds

Many methods for finding a suboptimal or approximate policy have been proposed; we describe two of these below. For more details see [50, 51, 52, 53, 4, 5, 6].

Approximate dynamic programming policy. Following the basic idea in approximate dynamic programming, we define the *approximate dynamic programming policy* (or *ADP policy*) as

$$\psi^{\mathrm{adp}}(z) = \operatorname*{argmin}_{v\in\mathcal{U}} \{\ell(z, v) + \gamma\, \mathbf{E}\, V^{\mathrm{adp}}(f(z, v, w_t))\}, \tag{9}$$

where the function V^{adp} is called the approximate value function, or control-Lyapunov function. The ADP policy is the same as the optimal policy (6), with V^{adp} substituted for $V^\star$. This policy goes by several other names, including control-Lyapunov policy, and one-step receding-horizon control. With $V^{\mathrm{adp}} = V^\star$, the ADP policy is optimal. Far more interesting and important is the observation that the ADP policy often yields very good performance, even when V^{adp} is a not a particularly good approximation of $V^\star$.

Greedy policy. For many problems we will consider, the stage cost ℓ is state-input separable, *i.e.*, $\ell(z, v) = \ell_x(z) + \ell_u(v)$, where $\ell_x : \mathcal{X} \to \mathbf{R} \cup \{+\infty\}$ and $\ell_u : \mathcal{U} \to \mathbf{R} \cup \{+\infty\}$ are the stage costs for the states and inputs respectively. In this case, one of the simplest suboptimal policies is the one-step-ahead *greedy policy*, given by

$$\psi^{\mathrm{greedy}}(z) = \operatorname*{argmin}_{v\in\mathcal{U}} \{\ell_x(z) + \ell_u(v) + \gamma\, \mathbf{E}\, \ell_x(f(z, v, w_t))\}. \tag{10}$$

Comparing the greedy policy with the optimal policy (6), we see that the greedy policy simply minimizes the sum of the current stage cost and the expected stage cost at the next time period, ignoring the effect of the current action on the long-term future. (This is also just the ADP policy with $V^{\mathrm{adp}} = \ell_x$.) The greedy policy often performs very poorly; when $\mathcal{X}$ is infinite, we can even have $J = \infty$.

DOI: 10.1002/rnc

Performance bounds. For any policy we can evaluate the associated cost J (approximately) via Monte Carlo simulation, so we have an idea of how well the policy will perform. A question that arises immediately is, how suboptimal is the policy? In other words, how much larger is J than $J^\star$? To address this question we need a lower bound on $J^\star$, that is, a bound on the control performance that can be attained by any policy. One of the main purposes of this paper is to describe new tractable methods for obtaining performance bounds, in cases when computing the optimal value is not practical. As a side benefit of the lower bound computation, our method also provides approximate value functions for suboptimal ADP policies that appear to work very well, that is, yield cost J that is near the associated lower bound.

3. PERFORMANCE BOUNDS

In this section we work out the main ideas in the paper, in an abstract setting. In subsequent sections we address questions such as how the methods can be carried out for various specific cases.

3.1. Value function underestimators

All of our performance bounds will be based on the following observation: if the function $\hat{V} : \mathcal{X} \to \mathbf{R}$ satisfies

$$\hat{V} \leq V^\star, \tag{11}$$

then, by monotonicity of expectation,

$$\mathbf{E}\,\hat{V}(x_0) \leq \mathbf{E}\,V^\star(x_0) = J^\star. \tag{12}$$

Thus, we obtain the performance bound (*i.e.*, lower bound on $J^\star$) $\mathbf{E}\,\hat{V}(x_0)$. The challenge, of course, is to find an underestimator $\hat{V}$ of $V^\star$. Indeed, depending on the specific case, it can be difficult to verify that $\hat{V} \leq V^\star$ given a fixed $\hat{V}$, let alone find such a function.

3.2. Bound optimization

Our approach, which is the same as the basic approach in approximate dynamic programming (ADP), is to restrict our attention to a finite-dimensional subspace of candidate value function underestimators,

$$\hat{V} = \sum_{i=1}^{K} \alpha_i V^{(i)}, \tag{13}$$

where α_i are coefficients, and $V^{(i)}$ are basis functions for our candidate functions. We then optimize our lower bound over the coefficients, subject to a constraint that guarantees $\hat{V} \leq V^\star$:

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\,\hat{V}(x_0) = \alpha_1\,\mathbf{E}\,V^{(1)}(x_0) + \cdots + \alpha_K\,\mathbf{E}\,V^{(K)}(x_0) \\ \text{subject to} & \text{[condition that implies (11)]}, \end{array} \tag{14}$$

with variables $\alpha \in \mathbf{R}^K$. In the sequel we derive a condition that is convex in α and implies (11); this, along with the fact that the objective is linear, ensures that (14) is a convex optimization problem [54, 55]. By solving the problem (14), we obtain the best lower bound on $J^\star$ that can be obtained using the condition selected, and restricting $\hat{V}$ to the given subspace.

The associated optimal $\hat{V}$ for (14) can be interpreted (roughly) as an approximation of $V^\star$ (which always underestimates $V^\star$). Thus, $\hat{V}$ is a natural choice for V^{adp} in the ADP policy.

3.3. Bellman inequality

Let $\hat{V} : \mathcal{X} \to \mathbf{R}$ be a function that satisfies the *Bellman inequality* [2, 3],

$$\hat{V} \leq \mathcal{T}\hat{V}. \tag{15}$$

Prepared using rncauth.cls
DOI: 10.1002/rnc

By monotonicity of the Bellman operator, this implies $\hat{V} \leq \mathcal{T}\hat{V} \leq \mathcal{T}(\mathcal{T}\hat{V})$; iterating, we see that $\hat{V} \leq \mathcal{T}^k\hat{V}$ for any $k \geq 1$. Thus we get

$$\hat{V}(x) \leq \lim_{k\to\infty} (\mathcal{T}^k\hat{V})(x) = V^\star(x), \quad \forall x \in \mathcal{X}.$$

Thus, the Bellman inequality is a sufficient condition for $\hat{V} \leq V^\star$.

If we restrict $\hat{V}$ to a finite dimensional subspace, the Bellman inequality is a convex constraint on the coefficients, since it can be stated as

$$\hat{V}(z) \leq \inf_{v\in\mathcal{U}} \left\{ \ell(z,v) + \gamma\, \mathbf{E}\, \hat{V}(f(z,v,w_t)) \right\}, \quad \forall z \in \mathcal{X}.$$

For each $z \in \mathcal{X}$, the lefthand side is linear in α; the righthand side is a concave function of α, since it is the infimum over a family of affine functions [54, §3.2.3].

In the case of finite state and input spaces, using the Bellman inequality (15) as the condition in (14), we obtain a linear program. This was first introduced by De Farias and Van Roy [9], who showed that if the true value function is close to the subspace spanned by the basis functions, then $\hat{V}$ is guaranteed to be close to $V^\star$. In a different context, for problems with linear dynamics, quadratic costs and quadratic constraints (with infinite numbers of states and inputs), Wang and Boyd derived a sufficient condition for (15) that involves a linear matrix inequality (LMI) [10, 11]. The optimization problem (14) becomes a semidefinite program (SDP), which can be efficiently solved using convex optimization methods [54, 56, 55, 57].

3.4. Iterated Bellman inequality

Suppose that $\hat{V}$ satisfies the *iterated Bellman inequality*,

$$\hat{V} \leq \mathcal{T}^M\hat{V}, \tag{16}$$

where $M \geq 1$ is an integer. By the same argument as for the Bellman inequality, this implies $\hat{V} \leq \mathcal{T}^{kM}\hat{V}$ for any integer $k \geq 1$, which implies

$$\hat{V}(x) \leq \lim_{k\to\infty} (\mathcal{T}^{kM}\hat{V})(x) = V^\star(x), \quad \forall x \in \mathcal{X},$$

so the iterated Bellman inequality also implies $\hat{V} \leq V^\star$. If $\hat{V}$ satisfies the Bellman inequality (15), then it must satisfy the iterated Bellman inequality (16). The converse is not always true, so the iterated bound is a more general sufficient condition for $\hat{V} \leq V^\star$.

In general, the iterated Bellman inequality (16) is not a convex constraint on the coefficients $\hat{V}$, when we restrict $\hat{V}$ to a finite-dimensional subspace. However, we can derive a sufficient condition for (16) that is convex in the coefficients. The iterated Bellman inequality (16) is equivalent to the existence of functions $\hat{V}_1, \ldots, \hat{V}_{M-1}$ satisfying

$$\hat{V} \leq \mathcal{T}\hat{V}_1, \quad \hat{V}_1 \leq \mathcal{T}\hat{V}_2, \quad \ldots \quad \hat{V}_{M-1} \leq \mathcal{T}\hat{V}. \tag{17}$$

(Indeed, we can take $\hat{V}_{M-1} = \mathcal{T}\hat{V}$, and $\hat{V}_i = \mathcal{T}\hat{V}_{i+1}$ for $i = M-2, \ldots, 1$.) Defining $\hat{V}_0 = \hat{V}_M = \hat{V}$, we can write this more compactly as

$$\hat{V}_{i-1} \leq \mathcal{T}\hat{V}_i, \quad i = 1, \ldots, M. \tag{18}$$

Now suppose we restrict each $\hat{V}_i$ to a finite-dimensional subspace:

$$\hat{V}_i = \sum_{j=1}^{K} \alpha_{ij} V^{(j)}, \quad i = 0, \ldots, M-1.$$

(Here we use the same basis for each $\hat{V}_i$ for simplicity.) On this subspace, the iterated Bellman inequality (18) is a set of convex constraints on the coefficients α_{ij}. To see this, we note that for

DOI: 10.1002/rnc

each $x \in \mathcal{X}$, the lefthand side of each inequality is linear in the coefficients, while the righthand sides (*i.e.*, $\mathcal{T}\hat{V}_i$) are concave functions of the coefficients, since each is an infimum of affine functions.

Using (18) as the condition in the bound optimization problem (14), we get a convex optimization problem. For $M = 1$, this reduces to the finite-dimensional restriction of the single Bellman inequality. For $M > 1$, the performance bound obtained can only be better than (or equal to) the bound obtained for $M = 1$. To see this, we argue as follows. If $\hat{V}$ satisfies $\hat{V} \leq \mathcal{T}\hat{V}$, then $\hat{V}_i = \hat{V}$, $i = 0, \ldots, M$, must satisfy the finite-dimensional restriction of the iterated Bellman inequality (18). Thus, the condition (18) defines a larger set of underestimators compared with the single Bellman inequality. A similar argument shows that if M_2 divides M_1, then the bound we get with $M = M_1$ must be better than (or equal to) the bound with $M = M_2$.

The computational complexity of the convex optimization problem grows *linearly* with M. This is because each $\hat{V}_i$ appears in constraints only with the previous and the subsequent functions in the sequence, which yields a problem with a block-banded Hessian. This special structure can be exploited by most convex optimization algorithms, such as interior point methods [54, §9.7.2], [58].

3.5. Pointwise supremum underestimator

Suppose $\{\hat{V}_\alpha \mid \alpha \in \mathcal{A}\}$ is a family of functions parametrized by $\alpha \in \mathcal{A}$, all satisfying $\hat{V}_\alpha \leq V^\star$. For example, the set of underestimators obtained from the feasible coefficient vectors α from the Bellman inequality (15) or the iterated Bellman inequality (18) is such a family. Then the pointwise supremum is also an underestimator of V:

$$\bar{V}(z) = \sup_{\alpha \in \mathcal{A}} \hat{V}_\alpha(z) \leq V^\star(z), \quad \forall z \in \mathcal{X}.$$

It follows that $\mathbf{E}\,\bar{V}(x_0) \leq J^\star$. Moreover, this performance bound is as good as any of the individual performance bounds: for any $\alpha \in \mathcal{A}$,

$$\mathbf{E}\,\bar{V}(x_0) \geq \mathbf{E}\,\hat{V}_\alpha(x_0).$$

This means that we can switch the order of expectation and maximization in (14), to obtain a better bound: $\mathbf{E}\,\bar{V}(x_0)$, which is the expected value of the optimal value of the (random) problem

$$\begin{array}{ll} \text{maximize} & \hat{V}(x_0) = \alpha_1 V^{(1)}(x_0) + \cdots + \alpha_K V^{(K)}(x_0) \\ \text{subject to} & \text{[condition that implies (11)]}, \end{array} \tag{19}$$

over the distribution of x_0. This pointwise supremum bound is guaranteed to be a better lower bound on $J^\star$ than the basic bound obtained from problem (14).

This bound can be computed using a Monte Carlo procedure: We draw samples $z_1, z_2, \ldots, z_N$ from the distribution of x_0, solve the optimization problem (19) for each sample value, which gives us $\bar{V}(z_i)$. We then form the (Monte Carlo estimate) lower bound $(1/N)\sum_{i=1}^N \bar{V}(z_i)$. This evidently involves substantial, and in many cases prohibitive, computation.

3.6. Pointwise maximum underestimator

An alternative to the pointwise supremum underestimator is to choose a modest number of representative functions $\hat{V}_{\alpha_1}, \ldots, \hat{V}_{\alpha_L}$ from the family and form the function

$$\hat{V}(z) = \max_{i=1,\ldots,L} \hat{V}_{\alpha_i}(z),$$

which evidently is an underestimator of V. (We call this the pointwise maximum underestimator.) This requires solving L optimization problems to find $\alpha_1, \ldots, \alpha_L$. Now, Monte Carlo simulation, *i.e.*, evaluation of $\hat{V}(z_i)$, involves computing the maximum of L numbers; in particular, it involves no optimization. For this reason we can easily generate a large number of samples to evaluate $\mathbf{E}\,\hat{V}(x_0)$, which is a lower bound on $J^\star$. Another advantage of using $\hat{V}$ instead of $\bar{V}$ is that $\hat{V}$ can be used as an approximate value function in a approximate policy, as described in §2.3. The use of pointwise maximum approximate value functions has also been explored in a slightly different context in [48].

DOI: 10.1002/rnc

One generic method for finding good representative functions is to find extremal points in our family of underestimators. To do this, we let y be a random variable that takes values in $\mathcal{X}$ with some distribution. Then we solve the (convex) optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\,\hat{V}_\alpha(y) \\ \text{subject to} & \alpha \in \mathcal{A}, \end{array}$$

with variable α. When $\mathcal{X}$ is finite the distribution of y can be interpreted as state-relevance weights: If the state relevance weights for a particular subset of $\mathcal{X}$ are large, then our goal in the problem above is to make $\tilde{V}_\alpha(z)$ as large as possible for z in this subset, and hence as close as possible to $V^\star$. To get a different extremal point in the family, we pick a different distribution for y, where the probability density is concentrated around a different subset of $\mathcal{X}$ (see, *e.g.*, [59]).

4. TIGHTNESS

4.1. Notation and Assumptions

In this section we use similar methods as [9] to derive a simple tightness guarantee for our iterated Bellman bound. For simplicity, we will assume that all our functions are continuous, *i.e.*, $f \in C(\mathcal{X} \times \mathcal{U} \times \mathcal{W})$, $\ell \in C(\mathcal{X} \times \mathcal{U})$, the spaces $\mathcal{X}$ and $\mathcal{U}$ are compact, and x_0 has finite mean and covariance. This implies that the optimal value function $V^\star$ is continuous on $\mathcal{X}$, and the Bellman operator $\mathcal{T}$ is a sup-norm γ-contraction:

$$\|\mathcal{T}h_1 - \mathcal{T}h_2\|_\infty \leq \gamma \|h_1 - h_2\|_\infty,$$

where $h_1,\ h_2 \in C(\mathcal{X})$, $\|h_1 - h_2\|_\infty = \sup_{x\in\mathcal{X}} |h_1(x) - h_2(x)|$, and γ is the discount factor of the problem. As before, we assume that $\hat{V}$ has the representation (13), where each $V^{(i)} \in C(\mathcal{X})$. We let

$$\mathcal{H} = \left\{ \hat{V} \;\middle|\; \hat{V} = \sum_{i=1}^{K} \alpha_i V^{(i)},\ \alpha \in \mathbf{R}^K \right\}$$

denote the subspace spanned by the basis functions. In addition, we denote by $\mathbf{1} \in C(\mathcal{X})$ the constant function that assigns the value $1 \in \mathbf{R}$ to every $x \in \mathcal{X}$.

4.2. Main result

We will derive the following result: If $\mathbf{1} \in \mathcal{H}$, then

$$\mathbf{E}\,|V^\star(x_0) - \hat{V}^\star(x_0)| \leq \frac{2}{1-\gamma^M} \|V^\star - V^p\|_\infty, \tag{20}$$

where $\hat{V}^\star$ denotes the solution to the bound optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\,\hat{V}(x_0) \\ \text{subject to} & \hat{V} \leq \mathcal{T}^M \hat{V}, \end{array} \tag{21}$$

with variable $\alpha \in \mathbf{R}^K$, and V^p is an $\mathcal{L}^\infty$ projection of $V^\star$ onto the subspace $\mathcal{H}$, *i.e.*, it minimizes $\|V^\star - \hat{V}\|_\infty$ over $\mathcal{H}$.

This result can be interpreted as follows: If $V^\star$ is close to the subspace spanned by the basis functions, (*i.e.*, $\|V^\star - V^p\|_\infty$ is small), our underestimator will be close to the true value function. For the single Bellman inequality condition ($M = 1$), our result is the same as the one in [9]. In this case, the constant factor is equal to $2/(1-\gamma)$, which is large if γ is close to one. In the other extreme, as $M \to \infty$ the constant factor converges to 2, so we get much tighter suboptimality guarantees with the iterated Bellman inequality. In practice, however, we will see that even a factor of 2 is overly conservative—the suboptimality gaps we observe in practice are typically much smaller.

DOI: 10.1002/rnc

4.3. Proof

In order to prove this result, we need to be able to relate $\hat{V}^\star$ to the projection V^p. First we notice that

$$\|V^\star - \mathcal{T}^M V^p\|_\infty = \|\mathcal{T}^M V^\star - \mathcal{T}^M V^p\|_\infty \leq \gamma^M \|V^\star - V^p\|_\infty,$$

where the inequality follows from the fact that $\mathcal{T}$ is a γ-contraction. This implies

$$-\gamma^M \|V^\star - V^p\|_\infty \leq V^\star - \mathcal{T}^M V^p \leq \gamma^M \|V^\star - V^p\|_\infty,$$

so we get

$$\begin{aligned} \mathcal{T}^M V^p &\geq V^\star - \gamma^M \|V^\star - V^p\|_\infty \\ &\geq V^p - \|V^\star - V^p\|_\infty - \gamma^M \|V^\star - V^p\|_\infty \\ &= V^p - (1+\gamma^M)\|V^\star - V^p\|_\infty. \end{aligned}$$

(Here the notation $h + \alpha$, where h is a function and α is a scalar, means $h + \alpha\mathbf{1}$.) The second inequality follows because $V^p - V^\star \leq \|V^\star - V^p\|_\infty$. Now we will see that if we shift V^p downwards by a constant amount, it will satisfy the iterated Bellman inequality. Let

$$\tilde{V} = V^p - \frac{1+\gamma^M}{1-\gamma^M}\|V^\star - V^p\|_\infty.$$

We know $\tilde{V} \in \mathcal{H}$, since $V^p \in \mathcal{H}$ (by definition) and $\mathbf{1} \in \mathcal{H}$ (by assumption). Thus we can write

$$\begin{aligned} \mathcal{T}^M \tilde{V} &\geq \mathcal{T}^M V^p - \gamma^M \frac{1+\gamma^M}{1-\gamma^M}\|V^\star - V^p\|_\infty \\ &\geq V^p - (1+\gamma^M)\|V^\star - V^p\|_\infty - \gamma^M \frac{1+\gamma^M}{1-\gamma^M}\|V^\star - V^p\|_\infty \\ &= V^p - \frac{1+\gamma^M}{1-\gamma^M}\|V^\star - V^p\|_\infty = \tilde{V}, \end{aligned}$$

so $\tilde{V}$ satisfies the iterated Bellman inequality. This means that $\tilde{V}$ must be feasible for the problem

$$\begin{array}{ll} \text{minimize} & \mathbf{E}(V^\star(x_0) - \hat{V}(x_0)) \\ \text{subject to} & \hat{V} \leq \mathcal{T}^M \hat{V}. \end{array} \tag{22}$$

Since $\hat{V}^\star$ solves (21) it must also solve (22), which implies

$$\begin{aligned} \mathbf{E}\,|V^\star(x_0) - \hat{V}^\star(x_0)| &\leq \mathbf{E}\,|V^\star(x_0) - \tilde{V}(x_0)| \\ &\leq \|V^\star - \tilde{V}\|_\infty \\ &\leq \|V^\star - V^p\|_\infty + \|V^p - \tilde{V}\|_\infty \\ &= \frac{2}{1-\gamma^M}\|V^\star - V^p\|_\infty. \end{aligned}$$

This proves our result.

5. FINITE STATE AND INPUT SPACES

In this section we describe how to compute our bounds when the number of states, inputs and disturbances is finite. We take

$$\mathcal{X} = \{1, \ldots, N_x\}, \quad \mathcal{U} = \{1, \ldots, N_u\}, \quad \mathcal{W} = \{1, \ldots, N_w\}.$$

We define $p_i = \mathbf{Prob}\{w_t = i\}$ for $i = 1, \ldots, N_w$.

DOI: 10.1002/rnc

5.1. Value iteration

In principle, since the number of states and inputs is finite, we can carry out value iteration explicitly. We will consider here a naive implementation that does not exploit any sparsity or other structure in the problem. Given a function $V : \mathcal{X} \to \mathbf{R}$, we evaluate $V^+ = \mathcal{T}V$ as follows. For each (z, v), we evaluate

$$\ell(z, v) + \mathbf{E}\, V(f(x, z, w_t) = \ell(z, v) + \sum_{i=1}^{N_w} p_i V(f(x, z, i)),$$

which requires around $N_x N_u N_w$ arithmetic operations. We can then take the minimum over v for each z to obtain $V^+(z)$. So one step of value iteration costs around $N_x N_u N_w$ arithmetic operations. When $N_x N_u N_w$ is not too large, say more than 10^8 or so, it is entirely practical to compute the value function using value iteration. In such cases, of course, there is no need to compute a lower bound on performance. Thus, we are mainly interested in problems with $N_x N_u N_w$ larger than, say, 10^8, or where exact calculation of the value function is not practical. In these cases we hope that a reasonable performance bound can be found using a modest number of basis functions.

5.2. Iterated Bellman inequality

The iterated Bellman inequality (18), with K basis functions for $\hat{V}_i$, leads to the linear inequalities

$$\hat{V}_{i-1}(z) \leq \ell(z, v) + \gamma \sum_{j=1}^{N_w} p_j \hat{V}_i(f(z, v, w_j)), \quad i = 1, \ldots, M, \tag{23}$$

for all $z \in \mathcal{X}$, $v \in \mathcal{U}$. For each (z, v), (23) is a set of M linear inequalities in the MK variables α_{ij}. Thus, the iterated Bellman inequality (18) involves MK variables and MN_xN_u inequalities. Each inequality involves $2K$ variables.

Even when M is small and K is modest (say, a few tens), the number of constraints can be very large. Computing the performance bound (14), or an extremal underestimator for the iterated bound then requires the solution of an LP with a modest number of variables and a very large number of constraints. This can be done, for example, via constraint sampling [60], or using semi-infinite programming methods (see, *e.g.*, [61]).

6. CONSTRAINED LINEAR QUADRATIC CONTROL

In this and the following sections, we will restrict our candidate functions to the subspace of quadratic functions. We will use several key properties of quadratic functions which are presented in the appendix, in particular a technique known as the $\mathcal{S}$-procedure.

We consider here systems with $\mathcal{X} = \mathbf{R}^n$, $\mathcal{U} = \mathbf{R}^m$, and $\mathcal{W} = \mathbf{R}^n$, with linear dynamics

$$x_{t+1} = f(x_t, u_t, w_t) = Ax_t + Bu_t + w_t, \quad t = 0, 1, \ldots.$$

We will assume that $\mathbf{E}\, w_t = 0$, and let W denote the disturbance covariance, $W = \mathbf{E}\, w_t w_t^T$. The stage cost is a convex state-input separable quadratic, restricted to a unit box input constraint set,

$$\ell(z, v) = \begin{cases} z^T Q z + v^T R v & \|v\|_\infty \leq 1 \\ +\infty & \|v\|_\infty > 1, \end{cases}$$

where $Q \in \mathbf{S}^n_+$, $R \in \mathbf{S}^m_+$. ($\mathbf{S}^n_+$ is the set of $n \times n$ symmetric positive semidefinite matrices.)

The same approach described here can be applied to the more general case with nonzero disturbance mean, linear terms and state-input coupling terms in the stage cost, and constraint sets described by a set of quadratic equalities and inequalities. The formulas for the more general case are readily derived, but much more complex than for the special case considered here.

Prepared using rncauth.cls
DOI: 10.1002/rnc

6.1. Iterated Bellman inequality

We will use quadratic candidate functions

$$\hat{V}_i(z) = z^T P_i z + s_i = \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}, \quad i = 0, \ldots, M,$$

where $P_i \in \mathbf{S}^n$, $s_i \in \mathbf{R}$, $i = 0, \ldots, M$, with $\hat{V} = \hat{V}_0 = \hat{V}_M$. (Due to our assumptions, we do not need linear terms in $\hat{V}_i$.)

The iterated Bellman inequality (18) can be written as

$$\hat{V}_{i-1}(z) \leq \ell(z, v) + \gamma \, \mathbf{E} \, \hat{V}_i(Az + Bv + w_t), \quad \forall \|v\|_\infty \leq 1, \quad i = 1, \ldots, M. \tag{24}$$

Expanding $\mathbf{E} \, \hat{V}_i(Az + Bv + w_t)$ we get

$$\begin{aligned} \mathbf{E} \, \hat{V}_i(Az + Bv + w_t) &= (Az + Bv)^T P_i (Az + Bv) + 2(Az + Bv)^T P_i \, \mathbf{E} \, w_t + \mathbf{E} \, w_t^T P_i w_t + s_i \\ &= (Az + Bv)^T P_i (Az + Bv) + \mathbf{Tr}(P_i W) + s_i \\ &= \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} B^T P_i B & B^T P_i A & 0 \\ A^T P_i B & A^T P_i A & 0 \\ 0 & 0 & \mathbf{Tr}(P_i W) + s_i \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}. \end{aligned}$$

Thus (24) becomes:

$$\begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} R + \gamma B^T P_i B & \gamma B^T P_i A & 0 \\ \gamma A^T P_i B & Q + \gamma A^T P_i A - P_{i-1} & 0 \\ 0 & 0 & \gamma(\mathbf{Tr}(P_i W) + s_i) - s_{i-1} \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix} \geq 0, \tag{25}$$

for all $\|v\|_\infty \leq 1$, $i = 1, \ldots, M$. We can express $\|v\|_\infty \leq 1$ as a set of quadratic inequalities,

$$1 - v_i^2 = \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} -e_i e_i^T & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix} \geq 0, \quad i = 1, \ldots, m. \tag{26}$$

An arbitrary nonnegative linear combination of these quadratic functions can be expressed as

$$\begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} -D & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \mathbf{Tr} \, D \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}, \tag{27}$$

where $D \in \mathbf{S}_+^m$ is diagonal.

Now we use the $\mathcal{S}$-procedure to derive a sufficient condition: there exists diagonal nonnegative $D^{(i)} \in \mathbf{R}^N$, $i = 1, \ldots, M$, such that

$$\begin{bmatrix} R + \gamma B^T P_i B + D^{(i)} & \gamma B^T P_i A & 0 \\ \gamma A^T P_i B & Q + \gamma A^T P_i A - P_{i-1} & 0 \\ 0 & 0 & \gamma(\mathbf{Tr}(P_i W) + s_i) - s_{i-1} - \mathbf{Tr} \, D^{(i)} \end{bmatrix} \succeq 0. \tag{28}$$

for $i = 1, \ldots, M$. The condition (28) is an LMI in the variables P_i, s_i, and $D^{(i)}$, so the bound optimization problem is convex (and tractable); in fact, an SDP.

These 3×3 block LMIs can be split into 2×2 block LMIs,

$$\begin{bmatrix} R + \gamma B^T P_i B + D^{(i)} & \gamma B^T P_i A \\ \gamma A^T P_i B & Q + \gamma A^T P_i A - P_{i-1} \end{bmatrix} \succeq 0, \quad i = 1, \ldots, M, \tag{29}$$

Prepared using rncauth.cls
DOI: 10.1002/rnc

Performance Bound	**Value**
Optimal value, $J^\star$	37.8
Pointwise maximum bound	37.5
Iterated bound ($M = 200$)	28.2
Basic Bellman bound ($M = 1$)	16.1
Unconstrained bound	15.5

Table I: Comparison of $J^\star$ with various bounds for the one dimensional example

and linear scalar inequalities,

$$\gamma(\mathbf{Tr}(P_iW) + s_i) - s_{i-1} - \mathbf{Tr}\, D^{(i)} \geq 0, \quad i = 1, \ldots, M. \tag{30}$$

Using this sufficient condition for the iterated Bellman inequalities, the bound optimization problem (14) becomes the SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{E}\,\hat{V}_0(x_0) = \mathbf{Tr}\, P_0(\mathbf{E}\, x_0 x_0^T) + s_0 \\ \text{subject to} & (29),\ (30), \\ & D^{(i)} \succeq 0, \quad i = 1, \ldots, M, \end{array} \tag{31}$$

with the variables listed above. A monotonicity argument tells us that we will have

$$s_{i-1} = \gamma(\mathbf{Tr}(P_iW) + s_i) - \mathbf{Tr}\, D^{(i)}, \quad i = 1, \ldots, M,$$

at the optimum of (31).

Removing the variable D from (31) is equivalent to removing the constraint on the input. In that case the true value function is convex quadratic and the performance bound is tight for any $M \geq 1$. The solution to this modified problem is the unconstrainted linear-quadratic regulator (LQR) solution for an infinite horizon discrete-time system and provides another lower bound for comparison [1, 2, 3].

6.2. One dimensional example

In this section we illustrate our underestimators and bounds on an example problem with one state ($n = 1$) and one input ($m = 1$). The problem data are:

$$A = 1, \quad B = -0.5, \quad Q = 1, \quad R = 0.1, \quad \gamma = 0.95,$$

and we assume $w_t \sim \mathcal{N}(0, 0.1)$ and $x_0 \sim \mathcal{N}(0, 10)$. Since the problem dimensions are small, we can compute the exact value function $V^\star$ by discretizing the state and input, and using traditional dynamic programming methods, such as value iteration (see §5.1) or policy iteration. We can also compute $J^\star = \mathbf{E}\, V^\star(x_0)$, via Monte-Carlo simulation.

We compare various bounds in Table I. The unconstrained bound refers to the optimal cost of the same problem without the input constraint. We can see that the iterated Bellman bound is a much better bound compared with the basic Bellman bound and unconstrained bounds, which give similar values for this particular problem instance. The pointwise maximum bound (with 10 representative functions) significantly improves on the iterated bound, and is very close to $J^\star$.

Figure 1 shows a comparison of the underestimators. The left figure compares $V^\star$ (black) with the value function of the unconstrained problem $V^\star_{\mathrm{lq}}$ (green), the basic Bellman underestimator $\hat{V}_{\mathrm{be}}$ (blue), and the iterated Bellman underestimator $\hat{V}_{\mathrm{it}}$ (red). We see that the iterated underestimator is a much better overall underestimator, but deviates from $V^\star$ for small z. The right figure compares $V^\star$ (black) with V_{pwq} (red), which is the pointwise maximum underestimator with 10 representative functions. It is clear that the two are almost indistinguishable.

Prepared using rncauth.cls
DOI: 10.1002/rnc

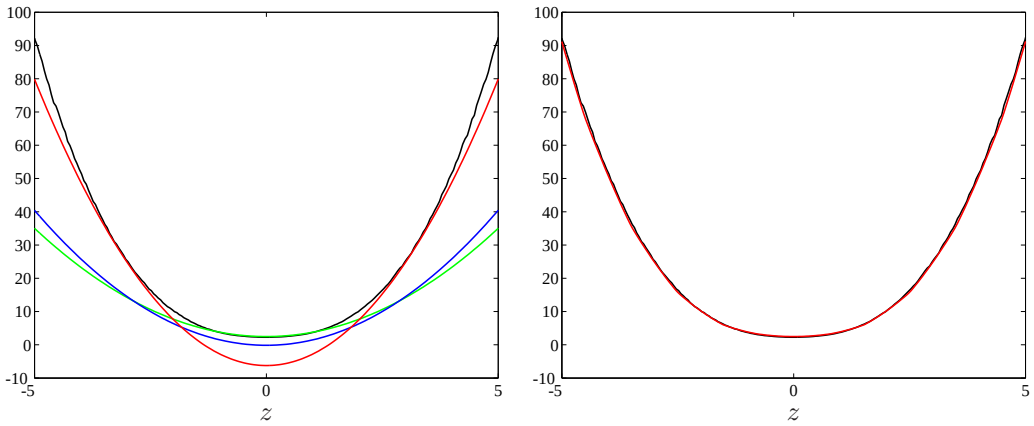

Figure 1: Left: Comparison of $V^\star$ (black) with $V_{\mathrm{lq}}^\star$ (green), $\hat{V}_{\mathrm{be}}$ (blue) and $\hat{V}_{\mathrm{it}}$ (red). Right: Comparison of $V^\star$ (black) with $\hat{V}_{\mathrm{pwq}}$ (red).

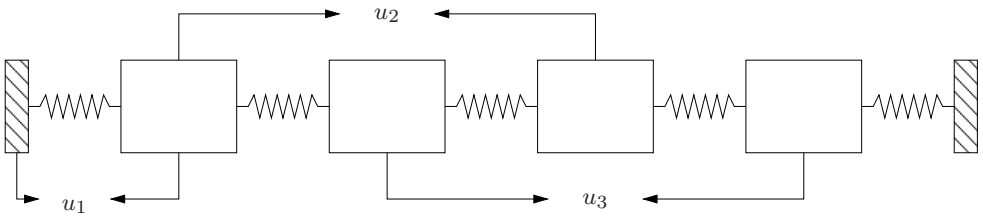

Figure 2: Mechanical control example.

6.3. Mechanical control example

Now we evaluate our bounds against the performance of various suboptimal policies for a discretized mechanical control system, consisting of 4 masses, connected by springs, with 3 input forces that can be applied between pairs of masses. This is shown in figure 2. For this problem, there are $n = 8$ states and $m = 3$ controls. The first four states are the positions of the masses, and the last four are their velocities. The stage costs are quadratic with $R = 0.01I$, $Q = 0.1I$ and $\gamma = 0.95$. The process noise w_t has distribution $\mathcal{N}(0, W)$, where $W = 0.1\,\mathbf{diag}(0, 0, 0, 0, 1, 1, 1, 1)$ (*i.e.*, the disturbances are random forces). The initial state x_0 has distribution $\mathcal{N}(0, 10I)$.

The results are shown in table II. The pointwise supremum bound is computed via Monte Carlo simulation, using an iterated Bellman inequality condition with $M = 100$. The unconstrained bound refers to the optimal objective of the problem without the input constraint (which we can compute analytically). We can clearly see that the gap between the ADP policy and the pointwise supremum bound is very small, which shows both are nearly optimal. This confirms our empirical observation from the one dimensional case that the pointwise maximum underestimator is almost indistinguishable from the true value function. We also observe that the greedy policy, which uses a naive approximate value function, performs much worse compared with our ADP policy, obtained from our bound optimization procedure.

7. AFFINE SWITCHING CONTROL

Here we take $\mathcal{X} = \mathbf{R}^n$, $\mathcal{W} = \mathbf{R}^n$, and $\mathcal{U} = \{1, \ldots, N\}$. The dynamics is affine in x_t and w_t, for each choice of u_t:

$$x_{t+1} = f(x_t, u_t, w_t) = A_{u_t} x_t + b_{u_t} + w_t, \quad t = 0, 1, \ldots,$$

DOI: 10.1002/rnc

Policy	**Objective**
Greedy	106.6
ADP, $V^{\mathrm{adp}} = \hat{V}$ (from iterated bound)	87.9
Performance bound	**Value**
Pointwise supremum bound	84.9
Iterated bound ($M = 100$)	69.4
Basic Bellman bound ($M = 1$)	51.6
Unconstrained bound	26.3

Table II: Performance of suboptimal policies (top half) and performance bounds (bottom half) for mechanical control example.

where $A_j \in \mathbf{R}^{n\times n}$ and $b_j \in \mathbf{R}^n$, $j = 1, \ldots, N$ give the dynamics matrices for inputs $u_t = 1, \ldots, N$, respectively. Roughly speaking, the control input u_t allows us to switch between a finite set of affine dynamical systems. We assume $\mathbf{E}\, w_t = 0$, and define $W = \mathbf{E}\, w_t w_t^T$.

The stage cost ℓ has the form

$$\ell(z, v) = z^T Q z + 2q^T z + l_v,$$

where $Q \in \mathbf{S}^n_+$, $q \in \mathbf{R}^n$, and $l \in \mathbf{R}^m$. If $u_t = j$, the input cost l_{u_t} can be interpreted as the cost of choosing system j, at time t.

In this formulation we consider only systems whose dynamics switch depending on the input. We can also derive similar lower bound conditions for more general cases with state-dependent switching, state constraints, as well as input-state coupling costs. Switching systems arise frequently in practical control problems; one example is the control of switch mode power converters, such as buck/boost converters [62, 63].

7.1. Iterated Bellman inequality

We use quadratic candidate functions $\hat{V}_0, \ldots, \hat{V}_M$:

$$\hat{V}_i(z) = z^T P_i z + 2p_i^T z + s_i, \quad i = 0, \ldots, M,$$

where $P_i \in \mathbf{S}^n$, $p_i \in \mathbf{R}^n$, $s_i \in \mathbf{R}$, $i = 0, \ldots, M$, with $\hat{V} = \hat{V}_0 = \hat{V}_M$. We can write the iterated Bellman inequality (18) as

$$\hat{V}_{i-1}(z) \leq \ell(z, j) + \gamma\, \mathbf{E}\, \hat{V}_i(A_j z + b_j + w_t), \quad \forall z \in \mathbf{R}^n, \quad i = 1, \ldots, M, \quad j = 1, \ldots, N. \quad (32)$$

The expectation can be evaluated using

$$\begin{aligned} \mathbf{E}\, \hat{V}_i(y + w_t) &= y^T P_i y + 2y^T P_i\, \mathbf{E}\, w_t + \mathbf{E}\, w_t^T P_i w_t + 2p_i^T (y + \mathbf{E}\, w_t) + s_i \\ &= y^T P_i y + 2p_i^T y + s_i + \mathbf{Tr}(P_i W). \end{aligned}$$

Using this to expand $\mathbf{E}\, \hat{V}_i(A_j z + b_j + w_t)$ we get

$$\begin{aligned} \mathbf{E}\, \hat{V}_i(A_j z + b_j + w_t) &= (A_j z + b_j)^T P_i (A_j z + b_j) + 2p_i^T (A_j z + b_j) + s_i + \mathbf{Tr}(P_i W) \\ &= \begin{bmatrix} z \\ 1 \end{bmatrix}^T \begin{bmatrix} H^{(ij)} & g^{(ij)} \\ g^{(ij)T} & c^{(ij)} \end{bmatrix} \begin{bmatrix} z \\ 1 \end{bmatrix}, \end{aligned}$$

where

$$H^{(ij)} = A_j^T P_i A_j, \quad g^{(ij)} = A_j^T P_i b_j + A_j^T p_i, \quad c^{(ij)} = b_j^T P_i b_j + 2b_j^T p_i + s_i + \mathbf{Tr}(P_i W).$$

DOI: 10.1002/rnc

Policy	**Objective**
Greedy	120.9
ADP, $V^{\mathrm{adp}} = \hat{V}$ (from iterated bound)	107.7
Performance bound	**Value**
Pointwise supremum bound	100.1
Iterated bound ($M = 50$)	89.9
Basic Bellman bound ($M = 1$)	72.8

Table III: Performance of suboptimal policies (top half) and performance bounds (bottom half) for affine switching control example.

Thus we can write (32) as

$$\left[\begin{array}{c} z \\ 1 \end{array}\right]^T \left[\begin{array}{cc} Q+\gamma H^{(ij)}-P_{i-1} & q+\gamma g^{(ij)}-p_{i-1} \\ q^T+\gamma g^{(ij)T}-p_{i-1}^T & l_j+\gamma c^{(ij)}-s_{i-1} \end{array}\right] \left[\begin{array}{c} z \\ 1 \end{array}\right] \geq 0, \quad \forall z \in \mathbf{R}^n,$$

and for $i = 1, \ldots, M$, $j = 1, \ldots, N$. This is equivalent to the LMIs

$$\left[\begin{array}{cc} Q+\gamma H^{(ij)}-P_{i-1} & q+\gamma g^{(ij)}-p_{i-1} \\ q^T+\gamma g^{(ij)T}-p_{i-1}^T & l_j+\gamma c^{(ij)}-s_{i-1} \end{array}\right] \succeq 0, \quad i = 1, \ldots, M, \quad j = 1, \ldots, N. \tag{33}$$

Clearly, (33) is convex in the variables P_i, p_i, s_i, and hence is tractable. The bound optimization problem is therefore a convex optimization problem and can be efficiently solved.

7.2. Numerical examples

We compute our bounds for a randomly generated example, and compare them to the performance achieved by the greedy and ADP policies. Our example is a problem with $n = 3$ and $N = 6$. The matrices $A_1, \ldots, A_N$, and $b_1, \ldots, b_N$ are randomly generated, with entries drawn from a standard normal distribution. Each A_i is then scaled so that its singular values are between 0.9 and 1. The stage cost matrices are $Q = I$, $q = 0$, $l = 0$, and we take $\gamma = 0.9$. We assume that the disturbance w_t has distribution $\mathcal{N}(0, 0.05I)$, and the initial state x_0 has distribution $\mathcal{N}(0, 10I)$.

The results are shown in table III. The pointwise supremum bound is computed via Monte Carlo simulation, using an iterated Bellman inequality condition with $M = 50$. Again we see that our best bound, the pointwise supremum bound, is very close to the performance of the ADP policy (within 10%).

8. MULTI-PERIOD PORTFOLIO OPTIMIZATION

The state (portfolio) $x_t \in \mathbf{R}_+^n$ is a vector of holdings in n assets at the beginning of period t, in dollars (not shares), so $\mathbf{1}^T x_t$ is the total portfolio value at time t. In this example we will assume that the portfolio is long only, *i.e.*, $x_t \in \mathbf{R}_+^n$, and that the initial portfolio x_0 is given. The input u_t is a vector of trades executed at the beginning of period t, also denominated in dollars: $(u_t)_i > 0$ means we purchase asset i, and $(u_t)_i < 0$ means we sell asset i. We will assume that $\mathbf{1}^T u_t = 0$, which means that the total cash obtained from sales equals the total cash required for the purchases, *i.e.*, the trades are self-financing. The trading incurs a quadratic transaction cost $u_t^T R u_t$, where $R \succeq 0$, which we will take into account directly in our objective function described below.

The portfolio propagates (over an investment period) as

$$x_{t+1} = A_t(x_t + u_t), \quad t = 0, 1, \ldots,$$

DOI: 10.1002/rnc

where $A_t = \mathbf{diag}(r_t)$, and r_t is a vector of random positive (total) returns, with $r_0, r_1, \ldots$ IID with known distribution on $\mathbf{R}^n_{++}$. We let $\mu = \mathbf{E}\, r_t$ be the mean of r_t, and $\Sigma = \mathbf{E}\, r_t r_t^T$ its second moment.

Our investment earnings in period t (*i.e.*, increase in total portfolio value), conditioned on $x_t = z$ and $u_t = v$, is $\mathbf{1}^T A_t(z+v) - \mathbf{1}^T z$, which has mean and variance

$$(\mu - \mathbf{1})^T(z+v), \quad (z+v)^T(\Sigma - \mu\mu^T)(z+v),$$

respectively. We will use a traditional risk adjusted mean earnings utility function (which is to be maximized),

$$U(z+v) = (\mu - \mathbf{1})^T(z+v) - \lambda(z+v)^T(\Sigma - \mu\mu^T)(z+v),$$

where $\lambda > 0$ is the risk aversion parameter. The stage utility is a concave quadratic function.

The stage cost (to be minimized) is

$$\ell(z,v) = \begin{cases} -U(z+v) + v^T R v & (z,v) \in \mathcal{C} \\ +\infty & (z,v) \notin \mathcal{C}, \end{cases}$$

where

$$\mathcal{C} = \{(z,v) \mid z + v \geq 0,\ \mathbf{1}^T v = 0\}.$$

Thus our stage cost (to be minimized) is the negative utility, adjusted to account for transaction cost. It is convex quadratic, on a set defined by some linear equality and inequality constraints. We will write the quadratic part of the stage cost as

$$-U(z+v) + v^T R v = \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T F \begin{bmatrix} v \\ z \\ 1 \end{bmatrix},$$

where

$$F = \begin{bmatrix} Q+R & Q & (\mathbf{1}-\mu)/2 \\ Q & Q & (\mathbf{1}-\mu)/2 \\ (\mathbf{1}-\mu)^T/2 & (\mathbf{1}-\mu)^T/2 & 0 \end{bmatrix},$$

with $Q = \lambda(\Sigma - \mu\mu^T)$.

8.1. Iterated Bellman inequality

We will look for quadratic candidate functions $\hat{V}_0, \ldots, \hat{V}_M$:

$$\hat{V}_i(z) = z^T P_i z + 2p_i^T z + s_i, \quad i = 0, \ldots, M,$$

where $P_i \in \mathbf{S}^n$, $p_i \in \mathbf{R}^n$, $s_i \in \mathbf{R}$, $i = 0, \ldots, M$, and $\hat{V} = \hat{V}_0 = \hat{V}_M$. We write this as

$$\hat{V}_i(z) = \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T S_i \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}, \qquad S_i = \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_i & p_i \\ 0 & p_i^T & s_i \end{bmatrix},$$

for $i = 0, \ldots, M$.

The iterated Bellman inequality (18) is:

$$\hat{V}_{i-1}(z) \leq \ell(z,v) + \gamma\, \mathbf{E}\, \hat{V}_i(A_t(z+v)), \quad i = 1, \ldots, M, \tag{34}$$

for all $z + v \geq 0$, $\mathbf{1}^T v = 0$. The expectations above can be evaluated as

$$\begin{aligned} \mathbf{E}\, \hat{V}_i(A_t y) &= \mathbf{E}\left(y^T A_t^T P_i A_t y + 2p_i^T A_t y + s_i\right) \\ &= y^T(\mathbf{E}\, A_t^T P_i A_t) y + 2p_i^T(\mathbf{E}\, A_t y) + s_i \\ &= y^T\left(\Sigma \circ P_i\right) y + 2(\mu \circ p_i)^T y + s_i, \end{aligned}$$

DOI: 10.1002/rnc

where $\circ$ denotes the Hadamard or elementwise product. Therefore we have

$$\mathbf{E}\,\hat{V}_i(A_t(z+v)) = \begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T G_i \begin{bmatrix} v \\ z \\ 1 \end{bmatrix},$$

where

$$G_i = \begin{bmatrix} \Sigma \circ P_i & \Sigma \circ P_i & \mu \circ p_i \\ \Sigma \circ P_i & \Sigma \circ P_i & \mu \circ p_i \\ (\mu \circ p_i)^T & (\mu \circ p_i)^T & s_i \end{bmatrix}.$$

Putting these together, we can write the iterated Bellman inequality as

$$\begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T (\gamma G_i + F - S_{i-1}) \begin{bmatrix} v \\ z \\ 1 \end{bmatrix} \geq 0$$

whenever $z + v \geq 0$ and $\mathbf{1}^T v = 0$. We express these last conditions as

$$\begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 & e_i \\ 0 & 0 & e_i \\ e_i^T & e_i^T & 0 \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix} \geq 0, \quad i = 1, \ldots, n,$$

and

$$\begin{bmatrix} v \\ z \\ 1 \end{bmatrix}^T \begin{bmatrix} 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 \\ \mathbf{1}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ z \\ 1 \end{bmatrix} = 0.$$

Finally, we can use the $\mathcal{S}$-procedure to find a sufficient condition for the Bellman inequalities: There exist $\nu_i \in \mathbf{R}$, and $\lambda^{(i)} \in \mathbf{R}^n_+$, $i = 1, \ldots, M$ such that for $i = 1, \ldots, M$,

$$\gamma G_i + F - S_{i-1} - \begin{bmatrix} 0 & 0 & \lambda^{(i)} + \nu_i \mathbf{1} \\ 0 & 0 & \lambda^{(i)} \\ (\lambda^{(i)} + \nu_i \mathbf{1})^T & \lambda^{(i)T} & 0 \end{bmatrix} \succeq 0. \tag{35}$$

Since G_i and S_i are linear functions of P_i, p_i, and s_i, (35) is a set of LMIs in the variables P_i, p_i, s_i, $\lambda^{(i)}$ and ν_i.

Thus, the bound optimization problem (14) becomes the SDP

$$\begin{array}{ll} \text{maximize} & \hat{V}_0(x_0) = \mathbf{Tr}(P_0 x_0 x_0^T) + p_0^T x_0 + s_0 \\ \text{subject to} & (35),\ \lambda^{(i)} \succeq 0, \quad i = 1, \ldots, M, \end{array} \tag{36}$$

with the variables listed above.

8.2. Numerical example

We consider a problem with $n = 3$ assets, with the last asset corresponding to a cash account. We take the total returns r_t to be log-normal, $\log r_t \sim \mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$, where $\tilde{\mu}$ and $\tilde{\Sigma}$ are the mean and variance of the log returns, which we take to be

$$\tilde{\mu} = \begin{bmatrix} 0.10 \\ 0.05 \\ 0 \end{bmatrix}, \quad \tilde{\Sigma} = \begin{bmatrix} (0.10)^2 & (0.1)(0.05)(0.3) & 0 \\ (0.1)(0.05)(0.3) & (0.05)^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The first asset has a mean log return and standard deviation of 0.10, the second asset has a mean log return and standard deviation of 0.05, and the cash account earns no interest. The first two asset log

DOI: 10.1002/rnc

Policy	**Objective**
ADP, $V^{\mathrm{adp}} = V^{\mathrm{unc}}$	-1.68
ADP, $V^{\mathrm{adp}} = \hat{V}$ ($M = 150$ iterated bound)	-1.96
Performance bound	**Value**
Iterated bound ($M = 150$)	-2.16
Basic Bellman inequality bound ($M = 1$)	-2.82
Without long-only constraint	-4.19

Table IV: Performance of suboptimal policies (top half) and performance bounds (bottom half) for portfolio optimization example.

returns are 30% correlated. The associated mean and second moment returns are

$$\mu_i = \mathbf{E}(r_t)_i = \exp(\tilde{\mu}_i + \tilde{\Sigma}_{ii}/2),$$

and

$$\begin{aligned} \Sigma_{ij} &= \mathbf{E}(r_t)_i(r_t)_j = \mathbf{E}\exp(w_i + w_j) \\ &= \exp(\tilde{\mu}_i + \tilde{\mu}_j + (\tilde{\Sigma}_{ii} + \tilde{\Sigma}_{jj} + 2\tilde{\Sigma}_{ij})/2) \\ &= \mu_i\mu_j \exp\tilde{\Sigma}_{ij}. \end{aligned}$$

We take $x_0 = (0, 0, 1)$, *i.e.*, an all cash initial portfolio. We take transaction cost parameter $R = \mathbf{diag}(1, 0.5, 0)$, risk aversion parameter $\lambda = 0.1$, and discount factor $\gamma = 0.9$.

Numerical results. We compute several performance bounds for this problem. The simplest bound is obtained by ignoring the long-only constraint $z + v \geq 0$. The resulting problem is then linear quadratic, so the optimal value function is quadratic, the optimal policy is affine, and we can evaluate its cost exactly (*i.e.*, without resorting to Monte Carlo simulation). The next bound is the basic Bellman inequality bound, *i.e.*, the iterated bound with $M = 1$. Our most sophisticated bound is the iterated bound, with $M = 150$. (We increased M until no significant improvement in the bound was observed.) Using Monte Carlo simulation, we evaluated the objective for the greedy policy and the ADP policy, using $V^{\mathrm{adp}} = \hat{V}$, obtained from the iterated Bellman bound.

We compare these performance bounds with the performance obtained by two ADP policies. The first ADP policy is a 'naive' policy, where we take V^{adp} to be the optimal value function of the same problem without the long-only constraint, V^{unc}. In the second ADP policy we take $V^{\mathrm{adp}} = \hat{V}$ from our iterated bellman bound.

The results are shown in table IV. We can see that the basic Bellman inequality bound outperforms the bound we obtain by ignoring the long-only constraint, while the iterated bound with $M = 150$ is better than both. The ADP policy with $V^{\mathrm{adp}} = V^{\mathrm{unc}}$ performs worse compared with the ADP policy with $V^{\mathrm{adp}} = \hat{V}$, which performs very well. The gap between the cost achieved by the ADP policy with $V^{\mathrm{adp}} = \hat{V}$ and the iterated Bellman inequality bound is small, which tells us that the ADP policy is nearly optimal.

Figure 3 shows a histogram of costs achieved by the two ADP policies over 10000 runs, where each run simulates the system with the ADP policy over 100 time steps.

9. CONCLUSIONS AND COMMENTS

9.1. Extensions and variations

In this paper we focussed mainly on cases where the dynamical system is linear, and the cost functions are quadratic. The same methods we used directly extends to problems with polynomial

Prepared using rncauth.cls
DOI: 10.1002/rnc

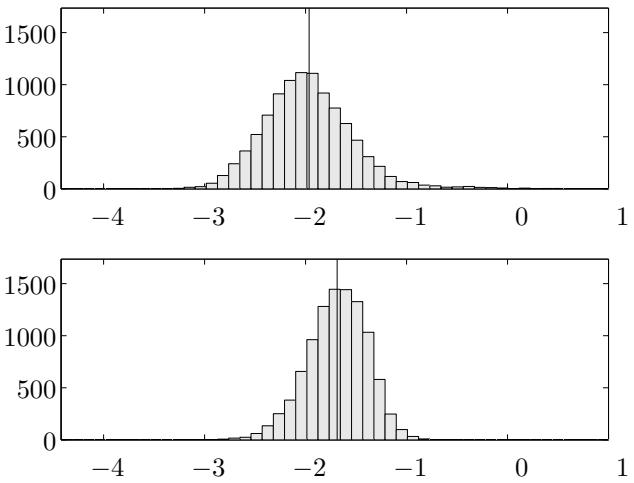

Figure 3: Histogram of costs over 10000 runs. *Top*: ADP policy with $V^{\mathrm{adp}} = \hat{V}$. *Bottom*: ADP policy with $V^{\mathrm{adp}} = V^{\mathrm{unc}}$. Vertical lines indicate means of each distribution.

dynamics functions, stage costs and constraints. In this case, we look for polynomial $\hat{V}_0, \ldots, \hat{V}_M$. The development of the bound is exactly the same as for the linear quadratic case, except that to get a sufficient condition for (18), we use the sum-of-squares (SOS) procedure instead of the $\mathcal{S}$-procedure. See [64, 65] for more on SOS, and [66, 67, 13] for other work on value function approximation with polynomial data. The resulting set of inequalities is still convex, with a tractable number of variables and constraints when the degree of the polynomials is not too large.

There are many other simple extensions. For instance, we can easily extend the affine switching example to include both state and input dependent switching (and also combine this with polynomial dynamics and costs). For arbitrary dynamics, costs and constraints, the iterated Bellman condition is a semi-infinite constraint, and is difficult to handle in general. In this case, we can use similar constraint sampling methods as in [9] to obtain good approximate value functions, but these are not guaranteed to be value function underestimators.

9.2. Implementation

For problems described in §6 and §8, evaluating the ADP policy reduces to solving a small convex quadratic program (QP), where the number of variables is equal to the number of inputs m. Recent advances allow such problems to be solved at stunning speeds. One popular approach is to solve the QP explicitly as a function of the problem parameters [68, 69], in which case evaluating the control policy reduces to searching through a look-up table. This works very well for problems where the numbers of states and inputs are small (around $n = 5$, $m = 5$ or less). The method is less practical for larger problems, since the number of entries in the look-up table can be very large. However, there are many ways to reduce the complexity of the explicit solution in these cases [69, 70, 71, 72].

Another method is to solve the QP on-line, in real-time, exploiting the structure in the problem, which results in extremely fast solve times [6]. To give an idea of the speeds, for a problem with 100 states and 10 inputs, the quadratic ADP policy can be evaluated in around $67\mu s$ on a 2GHz AMD processor. Recent advances in optimization modeling and code generation make it possible to automatically generate solvers that exploit problem specific sparsity structure, further reducing computation times [7].

The ability to solve these optimization problems at very high speeds means that the techniques described in this paper can be used for stochastic control problems with fast sample times, measured

DOI: 10.1002/rnc

in kHz (thousands of samples per second). Even in applications where such speeds are not needed, the high solution speed is very useful for simulation, which requires the solution of a very large number of QPs.

9.3. Summary

In this paper we have outlined a method for finding both a lower bound on the optimal objective value of a stochastic control problem, as well as a policy that often comes close in performance. We have demonstrated this on several examples, where we showed that the bound is close to the performance of the ADP policy. Our method is based on solving linear and semidefinite programming problems, hence is tractable even for problems with high state and input dimension.

ACKNOWLEDGMENTS

The authors thank Mark Mueller, Ben Van Roy, Sanjay Lall, Ciamac Moallemi, Vivek Farias, David Brown, Carlo Savorgnan, and Moritz Diehl for helpful discussions.

A. QUADRATIC FUNCTIONS AND THE $\mathcal{S}$-PROCEDURE

In this appendix we outline a basic result called the $\mathcal{S}$-procedure [54, §B.2][12, §2.6.3], which we can use to derive tractable convex conditions on the coefficients, expressed as linear matrix inequalities, that guarantee the iterated Bellman inequality holds. Using these conditions, the bound optimization problems will become semdefinite programs.

A.1. Quadratic functions and linear matrix inequalities

Quadratic functions. We represent a general quadratic function g in the variable $z \in \mathbf{R}^n$ as a quadratic form of $(z, 1) \in \mathbf{R}^{n+1}$, as

$$g(z) = z^T P z + 2p^T z + s,$$

where $P \in \mathbf{S}^n$ (the set of symmetric $n \times n$ matrices), $p \in \mathbf{R}^n$ and $s \in \mathbf{R}$. Thus g is a linear combination of the quadratic functions, $x_i x_j$, $i, j = 1, \ldots, n$, $i \leq j$, the linear functions x_i, $i = 1, \ldots, n$ and the constant 1, where the coefficients are given by the matrices P, p and s.

Global nonnegativity. For a quadratic function we can express global nonnegativity in a simple way:

$$g \geq 0 \iff \left[\begin{array}{cc} P & p \\ p^T & s \end{array} \right] \succeq 0, \tag{37}$$

where the inequality on the left is pointwise (*i.e.*, for all $z \in \mathbf{R}^n$), and the righthand inequality $\succeq$ denotes matrix inequality. Since we can easily check if a matrix is positive semidefinite, global nonnegativity of a quadratic function is easy to check. (It is precisely this simple property that will give us tractable nonheuristic conditions that imply that the Bellman inequality, or iterated Bellman inequality, holds on state spaces such as $\mathcal{X} = \mathbf{R}^{30}$, where sampling or exhaustive search would be entirely intractable.)

Linear matrix inequalities. A linear matrix inequality (LMI) in the variable $x \in \mathbf{R}^n$ has the form

$$F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n \succeq 0,$$

for matrices $F_0, \ldots, F_n \in \mathbf{S}^m$. LMIs define convex sets; and we can easily solve LMIs, or more generally convex optimization problems that include LMIs, using standard convex optimization techniques; see, *e.g.*, [12, 54, 73, 74].

As a simple example, the condition that $g \geq 0$ (pointwise) is equivalent to the matrix inequality in (37), which is an LMI in the variables P, p, and s.

DOI: 10.1002/rnc

A.2. $\mathcal{S}$-procedure

Let g be a quadratic function in the variable $z \in \mathbf{R}^n$, with associated coefficients (P, p, s). We seek a sufficient condition for g to be nonnegative on a set $\mathcal{Q}$ defined by a set of quadratic equalities and inequalities, *i.e.*,

$$g(z) \geq 0, \quad \forall z \in \mathcal{Q}, \tag{38}$$

where

$$\mathcal{Q} = \{z \mid g_1(z) \geq 0, \ldots, g_r(z) \geq 0,\ g_{r+1}(z) = \cdots = g_N(z) = 0\},$$

and

$$g_i(z) = z^T P_i z + 2p_i^T z + s_i, \quad i = 1, \ldots, N.$$

One simple condition that implies this is the existence of nonnegative $\lambda_1, \ldots, \lambda_r \in \mathbf{R}$, and arbitrary $\lambda_{r+1}, \ldots, \lambda_N \in \mathbf{R}$, for which

$$g(z) \geq \sum_{i=1}^{N} \lambda_i g_i(z), \quad \forall z \in \mathbf{R}^n. \tag{39}$$

(The argument is simple: for $z \in \mathcal{Q}$, $g_i(z) \geq 0$ for $i = 1, \ldots, r$, and $g_i(z) = 0$ for $i = r+1, \ldots, N$, so the righthand side is nonnegative.) But (39) is equivalent to

$$\left[\begin{array}{cc} P & p \\ p^T & s \end{array}\right] - \sum_{i=1}^{N} \lambda_i \left[\begin{array}{cc} P_i & p_i \\ p_i^T & s_i \end{array}\right] \succeq 0, \tag{40}$$

which is an LMI in the variables P, p, s and $\lambda_1, \ldots, \lambda_N$ (with P_i, p_i, and s_i, for $i = 1, \ldots, N$ considered data). (We also have nonnegativity conditions on $\lambda_1, \ldots, \lambda_r$.) The numbers λ_i are called *multipliers*.

This so-called $\mathcal{S}$-procedure gives a sufficient condition for the (generally) infinite number of inequalities in (38) (one for each $z \in \mathcal{Q}$) as a single LMI that involves a finite number of variables. In some special cases, the $\mathcal{S}$-procedure condition is actually equivalent to the inequalities; but for our purposes here we only need that it is a sufficient condition, which is obvious. The $\mathcal{S}$-procedure generalizes the (global) nonnegativity condition (37), which is obtained by taking $\lambda_i = 0$.

Example. As an example, let us derive an LMI condition on P, p, s (and some multipliers) that guarantees $g(z) \geq 0$ on $\mathcal{Q} = \mathbf{R}^n_+$. (When g is a quadratic form, this condition is the same as *copositivity* of the matrix, which is not easy to determine [75].) We first take the quadratic inequalities defining $\mathcal{Q}$ to be the linear inequalities $2z_i \geq 0$, $i = 1, \ldots, n$, which correspond to the coefficient matrices

$$\left[\begin{array}{cc} 0 & e_i \\ e_i^T & 0 \end{array}\right], \quad i = 1, \ldots, n,$$

where e_i is the ith standard unit vector. The $\mathcal{S}$-procedure condition for $g(z) \geq 0$ on $\mathbf{R}^n_+$ is then

$$\left[\begin{array}{cc} P & p - \lambda \\ (p-\lambda)^T & s \end{array}\right] \succeq 0,$$

for some $\lambda \in \mathbf{R}^n_+$.

We can derive a stronger $\mathcal{S}$-procedure condition by using a larger set of (redundant!) inequalities to define $\mathcal{Q}$:

$$2z_i \geq 0, \quad i = 1, \ldots, n, \quad 2z_i z_j \geq 0, \quad i, j = 1, \ldots, n,\ i < j,$$

which correspond to the coefficient matrices

$$\left[\begin{array}{cc} 0 & e_i \\ e_i^T & 0 \end{array}\right], \quad i = 1, \ldots, n, \qquad \left[\begin{array}{cc} e_i e_j^T + e_j e_i^T & 0 \\ 0 & 0 \end{array}\right], \quad i, j = 1, \ldots, n,\ i < j.$$

DOI: 10.1002/rnc

The $\mathcal{S}$-procedure condition for $g(z) \geq 0$ on $\mathbf{R}^n_+$ is then

$$\left[\begin{array}{cc} P - \Lambda & p - \lambda \\ (p - \lambda)^T & s \end{array} \right] \succeq 0, \tag{41}$$

for some $\Lambda \in \mathbf{S}^n$ with all entries nonnegative and zero diagonal entries, and some $\lambda \in \mathbf{R}^n_+$. The condition (41) is an LMI in $P, p, s, \Lambda, \lambda$.

For fixed P, p, s, the sufficient condition (41) for copositivity is interesting. While it is not in general a necessary condition for copositivity, it is a sophisticated, and tractably computable, sufficient condition. Even more interesting is that we can tractably solve (convex) optimization problems over P, p, s using the LMI sufficient condition (41) (which implies copositivity).

DOI: 10.1002/rnc

REFERENCES

1. Kalman R. When is a linear control system optimal? *Journal of Basic Engineering* 1964; **86**(1):1–10.
2. Bertsekas D. *Dynamic Programming and Optimal Control: Volume 1*. Athena Scientific, 2005.
3. Bertsekas D. *Dynamic Programming and Optimal Control: Volume 2*. Athena Scientific, 2007.
4. Bertsekas D, Shreve S. *Stochastic optimal control: The discrete-time case*. Athena Scientific, 1996.
5. Powell W. *Approximate dynamic programming: solving the curses of dimensionality*. John Wiley & Sons, Inc., 2007.
6. Wang Y, Boyd S. Fast evaluation of control-Lyapunov policy 2009. Manuscript.
7. Mattingley J, Boyd S. Automatic code generation for real-time convex optimization. *Convex optimization in signal processing and communications*, 2009. To appear.
8. Wegbreit B, Boyd S. Fast computation of optimal contact forces. *IEEE Transactions on Robotics* Dec 2007; **23**(6):1117–1132.
9. De Farias D, Van Roy B. The linear programming approach to approximate dynamic programming. *Operations Research* 2003; **51**(6):850–865.
10. Wang Y, Boyd S. Performance bounds for linear stochastic control. *System and Control Letters* 2009; **58**(3):178–182.
11. Wang Y, Boyd S. Performance bounds and suboptimal policies for linear stochastic control via LMIs 2009. Manuscript, available at `www.stanford.edu/~boyd/papers/gen_ctrl_bnds.html`.
12. Boyd S, El Ghaoui L, Feron E, Balakrishnan V. *Linear Matrix Inequalities in Systems and Control Theory*. SIAM books: Philadelphia, 1994.
13. Savorgnan C, Lasserre J, Diehl M. Discrete-time stochastic optimal control via occupation measures and moment relaxations. *Proceedings of the 48th IEEE Conference on Decision and Control*, 2009; 4939–4944.
14. Bertsimas D, Caramanis C. Bounds on linear PDEs via semidefinite optimization. *Mathematical Programming, Series A* 2006; **108**(1):135–158.
15. Lincoln B, Rantzer A. Relaxing dynamic programming. *IEEE Transactions on Automatic Control* 2006; **51**(8):1249–1260.
16. Rantzer A. Relaxed dynamic programming in switching systems. *IEE Proceedings — Control Theory and Applications* 2006; **153**(5):567–574.
17. Manne A. Linear programming and sequential decisions. *Management Science* 1960; **60**(3):259–267.
18. Schweitzer P, Seidmann A. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications* 1985; **110**(2):568–582.
19. Kumar S, Kumar P. Performance bounds for queueing networks and scheduling policies. *IEEE Transactions on Automatic Control* 1994; **39**(8):1600–1611.
20. Morrison J, Kumar P. New linear program performance bounds for queueing networks. *Journal of Optimization Theory and Applications* 1999; **100**(3):575–597.
21. Moallemi C, Kumar S, Van Roy B. Approximate and data-driven dynamic programming for queueing networks 2008. Manuscript.
22. Adelman D. Dynamic bid prices in revenue management. *Operations Research* 2007; **55**(4):647–661.
23. Adelman D. A price-directed approach to stochastic inventory/routing. *Operations Research* 2004; **52**(4):449–514.
24. Farias V, Van Roy B. An approximate dynamic programming approach to network revenue management 2007. Manuscript.
25. Farias V, Saure D, Weintraub G. An approximate dynamic programming approach to solving dynamic oligopoly models 2010. Manuscript.
26. Han J. Dynamic portfolio management—an approximate linear programming approach. PhD Thesis, Stanford University 2005.
27. Cogill R, Rotkowitz M, Van Roy B, Lall S. An approximate dynamics programming approach to decentralized control of stochastic systems. *Control of uncertain systems: Modelling, Approximation and Design*, 2006; 243–256.
28. Bertsimas D, Iancu D, Parrilo P. Optimality of affine policies in multi-stage robust optimization 2009. Manuscript.
29. Desai V, Farias V, Moallemi C. A smoothed approximate linear program. *Advances in Neural Information Processing Systems* 2009; **22**:459–467.
30. Cogill R, Lall S. Suboptimality bounds in stochastic control: A queueing example. *Proceedings of the 2006 American Control Conference*, 2006; 1642–1647.
31. Cogill R, Lall S, Hespanha J. A constant factor approximation algorithm for event-based sampling. *Proceedings of the 2007 American Control Conference*, 2007; 305–311.
32. Brown D, Smith J, Sun P. Information relaxations and duality in stochastic dynamic programs. *Operations Research* 2010; To appear.
33. Bertsimas D, Gamarnik D, Tsitsiklis J. Performance of multiclass Markovian queueing networks via piecewise linear Lyapunov functions. *Annals of Applied Probability* 2001; **11**(4):1384–1428.
34. Castañón D. Stochastic control bounds on sensor network performance. *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005; 4939–4944.
35. Altman E. *Constrained Markov Decision Processes*. Chapman & Hall, 1999.
36. Peters A, Salgado M, Silva-Vera E. Performance bounds in MIMO linear control with pole location constraints. *Proceedings of the 2007 Mediterranean Conference on Control and Automation*, 2007; 1–6.
37. Vuthandam P, Genceli H, Nikolaou M. Performance bounds for robust quadratic dynamic matrix control with end condition. *AIChE Journal* 2004; **41**(9):2083–2097.
38. Bertsekas D, Castañón D. Adaptive aggregation methods for infinite horizon dynamic programming. *IEEE Transactions on Automatic Control* 1989; **34**(6):589–598.
39. Sutton R, Barto A. *Reinforcement learning: An introduction*. Cambridge Univ Press, 1998.

DOI: 10.1002/rnc

40. Ziv O, Shimkin N. Multigrid algorithms for temporal difference reinforcement learning. *Proc. ICML workshop on rich representations for RL*, 2005.
41. Menache I, Mannor S, Shimkin N. Basis function adaptation in temporal difference reinforcement learning. *Annals of Operations Research* 2005; **134**(1):215–238.
42. Smart W. Explicit manifold representations for value-function approximation in reinforcement learning. *Proc. of the 8th international symposium on AI and mathematics*, 2004.
43. Mahadevan S. Samuel meets Amarel: Automating value function approximation using global state space analysis. *Proc. of the 20th National Conference on Artificial Intelligence*, vol. 5, 2005; 1000–1005.
44. Keller P, Mannor S, Precup D. Automatic basis function construction for approximate dynamic programming and reinforcement learning. *Proc. of the 23rd international conference on Machine learning*, ACM, 2006; 449–456.
45. Huizhen Y, Bertsekas D. Basis function adaptation methods for cost approximation in MDP. *2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 2009; 74–81.
46. Witsenhausen H. On performance bounds for uncertain systems. *SIAM Journal on Control* 1970; **8**(1):55–89.
47. Rieder U, Zagst R. Monotonicity and bounds for convex stochastic control models. *Mathematical Methods of Operations Research* 1994; **39**(2):1432–5217.
48. McEneaney W. A curse-of-dimensionality-free numerical method for solution of certain HJB PDEs. *SIAM Journal on Control and Optimization* 2007; **46**(4):1239–1276.
49. Whittle P. *Optimization over Time*. John Wiley & Sons, Inc., 1982.
50. Sontag E. A Lyapunov-like characterization of asymptotic controllability. *SIAM Journal on Control and Optimization* 1983; **21**(3):462–471.
51. Freeman R, Primbs J. Control Lyapunov functions, new ideas from an old source. *Proceedings of the 35th IEEE Conference on Decision and Control*, vol. 4, 1996; 3926–3931.
52. Corless M, Leitmann G. Controller design for uncertain systems via Lyapunov functions. *Proceedings of the American Control Conference*, vol. 3, 1988; 2019–2025.
53. Sznaier M, Suarez R, Cloutier J. Suboptimal control of constrained nonlinear systems via receding horizon constrained control Lyapunov functions. *International Journal on Robust and Nonlinear Control* 2003; **13**(3-4):247–259.
54. Boyd S, Vandenberghe L. *Convex Optimization*. Cambridge University Press, 2004.
55. Nocedal J, Wright S. *Numerical Optimization*. Springer, 1999.
56. Vandenberghe L, Boyd S. Semidefinite programming. *SIAM Review* 1996; **38**(1):49–95.
57. Potra F, Wright S. Interior-point methods. *Journal of Computational and Applied Mathematics* 2000; **124**(1-2):281–302.
58. Wang Y, Boyd S. Fast model predictive control using online optimization. *Proceedings of the 17th IFAC world congress*, 2008; 6974–6997.
59. Skaf J, Boyd S. Techniques for exploring the suboptimal set. *Optimization and Engineering* 2010; :1–19.
60. De Farias D, Van Roy B. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research* 2004; **29**(3):462–478.
61. Mutapcic A, Boyd S. Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods and Software* 2009; **24**(3):381–406.
62. Geyer T, Papafotiou G, Morari M. On the optimal control of switch-model DC-DC converters. *Hybrid Systems: Computation and Control*, 2004.
63. Prodic A, Maksimovic D, Erickson R. Design and implementation of a digital PWM controller for a high-frequency switching DC-DC power converter. *Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society*, 2001; 893–898.
64. Parrilo P. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming Series B* 2003; **96**(2):293–320.
65. Parrilo P, Lall S. Semidefinite programming relaxations and algebraic optimization in control. *European Journal of Control* 2003; **9**(2-3):307–321.
66. Henrion D, Lasserre J, Savorgnan C. Nonlinear optimal control synthesis via occupation measures. *Proceedings of the 47th IEEE Conference on Decision and Control*, 2008; 4749–4754.
67. Lasserre J, Henrion D, Prieur C, Trelat E. Nonlinear optimal control via occupation measures and LMI-relaxations. *SIAM Journal on Control and Optimization* June 2008; **47**(4):1643–1666.
68. Bemporad A, Morari M, Dua V, Pistikopoulos E. The explicit linear quadratic regulator for constrained systems. *Automatica* Jan 2002; **38**(1):3–20.
69. Zeilinger M, Jones C, Morari M. Real-time suboptimal model predictive control using a combination of explicit MPC and online computation. *IEEE Conference on Decision and Control*, 2008; 4718–4723.
70. Christophersen C, Zeilinger M, Jones C, Morari M. Controller complexity reduction for piecewise affine systems through safe region elimination. *IEEE Conference on Decision and Control*, 2007; 4773–4778.
71. Jones C, Grieder P, Rakovic S. A logarithmic-time solution to the point location problem. *Automatica* Dec 2006; **42**(12):2215–2218.
72. Bemporad A, Filippi C. Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming. *Journal of Optimization Theory and Applications* Nov 2004; **117**(1):9–38.
73. Vandenberghe L, Balakrishnan V. Algorithms and software tools for LMI problems in control. *IEEE Control Systems Magazine*, 1997; 89–95.
74. Wolkowicz H, Saigal R, Vandenberghe L. *Handbook of Semidefinite Programming*. Kluwer Academic Publishers, 2000.
75. Johnson C, Reams R. Spectral theory of copositive matrices. *Linear algebra and its applications* 2005; **395**:275–281.