2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Convex Q Learning in a Stochastic Environment

Fan Lu and Sean P. Meyn*

Abstract— **The paper introduces the first formulation of convex Q-learning for Markov decision processes with function approximation. The algorithms and theory rest on a relaxation of a dual of Manne's celebrated linear programming characterization of optimal control. The main contributions firstly concern properties of the relaxation, described as a deterministic convex program: we identify conditions for a bounded solution, and a significant relationship between the solution to the new convex program, and the solution to standard Q-learning. The second set of contributions concern algorithm design and analysis: (i) A direct model-free method for approximating the convex program for Q-learning shares properties with its ideal. In particular, a bounded solution is ensured subject to a simple property of the basis functions; (ii) The proposed algorithms are convergent and new techniques are introduced to obtain the rate of convergence in a mean-square sense; (iii) The approach can be generalized to a range of performance criteria, and it is found that variance can be reduced by considering "relative" dynamic programming equations; (iv) The theory is illustrated with an application to a classical inventory control problem.**

I. Introduction

The Q-learning algorithm introduced in [24] is a highly celebrated approach to reinforcement learning, that has evolved over the past decades to form part of the solution to complex optimal control problems. It was originally designed to compute the state-action value function (known as the Q-function). This early work considered the discounted-cost optimal control problem for Markov decision processes (MDPs), in the *tabular* setting so that the function class spans all functions.

The ultimate goal then and now is to approximate the Q-function within a restricted function class, notably neural networks, though much of the theory is restricted to a linearly parameterized function class. Counterexamples show that conditions on the function class are required in general, even in a linear function approximation setting [1], [23]. Criteria for stability based on sufficient exploration are contained in the recent work [16].

Moreover, when convergent, the limit of Q-learning or DQN solves a "projected Bellman equation" (see (11)), but we know little about the implication of this conclusion. These concerns have motivated new ways of thinking about how to approximate a Q-function [16], [15].

The linear programming (LP) approach to optimal control pioneered by Manne [13] has inspired alternative approaches to RL and approximate dynamic programming. The earliest such work was found in [21], with error bounds appearing in [4], [3], [9]. Model-free algorithms appeared in [14], [10], [11] and [15, Ch. 5], where the term *convex Q-learning* (CvxQ) was coined. In parallel came *logistic Q-learning* [2], which solves a regularized dual of the LP in [10]. There is however a gap in the settings: CvxQ was developed for deterministic control systems, while logistic Q-learning treats MDPs. Also, the stochastic setting is so far restricted to tabular [2] or linearly factorable MDPs [18].

LP approaches are attractive because we obtain by design a convergent algorithm. Moreover, the L_∞-framework is more likely to lead to an interpretable solution, since performance bounds on the resulting feedback policy can be obtained through Lyapunov function techniques [4], [9]. The main contributions are summarized here:

i) Convex Q-learning for optimal control is introduced in a stochastic environment for the first time. It is found that the constraint region is bounded subject to a persistence of excitation, generalizing the conclusions obtained recently for deterministic optimal control problems [11]. Several approaches to approximating the solution to the convex program are proposed and analyzed.

ii) Prop. 2.4 implies a surprising connection between CvxQ and standard Q-learning.

iii) Techniques are introduced to obtain the rate of convergence in a mean-square sense—see Prop. 4.2.

Comparison with existing literature. The new algorithms and some of the analysis might be anticipated from the theory for deterministic control systems in [11]. Prop. 2.4 is new (and was initially surprising to us) even in the deterministic setting. The variance analysis surveyed in Prop. 4.2 is novel; Complementary results appeared in [19], motivated by MDP LP relaxations. Conclusions in this prior work is based on i.i.d. samples of trajectories, designed to permit application of Hoeffding's inequality to obtain sample complexity bounds for constraint-sampled LPs. The covariance formula in Prop. 4.2 is similar to what is anticipated from stochastic approximation (SA) theory, even though CvxQ falls outside of standard SA recursions —see discussion following the proposition.

Organization: Section II conditions foundations for the CvxQ algorithms introduced in Section III. Theory for convergence rates is contained in Section IV. The theory is illustrated in Section V with an application to a classical inventory control problem.

II. Q-learning Convex Programs

The control model for which algorithm design and analysis is based on the standard Markov Decision Process (MDP)

*SPM and FL are with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611; SPM holds an Inria International Chair, Paris, France. Financial support from ARO award W911NF2010055 and National Science Foundation award EPCN 1935389 is gratefully acknowledged.

979-8-3503-0123-6/23/$31.00 ©2023 IEEE

with finite state space X, finite input space U, and non-negative cost function $c : \mathsf{Z} \to \mathbb{R}_+$, with $\mathsf{Z} := \mathsf{X} \times \mathsf{U}$.

The state process is denoted $\boldsymbol{X} = \{X(k) : k \geq 0\}$, the input (or action) sequence $\boldsymbol{U} = \{U(k) : k \geq 0\}$, and the pair process $\boldsymbol{Z} = (\boldsymbol{X}, \boldsymbol{U})$. The controlled transition matrix is denoted P_u for $u \in \mathsf{U}$, so that $\mathsf{P}\{X(k+1) = x' \mid X(k) = x, U(k) = u\} = P_u(x, x')$; it acts on functions $V : \mathsf{X} \to \mathbb{R}$ via $P_u V(x) := \sum_{x' \in \mathsf{X}} P_u(x, x') V(x')$.

Choice of training input. Theory and analysis are couched in a stationary setting: the input used for training is defined by a randomized stationary policy, so that the joint process $\boldsymbol{Z}$ is a time homogeneous Markov chain. It is assumed uni-chain, with unique invariant pmf denoted ϖ[1].

Theory is also restricted to the discounted-cost optimality criterion, with discount factor $\gamma \in (0, 1)$. The Q-function is the state-action value function,

$$Q^*(z) := \min_{\boldsymbol{U}} \sum_{k=0}^{\infty} \gamma^k \mathsf{E}[c(Z(k)) | Z(0) = z] , \quad z = (x, u) \in \mathsf{Z}$$

where the minimum is over all adapted input sequences. It is the unique solution to the Bellman equation,

$$Q^*(z) = c(z) + \gamma P_u \underline{Q}^*(x) \tag{1}$$

with $\underline{Q}(x) := \min_u Q(x, u)$. The optimal input is state feedback $U_k = \phi^*(X_k)$, using the "Q^*-greedy" policy,

$$\phi^*(x) \in \operatorname*{arg\,min}_{u \in \mathsf{U}} Q^*(x, u) , \qquad x \in \mathsf{X} \tag{2}$$

A. Convex programs for approximation

Convex Q-learning algorithm is motivated by the classical LP characterization of optimal control problem due to Manne [13]. The following is a simple corollary:

Proposition 2.1: For any pmf μ on Z, the Q-function Q^* is a solution to the convex program,

$$\begin{aligned} &\max_{Q} \ \langle \mu, Q \rangle \\ &\text{s.t.} \ \ Q(z) \leq c(z) + \gamma P_u \underline{Q}(x) , \quad z = (x, u) \in \mathsf{Z}. \end{aligned} \tag{3}$$

The proof follows from verification that $Q \leq Q^*$ whenever Q is feasible, which is a standard Lyapunov function argument.

This section is devoted to relaxations of (3) that are model-based. The conclusions motivate model-free algorithms and analysis in Section III.

To obtain a convex program we restrict to a linear family: $\{Q^\theta(x, u) = \theta^{\intercal} \psi(x, u) : \theta \in \mathbb{R}^d\}$, with $\psi : \mathsf{X} \times \mathsf{U} \to \mathbb{R}^d$ the vector of basis functions, and based on an appropriate approximation obtain a policy in analog with (2):

$$\phi^\theta(x) \in \operatorname*{arg\,min}_{u \in \mathsf{U}} Q^\theta(x, u) \tag{4}$$

The following is suggested by (3),

$$\max_{\theta} \ \langle \mu, Q^\theta \rangle \qquad \text{s.t.} \ \ Q^\theta(z) \leq c(z) + \gamma P_u \underline{Q}^\theta(x) \tag{5}$$

This is not practical in typical applications because it requires knowledge of the model, and there are so many constraints: one for each $z = (x, u) \in \mathsf{Z}$.

Galerkin relaxations Practical algorithms are obtained by expressing the constraints of (5) in sample path form: if a vector $\theta \in \mathbb{R}^d$ is feasible, then the following inequality is valid for *any* adapted input sequence: with $\mathcal{F}_k = \sigma(Z(i) : i \leq k)$ the filtration generated by the observations,

$$\begin{aligned} &\mathsf{E}\big[\mathcal{D}_{k+1}(\theta) | \mathcal{F}_k\big] \geq 0 , \quad \textit{for all } k \geq 0, \\ &\mathcal{D}_{k+1}(\theta) := -Q^\theta(Z(k)) + c(Z(k)) + \gamma \underline{Q}^\theta(X(k+1)) \end{aligned} \tag{6}$$

A relaxation of (5) is obtained by specifying a sequence of *non-negative* d_+-dimensional random vectors $\{\zeta_k : k \geq 0\}$, with $d_+ > d$. Denote $\overline{g}(\theta) := \mathsf{E}_\varpi\big[-\mathcal{D}_{k+1}(\theta)\zeta_k\big]$. A relaxation of (5) is then defined by

$$\max_{\theta} \ \ \langle \mu, Q^\theta \rangle \qquad \text{s.t.} \quad \overline{g}(\theta) \leq \ 0. \tag{7}$$

An equivalent LP formulation is required for analysis. Let Φ denote the set of all deterministic policies, and for each $\phi \in \Phi$ and $k \geq 0$ denote

$$\mathcal{D}_{k+1}(\theta, \phi) = c_{(k)} + \theta^{\intercal}\{-\psi_{(k)} + \gamma \psi^{\phi}_{(k+1)}\} , \tag{8}$$

in which the following conventions will be used to save space when necessary: $c_{(k)} := c(Z(k))$, and

$$\psi_{(k)} := \psi(X(k), U(k)) , \ \psi^{\phi}_{(k)} := \psi(X(k), \phi(X(k)))$$

Similar to (7), we denote $\overline{g}(\theta, \phi) := \mathsf{E}_\varpi\big[-\mathcal{D}_{k+1}(\theta, \phi)\zeta_k\big]$ for each $\phi \in \Phi$.

Proposition 2.2: Any solution to the *Q-learning convex program* (7) is also a solution to the linear program,

$$\max_{\theta} \ \langle \mu, Q^\theta \rangle \qquad \text{s.t.} \ \ \overline{g}(\theta, \phi) \leq 0 , \quad \phi \in \Phi , \tag{9}$$

with identical optimal values.

Proof. For any $\phi \in \phi$, we have

$$\mathsf{E}_\varpi\big[\mathcal{D}_{k+1}(\theta, \phi)\zeta^i_k\big] \geq \mathsf{E}_\varpi\big[\mathcal{D}_{k+1}(\theta)\zeta^i_k\big] , \quad 1 \leq i \leq d_+$$

The proof is completed on recognizing that this lower bound is achieved with $\phi = \phi^\theta$. □

Let $\Theta = \{\theta \in \mathbb{R}^d : \overline{g}(\theta) \leq 0\}$ denote the constraint set for (7). It is always non-empty since it contains the origin. Prop. 2.3 tells us that this set is bounded if the vectors $\{\psi_{(k)} : 0 \leq k\}$ are not restricted to any half space in $\mathbb{R}^d$, for almost every initial condition $[\varpi]$. See [12] for a proof.

Proposition 2.3: Suppose $\mathsf{P}_\varpi\{v^{\intercal}\psi_{(k)} \geq 0\} < 1$ for any non-zero $v \in \mathbb{R}^d$. Then, Θ is compact.

B. Comparison with Q-learning

The standard Q-learning algorithm is expressed,

$$\theta_{k+1} = \theta_k + \alpha_{k+1}\mathcal{D}_{k+1}(\theta_k)\zeta_k \tag{10}$$

where $\{\zeta_k\}$ is the sequence of d-dimensional eligibility vectors, typically taken as $\zeta_k = \nabla_\theta Q^\theta(Z(k))$ (which is $\psi_{(k)}$ with linear function approximation), and $\{\alpha_{k+1}\}$ the step-size sequence. When convergent, the limit θ^* solves the so-called *projected Bellman equation* (also known as a *Galerkin relaxation*),

$$\mathsf{E}_\varpi\big[\mathcal{D}_{k+1}(\theta^*)\zeta^i_k\big] = 0 , \qquad 1 \leq i \leq d , \tag{11}$$

[1] We often obtain faster convergence when using an epsilon-greedy policy; an explanation may be found through an extension of [16].

where the expectation is in steady-state [22], [15].

Prop. 2.4 that follows shows that (7) also solves a Galerkin relaxation. The proof follows from Prop. 2.2, and recognition that in (9) we may restrict to basic feasible solutions (BFS).

Proposition 2.4: If the convex program (7) admits at least one optimizer, then there is an optimizer θ^* together with indices $\{i_1, \dots, i_d\} \subset \{1, \dots, d_+\}$ satisfying

$$\mathsf{E}_\varpi\big[\mathcal{D}_{k+1}(\theta^*)\zeta_k^{i_\ell}\big] = 0\,, \qquad 1 \le \ell \le d\,. \tag{12}$$

III. Algorithms

In Convex Q-learning, the function $\overline{g} : \mathbb{R}^d \to \mathbb{R}^{d_+}$ in (7) is replaced by its approximation via Monte Carlo

$$\overline{g}_N(\theta) := \frac{1}{N}\sum_{k=0}^{N-1}[-\mathcal{D}_{k+1}(\theta)\zeta_k].$$

Convex Q-learning Given the data $\{Z(k) : 0 \le k < N\}$ and a pmf μ on Z, solve

$$\max_\theta \ \langle \mu, Q^\theta\rangle \qquad \text{s.t.} \quad \overline{g}_N(\theta) \le 0 \tag{13}$$

As we increase the time horizon N, the variance of the solution to (13) decreases, but the complexity of the linear program increases. The batch algorithms described next are designed to reduce complexity.

A. Batch algorithms

The two approaches below begin with the specification of intermediate times $T_0 = 0 < T_1 < T_2 < \cdots < T_{B-1} < T_B = N$. The parameter will be updated at these times to obtain $\{\theta_n : 0 \le n \le B\}$, initialized with $\theta_0 \in \mathbb{R}^d$. Also required are two positive step-size sequences satisfying $\lim_{n\to\infty} \alpha_n/\beta_n = 0$.

The empirical distribution over the nth batch of observations is denoted π_n. Hence, for any vector-valued function,

$$\langle \pi_n, g\rangle = \frac{1}{T_{n+1} - T_n}\sum_{k=T_n}^{T_{n+1}-1} g(\Phi_k)$$

In view of (7), we denote, for any $\theta \in \mathbb{R}^d$,

$$\langle \pi_n, \mathcal{D}(\theta)\zeta\rangle := \frac{1}{T_{n+1} - T_n}\sum_{k=T_n}^{T_{n+1}-1} \mathcal{D}_{k+1}(\theta)\zeta_k$$

We introduce a convex regularizer $\mathcal{R}_n$, so that the objective function at stage n of the algorithm becomes

$$\Gamma_n(\theta) := -\langle \mu, Q^\theta\rangle + \mathcal{R}_n(\theta)$$

and $\Gamma_n(\theta, \lambda) := \Gamma_n(\theta) - \langle \pi_n, \mathcal{D}(\theta)\zeta^\intercal\lambda\rangle$ for $\lambda \in \mathbb{R}_+^{d_+}$. Updates of λ are obtained to approximate the Lagrange multiplier associated with (7). Given parameter estimates $\{\theta_n : n \ge 0\}$, obtain a sequence of vectors in $\mathbb{R}^{d_+}$ via

$$v^{n+1} = v^n + \beta_{n+1}\big[\langle \pi_{n+1}, \mathcal{D}(\theta_{n+1})\zeta_n\rangle - v^n\big]$$

The sequence of Lagrange multiplier estimates are obtained via the recursion $\lambda_{n+1} = [\lambda_n + \alpha_{n+1}v^{n+1}]_+$, with $\lambda_0 \in \mathbb{R}_+^{d_+}$ an arbitrary initial condition and $[x]_+ = \max\{0, x\}$. Below are two choices for parameter updates:

Batch Convex Q-learning implicit update

$$\theta_{n+1} = \arg\min_\theta \Big\{\Gamma_n(\theta, \lambda_n) + \tfrac{1}{\alpha_{n+1}}\tfrac{1}{2}\|\theta - \theta_n\|^2\Big\}$$

It is called implicit because the solution is obtained via the fixed point equation,

$$\theta_{n+1} = \theta_n - \alpha_{n+1}\nabla_\theta\Gamma_n(\theta, \lambda_n)\Big|_{\theta=\theta_{n+1}} \tag{14}$$

The explicit update is obtained by introducing a one-step delay on the right-hand side.

Batch CvxQ explicit update

$$\theta_{n+1} = \theta_n - \alpha_{n+1}\nabla_\theta\Gamma_n(\theta, \lambda_n)\Big|_{\theta=\theta_n} \tag{15}$$

Assumptions on the regularizer are required to ensure convergence. For convenience we take

$$\mathcal{R}_n(\theta) = \kappa\{[\langle \pi_n, \mathcal{D}(\theta)\zeta^\intercal\lambda\rangle]_-\}^2 + \varepsilon\|\theta\|^2 \tag{16}$$

with $\kappa, \varepsilon > 0$ and $[x]_- := \max(0, -x)$. Let $\mathcal{R}(\theta)$ denote its steady-state mean, and

$$\overline{\Gamma}(\theta, \lambda) := -\langle \mu, Q^\theta\rangle - \langle \varpi, \mathcal{D}(\theta)\zeta^\intercal\lambda\rangle + \mathcal{R}(\theta)$$

Proposition 3.1: Consider either algorithm (implicit or explicit). The algorithm is convergent to a pair (θ^*, λ^*) that solves the saddle point problem: $\theta^* = \arg\min_\theta \overline{\Gamma}(\theta, \lambda^*)$, $\lambda^* = \arg\max_\lambda \min_\theta \overline{\Gamma}(\theta, \lambda)$.

The proof is a standard stochastic approximation analysis, in which the ODE approximation is precisely the *primal-dual flow* considered in [20], [7] and their references.

B. Relative Convex Q-Learning

Given any pmf ω on Z, denote $H^* := Q^* - \langle \omega, Q^*\rangle$. Since we are subtracting a constant, we have $\phi^*(x) = \arg\min_{u\in\mathsf{U}} H^*(x, u)$. The advantage of estimating H^* is that it remains bounded for $0 < \gamma < 1$ [6]. The extension of the preceding theory to this *relative Q-function* is straightforward, beginning with

Proposition 3.2: For any positive pmfs μ and ω on Z and positive scalar $\delta > 0$, H^* solves the convex program,

$$\begin{aligned}\max_H \ & \langle \mu, H\rangle \\ \text{s.t.} \ \ & H(z) \le c(z) + \gamma P_u\underline{H}(x) - \delta\langle \omega, H\rangle\,, \\ & \qquad \textit{for every } z = (x, u) \in \mathsf{Z}.\end{aligned} \tag{17}$$

This motivates one version of relative convex Q-learning:

Relative CvxQ Given the data $\{Z(k) : 0 \le k < N\}$ and probability measures μ and ω on Z, solve

$$\max_\theta \ \langle \mu, H^\theta\rangle \qquad \text{s.t.} \quad \overline{g}_N(\theta) \le 0 \tag{18}$$

where $\overline{g}_N$ is defined as in (13), with $\mathcal{D}_{k+1}$ replaced by the relative temporal difference:

$$\begin{aligned}\widehat{\mathcal{D}}_{k+1}(\theta) := &-H^\theta(Z(k)) + c(Z(k)) + \gamma\underline{H}^\theta(X(k+1)) \\ &- \delta\langle \omega, H^\theta\rangle.\end{aligned}$$

Formulation of batch algorithms is also straightforward.

IV. Rates of Convergence

We consider here the rate of convergence for the basic algorithm (13), whose solution is denoted θ_N. Subject to mild conditions we establish that $\theta_N \to \theta^*$ with probability one as $N \to \infty$, where θ^* solves (7); one assumption is that the solution is unique. It is more challenging to establish bounds on the mean-square error (MSE) $\mathsf{E}[\|\tilde{\theta}_N\|^2]$, with $\tilde{\theta}_N = \theta_N - \theta^*$. In fact, we have not been able to find a deterministic N_0 for which $\mathsf{E}[\|\tilde{\theta}_N\|^2] < \infty$ for $N \ge N_0$. We fix $r > 0$ satisfying $|\theta^*_i| < r$ for each i, and let θ^r_N denote the L_∞ projection of the solution of (13) to the region $\Theta_r = \{\theta \in \mathbb{R}^d : |\theta_i| \le r\,,\ 1 \le i \le d\}$. We set $\theta^r_N = 0$ if the convex program (13) is unbounded or infeasible.

While the sequence $\{\theta^r_N\}$ cannot be represented as the output of any recursive algorithm, key results from stochastic approximation theory would suggest that the MSE should decay as $O(1/N)$. We verify that this rate of convergence holds, and obtain finer results:

$$\begin{aligned} N\mathsf{E}[\tilde{\theta}^r_N(\tilde{\theta}^r_N)^{\intercal}] &= \Sigma_\theta + O\bigl(\tfrac{1}{\sqrt{N}}\bigr) \\ \sqrt{N}\tilde{\theta}^r_N &\xrightarrow{\text{dist}} N(0,\Sigma_\theta)\,, \qquad N \to \infty \end{aligned} \tag{19}$$

where the convergence is in distribution in the second limit, and $\tilde{\theta}^r_N = \theta^r_N - \theta^*$. The matrix $\Sigma_\theta \ge 0$ is identified in Prop. 4.2 after a few preliminary results.

For each $\theta \in \mathbb{R}^d$ and $\phi \in \Phi$ denote

$$\overline{g}_N(\theta,\phi) := -\frac{1}{N}\sum_{k=0}^{N-1} g_k(\theta,\phi)\,,$$

where $g_k(\theta,\phi) = -\mathcal{D}_{k+1}(\theta,\phi)\zeta_k$ (recall (8)).

Proposition 4.1: Any solution to the convex program (13) is also a solution to the linear program

$$\begin{aligned} &\max_{\theta}\ \ \langle \mu, Q^\theta\rangle \\ &\text{s.t.}\ \ \overline{g}^i_N(\theta,\phi) \le 0\,,\ \ 1 \le i \le d_+,\ \ \phi \in \Phi\,. \end{aligned} \tag{20}$$

Moreover, a vector $\theta \in \mathbb{R}^d$ is a BFS if and only if the following two properties hold: 1. there is a set $\mathcal{I}^\theta_+ = \{j_1,\dots,j_d\} \subset \{1,\dots,d_+\}$ such that

$$\begin{aligned} \overline{g}^i_N(\theta,\phi^\theta) &= 0\,,\ \ i \in \mathcal{I}^\theta_+ \\ &< 0\,,\ \ i \notin \mathcal{I}^\theta_+ \end{aligned}$$

where ϕ^θ is the Q^θ-greedy policy, and 2. There is no other Q^θ-greedy policy: if $\phi' \in \Phi$ satisfies

$$\phi'(x) \in \arg\min_u Q^\theta(x,u)\,, \qquad \textit{for all } x \in \mathsf{X},$$

then $\phi'(x) = \phi^\theta(x)$ for all x.

Proof. The proof of the LP characterization (20) is identical to the proof of Prop. 2.2. The characterization of a BFS is a consequence of the proof: By definition, if θ is a BFS then there are exactly d pairs $\{\phi^i, j_i : 1 \le i \le d\}$ for which the constraints are tight, meaning $\overline{g}^{j_i}_N(\theta,\phi^i) = 0$. We also have $\overline{g}^{j_i}_N(\theta,\phi^\theta) \ge \overline{g}^{j_i}_N(\theta,\phi^i) = 0$, so that by feasibility we must also have $\overline{g}^{j_i}_N(\theta,\phi^\theta) = 0$ for each i. Since there are exactly d active constraints, we must have $\phi^\theta = \phi^i$ for each i. □

Prop. 4.1 provides the ingredients required to establish convergence.

It is assumed that θ^* is unique, from which it follows that it is also a BFS. We let $\mathcal{I}_+ = \{j_1,\dots,j_d\}$ denote the set of d indices for which $\overline{g}^i_N(\theta,\phi^\theta) = 0$ when $\theta = \theta^*$ and $i \in \mathcal{I}_+$.

A d-dimensional stochastic process $\overline{W}_N$ is constructed as an average,

$$\overline{W}_N = \frac{1}{N}\sum_{k=1}^{N} W_k$$

in which the d-dimensional stochastic process $\{W_k\}$ is obtained in the following steps. First, construct a $d \times d$ matrix A^+_k, whose i,j element is given by

$$[A^+_k]_{i,j} := \bigl[-\psi_{(k-1)} + \gamma\underline{\psi}^{\theta^*}_{(k)}\bigr]_j \zeta^{j_i}_{k-1}$$

and let $\bar{A}^+ = \mathsf{E}_\varpi\bigl[A^+_k\bigr]$, where the expectation is in steady-state. Let β^+_k be the d-dimensional vector whose ith component is equal to $c_{(k-1)}\zeta^{j_i}_{k-1}$, whose steady state mean is the d-dimensional vector $\bar{\beta}^+ = \mathsf{E}_\varpi[\beta^+_k]$. We then take

$$W_k = \beta^+_k - \bar{\beta}^+ - [A^+_k - \bar{A}^+]\theta^*$$

This is analogous to the "disturbance sequence" that arises in variance analysis of standard Q-learning algorithms [5], [15].

Proposition 4.2: Suppose that the linear program (20) has a unique optimizer θ^*. Then, $\lim_{N\to\infty}\theta^r_N = \lim_{N\to\infty}\theta_N = \theta^*$ with probability one, and (19) holds.

The covariance matrix may be expressed

$$\begin{aligned} \Sigma_\theta &= [\bar{A}^+]^{-1}\Sigma_W([\bar{A}^+]^{-1})^{\intercal}\,, \\ \textit{with}\ \ \Sigma_W &= \lim_{N\to\infty} N\mathsf{E}[\overline{W}_N(\overline{W}_N)^{\intercal}] \end{aligned}$$

Invertibility of $\bar{A}^+$ is a consequence of the fact that θ^* is a BFS.

The form of the covariance Σ_θ is identical in form to the minimal covariance identified in the averaging theory of Polyak and Ruppert for stochastic approximation (see historical notes in [15, Ch. 8]). It is likely that it is also minimal here (the batch algorithms may satisfy a CLT, but the covariance matrix will dominate Σ_θ).

Proof of Prop. 4.2 (main concepts). The full proof follows from the more general Prop. A.1 in [12]. We present here the main ideas through a heuristic, which is conveniently explained for the more general setting of convex constraints[2].

Consider the minimization of a linear objective $v^{\intercal}\theta$ subject to convex constraints $\overline{g}_N(\theta) \le 0$, and let θ^* denote the minimizer for $N = \infty$: as in Prop. 4.2 there is a limiting function $\overline{g}$, and θ^* minimizes $v^{\intercal}\theta$ subject $\overline{g}(\theta) \le 0$. The heuristic is based on the convex program

$$\theta^*_N = \arg\min v^{\intercal}\theta \quad \text{subject to} \quad \overline{g}(\theta) \le b^*_N \tag{21}$$

with $b^*_N := \overline{g}(\theta^*) - \overline{g}_N(\theta^*)$. The remainder of the proof is based on the fact that $\mathsf{E}[\|b^*_N\|^2] = O(1/N)$, along with sensitivity theory for convex programming.

[2] Extension of variance theory to general convex constraints may be valuable when relaxing the assumption that U is finite, or when convex regularizers are introduced.

In analysis of (21) the definition of the set $\mathcal{I}_+$ remains the same, characterized in terms of the dual under mild conditions. Let $\lambda^* \in \mathbb{R}^{d_+}$ denote the Lagrange multiplier associated with the optimizer θ^*. If sensitivity is strictly positive for active constraints then $\mathcal{I}_+ = \{i : \lambda^*_i > 0\}$. □

Example To illustrate the CLT in a simple setting, consider a quadratic program on $\mathbb{R}^2$ with 10 constraints: $\theta \in \mathbb{R}^2$, $\overline{g} : \mathbb{R}^2 \to \mathbb{R}^{10}$, and $\overline{g}_N(\theta) = N^{-1}\sum_{k=0}^{N-1} g(\theta + \Delta_k)$, with $\{\Delta_k\}$ i.i.d., $N(0, I)$. The function $g : \mathbb{R}^2 \to \mathbb{R}^{10}$ is the quadratic $g(\vartheta) = (a^\intercal \vartheta) \odot (a^\intercal \vartheta) + b^\intercal \vartheta - 1$, where $a, b \in \mathbb{R}^{2\times 10}$, 1 is a vector of ones, and $\odot$ denotes element-wise multiplication.

Three quadratic programs were constructed by replacing the constraint in (21) with $\overline{g}_N(\theta) \le 0$, $\overline{g}(\theta) \le 0$, and $\overline{g}_N(\theta) \le b^*_N$. The solutions to each of these quadratic programs are denoted θ_N, θ^*, and θ^*_N respectively.

The values of a, b, and v were obtained by sampling independently from a normal distribution. The results that follow show typical results for one set of values.

The dual variable λ^* associated with θ^* was also obtained. Exactly two of ten inequality constraints were found to be tight, and exactly two values of λ^* were strictly positive.

The theoretical Gaussian density was compared to histograms obtained from repeated independent experiments. In each of 100 runs, the corresponding errors were recorded: $\sqrt{N}\tilde{\theta}^*_N := \sqrt{N}[\theta^*_N - \theta^*]$ and $\sqrt{N}\tilde{\theta}_N := \sqrt{N}[\theta_N - \theta^*]$. See [12] for histograms: CLT approximations are good even for $N = 10^4$, and nearly perfect for $N \ge 10^6$.

V. Application to Inventory Control

We survey here results from experiments on a classical inventory control problem focusing on two topics: (i) Stability and consistency of CvxQ and relative CvxQ; (ii) Comparison of convergence rates with Q-learning and CvxQ.[3]

Preliminaries Consider the inventory model with inventory level $X(k) \in \mathsf{X} = \mathbb{R}$ (a negative value indicating backlog), depletion rate $\beta > 0$, and stocking decision $U(k) \in \mathsf{U} = \{0, 1\}$. The MDP model has cost defined by parameters $c^+, c^- > 0$ and evolution equation,

$$\begin{aligned} X(k+1) &= X(k) - [\beta + W(k+1)] + U(k) \\ c(x,u) &= \max(c^+ x, -c^- x)\,, \quad x \in \mathbb{R}\,; \end{aligned} \tag{22a}$$

$\{W(k)\}$ is i.i.d. with zero mean and finite variance σ^2_W.

An optimal policy is of the threshold form:

$$\phi(x; \overline{r}) = \mathbf{1}\{x \le -\overline{r}\}. \tag{23}$$

For small $\beta > 0$ the optimal threshold is approximated by $\overline{r}^\dagger := \log(1 + c^+/c^-)/\varrho$ with ϱ the positive solution to $\sigma^2_W \varrho^2/2 - \beta\varrho - \gamma = 0$. See [17, Ch. 7] for background.

We set $\gamma = 0.99$, $\beta = 0.1$, $c^- = 1$, $c^+ = 10$ and $\sigma^2_W = 1$ in the experiments that follow, giving $\overline{r}^\dagger \approx 8.77$.

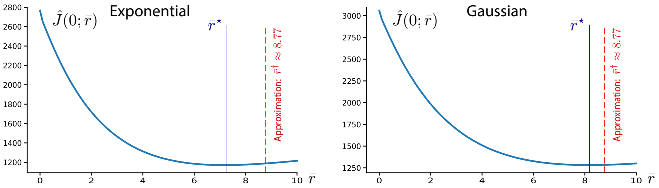


Fig. 1: Numerical Estimation of the Optimal Inventory Level.

Numerical Estimation of $\overline{r}^*$. To evaluate the results obtained using CvxQ we estimated the optimal threshold via Monte-Carlo, evaluating a range of 100 values of $\overline{r}$ evenly spaced in the interval $[0, 10]$. We used common randomness for the entire range of thresholds, meaning that we constructed a single i.i.d. family $\{W^i_k : 1 \le i \le \overline{N}, 1 \le k \le N\}$, with $\overline{N} = 2 \times 10^4$; in each experiment we chose initial condition $X^i(0) = 0$. For each threshold $\overline{r}$ we obtain the sample path $\{X^i(k; \overline{r}) : 0 \le k < N\}$, $N = 10^4$, and from this an estimate of the discounted total cost by truncating and averaging:

$$\widehat{J}(x; \overline{r}) = \frac{1}{\overline{N}} \sum_{i=1}^{\overline{N}} \sum_{k=0}^{N-1} \gamma^k c(X^i(k; \overline{r}))\,, \quad \overline{r} \in [0, 10]\,.$$

The plots in Fig. 1 show results for two choices of disturbance, $W_k \sim N(0, 1)$ and $W_k \sim \mathrm{Exp}(1) - 1$. In each case, the value of $\widehat{J}(0; \overline{r})$ at $\overline{r}^\dagger$ closely matches the empirical minimum $\overline{r}^*$.

A. Algorithm Comparisons

CvxQ, relative CvxQ, Q-learning, and relative Q-learning were applied to approximate the optimal policy under the same modeling assumptions used to obtain the plots in Fig. 1.

Details of Implementation Theory surveyed in [17, Ch. 7] tells us that the optimal value function $J^*(x) := \min_u Q^*(x, u)$ is convex, and is approximated by $c(x)/(1 - \gamma)$ for large x. This motivated the choice of 8-dimensional linear function class $\{Q^\theta = \theta^\intercal \psi : \theta \in \mathbb{R}^d\}$ using the continuously differentiable functions $\psi(z) = [\psi'(x)\mathbf{1}\{u = 0\}; \psi'(x)\mathbf{1}\{u = 1\}]^\intercal$ where $\psi'(x) = [\xi_1(x); \xi_2(x); x; 1]$, with $\xi_i(x) = (|x| + (e^{-\delta_i |x|} - 1))\mathbf{1}\{x \ge 0\}/\delta_i$. The values $\delta_1 = 0.5$, $\delta_2 = 0.1$, gave good results.

The following input sequence was used for training:

$$\begin{aligned} \phi(u|x) &:= P(U(k) = u | X(k) = x) \\ &= \varepsilon P_E(u) + (1 - \varepsilon)\mathbf{1}\{u = \phi(x)\} \end{aligned}$$

with $\varepsilon = 0.9$, P_E uniform on U, and ϕ a threshold policy that gave good performance.

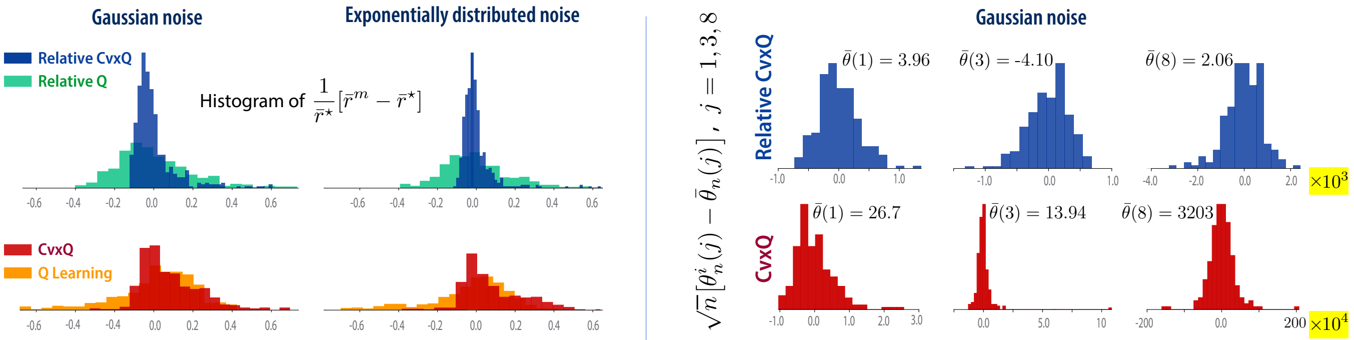
We chose $\{\zeta_k\}$ in (13), (18) to be the indicator functions:

$$\zeta^i_k = \mathbf{1}\{x^i \le X(k) \le x^{i+1}\}\,, \quad 1 \le i \le d_+$$

We found that choosing x^i to be evenly spaced in the range $[-28, 28]$ gave good performance. The results that follow applied this approach using $d_+ = 200$.

Several algorithms were tested: Q-learning using the recursion (10) with $\zeta_k = \psi_{(k)}$ and constant step size $\alpha_{k+1} = 10^{-3}$; relative Q-learning algorithm, defined by replacing $\mathcal{D}_{k+1}(\theta_k)$ with $\widehat{\mathcal{D}}_{k+1}(\theta_k)$ in (10); CvxQ (13) and relative

[3] The experiments go beyond the theory because the state space is not finite. We believe that numerical results are valuable for testing the boundaries of the theory. We know from experience that some disagree, so we respectfully ask the reviewers to accept our preferences.



(a) Histograms of relative error for target inventory estimates

(b) Variance is reduced by more than two orders of magnitude using relative CvxQ

Fig. 2: Performance Comparison between CvxQ, CvxRQ Q-learning and relative Q-learning.

CvxQ (18) using the convex solver from the Python library CVXPY.

Independent experiments were conducted with common disturbance $\{W(k)\}$ in each run (using the model (22a)). In each of $M = 100$ independent runs with time horizon $N = 10^4$, the following data was collected: the final parameter estimate θ^m, and the estimate $\bar{r}^m$ of the optimal threshold, defined as the minimal solution $Q^{\theta^m}(x, 0) \le Q^{\theta^m}(x, 1)$ for all $x \ge \bar{r}^m$. Selected results are illustrated in Fig. 2. Shown on the left hand side are histograms of the relative error $\{[\bar{r}^m - \bar{r}^*]/\bar{r}^* : 1 \le m \le M\}$. We see that relative CvxQ offers the lowest variance in these experiments.

The right hand side of Fig. 2 shows histograms of components of the scaled parameter error, $\sqrt{n}[\theta^*_m(i) - \bar{\theta}(i)]$ for several i, with $\bar{\theta}$ the average of θ^*_m. Only results for the CvxQ algorithms are shown. The CLT appears to hold, with relative CvxQ giving much lower variance.

VI. Conclusions

This paper lays out new approaches to reinforcement learning design and analysis. Important gaps include the augmentation of CvxQ to impose bounds on performance of resulting policies, techniques for analysis in non-stationary settings (motivated by the need for more efficient exploration), and development of algorithms for more advanced function approximation architectures (e.g., neural networks and RKHS). It is important to improve the numerical performance of CvxQ, build bridges with logistic Q-learning and recent approaches, such as the interacting particle approach of [8], and find ways to make comparisons in terms of both training efficiency and policy performance.

References

[1] L. Baird. Residual algorithms: Reinforcement learning with function approximation. In A. Prieditis and S. Russell, editors, *Proc. Machine Learning*, pages 30–37. Morgan Kaufmann, San Francisco (CA), 1995.

[2] J. Bas Serrano, S. Curi, A. Krause, and G. Neu. Logistic Q-learning. In A. Banerjee and K. Fukumizu, editors, *Proc. of The Intl. Conference on Artificial Intelligence and Statistics*, volume 130, pages 3610–3618, 13–15 Apr 2021.

[3] D. P. De Farias and B. Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of operations research*, 29(3):462–478, 2004.

[4] D. P. de Farias and B. Van Roy. A cost-shaping linear program for average-cost approximate dynamic programming with performance guarantees. *Math. Oper. Res.*, 31(3):597–620, 2006.

[5] A. M. Devraj and S. P. Meyn. Zap Q-learning. In *Proc. of the Intl. Conference on Neural Information Processing Systems*, pages 2232–2241, 2017.

[6] A. M. Devraj and S. P. Meyn. Q-learning with uniformly bounded variance. *IEEE Trans. on Automatic Control*, 67(11):5948–5963, 2022.

[7] D. Ding and M. R. Jovanović. Global exponential stability of primal-dual gradient flow dynamics based on the proximal augmented Lagrangian. In *Proc. of the American Control Conf.*, pages 3414–3419. IEEE, 2019.

[8] A. A. Joshi, A. Taghvaei, P. G. Mehta, and S. P. Meyn. Controlled interacting particle algorithms for simulation-based reinforcement learning. *Systems & Control Letters Control Letters*, 170:1–15, 2022.

[9] C. Lakshminarayanan, S. Bhatnagar, and C. Szepesvári. A linearly relaxed approximate linear program for Markov decision processes. *IEEE Transactions on Automatic control*, 63(4):1185–1191, 2017.

[10] F. Lu, P. G. Mehta, S. P. Meyn, and G. Neu. Convex Q-learning. In *American Control Conf.*, pages 4749–4756. IEEE, 2021.

[11] F. Lu, P. G. Mehta, S. P. Meyn, and G. Neu. Convex analytic theory for convex Q-learning. In *IEEE Conference on Decision and Control*, pages 4065–4071, Dec 2022.

[12] F. Lu and S. Meyn. Convex Q-learning in a stochastic environment: Extended version. *arXiv:2309.05105*, 2023.

[13] A. S. Manne. Linear programming and sequential decisions. *Management Sci.*, 6(3):259–267, 1960.

[14] P. G. Mehta and S. P. Meyn. Q-learning and Pontryagin's minimum principle. In *Proc. of the Conf. on Dec. and Control*, pages 3598–3605, Dec. 2009.

[15] S. Meyn. *Control Systems and Reinforcement Learning*. Cambridge University Press, Cambridge, 2022.

[16] S. Meyn. Stability of Q-learning through design and optimism. *arXiv 2307.02632*, 2023.

[17] S. P. Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2007. Pre-publication edition available online.

[18] G. Neu and N. Okolo. Efficient global planning in large MDPs via stochastic primal-dual optimization. In *International Conference on Algorithmic Learning Theory*, pages 1101–1123, 2023.

[19] M. Petrik and S. Zilberstein. Constraint relaxation in approximate linear programs. In *International Conference on Machine Learning*, pages 809–816, 2009.

[20] G. Qu and N. Li. On the exponential stability of primal-dual gradient dynamics. *Control Systems Letters*, 3(1):43–48, 2018.

[21] P. J. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of mathematical analysis and applications*, 110(2):568–582, 1985.

[22] C. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[23] J. N. Tsitsiklis and B. Van Roy. Feature-based methods for large scale dynamic programming. *Mach. Learn.*, 22(1-3):59–94, 1996.

[24] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, Cambridge, UK, 1989.