Structured Policy Iteration for Linear Quadratic Regulator

Youngsuk Park [1] **Ryan A. Rossi** [2] **Zheng Wen** [3] **Gang Wu** [2] **Handong Zhao** [2]

Abstract

Linear quadratic regulator (LQR) is one of the most popular frameworks to tackle continuous Markov decision process tasks. With its fundamental theory and tractable optimal policy, LQR has been revisited and analyzed in recent years, in terms of reinforcement learning scenarios such as the model-free or model-based setting. In this paper, we introduce the *Structured Policy Iteration* (S-PI) for LQR, a method capable of deriving a structured linear policy. Such a structured policy with (block) sparsity or low-rank can have significant advantages over the standard LQR policy: more interpretable, memory-efficient, and well-suited for the distributed setting. In order to derive such a policy, we first cast a regularized LQR problem when the model is known. Then, our Structured Policy Iteration (S-PI) algorithm, which takes a policy evaluation step and a policy improvement step in an iterative manner, can solve this regularized LQR efficiently. We further extend the S-PI algorithm to the model-free setting where a smoothing procedure is adopted to estimate the gradient. In both the known-model and model-free setting, we prove convergence analysis under the proper choice of parameters. Finally, the experiments demonstrate the advantages of S-PI in terms of balancing the LQR performance and level of structure by varying the weight parameter.

1. Introduction

Stochastic control for the class of linear quadratic regulator (LQR) has been applied in a wide variety of fields including supply-chain optimization, advertising, dynamic resource allocation, and optimal control (Sarimveis et al., 2008; Nerlove & Arrow, 1962; Elmaghraby, 1993; Anderson & Moore, 2007) spanning several decades.

This stochastic control has led to a wide class of fundamental machinery along the way, across theoretical analysis as well as tractable algorithms, where the model of transition dynamic and cost function are known. On the other hand, under the uncertain model of transition dynamics, reinforcement learning (RL) and data-driven approaches have achieved a great empirical success in recent years, from simulated game scenarios (Mnih et al., 2015; Silver et al., 2016) to robot manipulation (Tassa et al., 2012; Al Borno et al., 2012; Kumar et al., 2016). In recent years, LQR in discrete time domain in particular, has been revisited and analyzed under model uncertainty, not only in theoretical perspective like regret bound or sample complexity (Ibrahimi et al., 2012; Fazel et al., 2018; Recht, 2019; Mania et al., 2019), but also toward new real-world applications (Lewis et al., 2012; Lazic et al., 2018; Park et al., 2019).

Despite the importance and success of the standard LQR, discrete time LQR with a structured policy has been less explored in theoretical and practical perspective under both the known and unknown model settings, while such a policy may have a number of significant advantages over the standard LQR policy: interpretability, memory-efficiency, and is more suitable for the distributed setting. In this work, we describe a methodology for learning a structured policy for LQR along with theoretical analysis of it.

Summary of contributions.

- We formulate the regularized LQR problem for discrete time system that is able to capture a structured linear policy (in Section 2.1).
- To solve the regularized LQR problem when the model is known, we develop the Structured Policy Iteration (S-PI) algorithm that consists of a policy evaluation and policy improvement step (in Section 2.2).
- We further extend S-PI to the model-free setting, utilizing a gradient estimate from a smoothing procedure (in Section 3.1).
- We prove the linear convergence of S-PI to its stationary point under a proper choice of parameters in both the known-model (in Section 2.3) and model-free settings (in Section 3.2).
- We examine the properties of the S-PI algorithm and

[1]Stanford University [2]Adobe Research [3]Google DeepMind. Correspondence to: Youngsuk Park <youngsuk@stanford.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

demonstrate its capability of balancing the LQR performance and level of structure by varying the weight parameter (in Section 4).

1.1. Preliminary and background

Notations. For a symmetric matrix $Q \in \mathbf{R}^{n\times n}$, we denote $Q \succeq 0$ and $Q \succ 0$ as a positive semidefinite and positive definite matrix respectively. For a matrix $A \in \mathbf{R}^{n\times n}$, we denote $\rho(A)$ as its spectral radius, i.e., the largest magnitude among eigenvalues of A. For a matrix $K \in \mathbf{R}^{n\times m}$, $\sigma_1(K) \geq \sigma_2(K) \geq \cdots \geq \sigma_{\min(n,m)}(K)$ are defined as its ordered singular values where $\sigma_{\min}(K)$ is the smallest one. $\|A\|$ denotes the ℓ_2 matrix norm, i.e., $\max_{\|x\|_2=1} \|Ax\|_2$. We also denote Frobenius norm $\|A\|_F = \sqrt{\sum_{i,j} A_{i,j}^2}$ induced by Frobenius inner product $\langle A, B\rangle_F = \mathbf{Tr}(A^T B)$ where A and B are matrices of the same size. A ball with the radius r and its surface are denoted as $\mathbb{B}_r$ and $\mathbb{S}_r$ respectively.

1.1.1. Optimal control for infinite time horizon.

We formally define the class of problems we target here. Consider a Markov Decision Process (MDP) in continuous space where $x_t \in \mathcal{X} \subset \mathbf{R}^n$, $u_t \in \mathcal{U} \subset \mathbf{R}^m$, $w_t \in \mathcal{W} \subset \mathbf{R}^n$ denote a state, an action, and some random disturbance at time t, respectively. Further, let $g : \mathcal{X} \times \mathcal{U} \times \mathcal{X} \to \mathbf{R}$ be a stage-cost function, and $f : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$ denote a transition dynamic when action u_t is taken at state x_t with disturbance w_t[1].

Our goal is to find the optimal stationary policy $\pi : \mathcal{X} \to \mathcal{U}$ that solves

$$\begin{aligned} \underset{\pi}{\text{minimize}} \quad & \sum_{t=0}^{\infty} \gamma^t \mathbf{E}[g(x_t, u_t, x_{t+1})] \\ \text{subject to} \quad & x_{t+1} = f(x_t, u_t, w_t),\ x_0 \sim \mathcal{D} \end{aligned} \tag{1}$$

where $\gamma \in (0, 1]$ is the discounted factor, $\mathcal{D}$ is the distribution over the initial state x_0.

1.1.2. Background on classic LQR

Infinite horizon (discounted) LQR is a subclass of the problem (1) where the stage cost is quadratic and the dynamic is linear, $\mathcal{X} = \mathbf{R}^n, \mathcal{U} = \mathbf{R}^m$, $\gamma = 1$, and $w_t = 0$, i.e.,

$$\begin{aligned} \underset{\pi}{\text{minimize}} \quad & \mathbf{E}\left(\sum_{t=0}^{\infty} x_t^T Q x_t + u_t^T R u_t\right) \\ \text{subject to} \quad & x_{t+1} = Ax_t + Bu_t, \\ & u_t = \pi(x_t),\ x_0 \sim \mathcal{D}, \end{aligned} \tag{2}$$

where $A \in \mathbf{R}^{n\times n}, B \in \mathbf{R}^{n\times m}, Q \succeq 0$, and $R \succ 0$.

Optimal policy and value function. The optimal policy (or control gain) and value function (optimal cost-to-go) are known to be linear and convex quadratic on state respectively,

$$\pi^\star(x) = Kx, \quad V^\star(x) = x^T P x$$

where

$$\begin{aligned} P &= A^T P A + Q - A^T P B (B^T P B + R)^{-1} B^T P A, \\ K &= -(B^T P B + R)^{-1} B^T P A. \end{aligned} \tag{3}$$

Note that Eq. (3) is known as the discrete time Algebraic Riccati equation (DARE). Here, we assume (A, B) is controllable[2], then the solution is unique and can be efficiently computed via the Riccati recursion or some alternatives (Hewer, 1971; Laub, 1979; Lancaster & Rodman, 1995; Balakrishnan & Vandenberghe, 2003).

Variants and extensions on LQR. There are several variants of LQR including noisy, finite horizon, time-varying, and trajectory LQR. Including linear constraints, jumping and random model are regarded as extended LQR. In these cases, some pathologies such as infinite cost may occur (Bertsekas et al., 1995) but we do not focus on such cases.

1.1.3. Zeroth order optimization.

Zeroth order optimization is the framework optimizing a function $f(x)$, only accessing its function values (Conn et al., 2009; Nesterov & Spokoiny, 2017). It defines its perturbed function $f_{\sigma^2}(x) = \mathbf{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)} f(x+\epsilon)$, which is close to the original function $f(x)$ for small enough perturbation σ. And Gaussian smoothing provides its gradient form $\nabla f_{\sigma^2}(x) = \frac{1}{\sigma^2}\mathbf{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I)} f(x+\epsilon)\epsilon$. Similarly, from (Flaxman et al., 2004; Fazel et al., 2018), we can do a smoothing procedure over a ball with the radius r in $\mathbf{R}^n$. The perturbed function and its gradient can be defined as $f_r(x) = \mathbf{E}_{\epsilon\sim\mathbb{B}_r} f(x+\epsilon)$ and $\nabla f_r(x) = \frac{n}{r^2}\mathbf{E}_{\epsilon\sim\mathbb{S}_r} f(x+\epsilon)\epsilon$ where ϵ is sampled uniformly random from a ball $\mathbb{B}_r$ and its surface $\mathbb{S}_r$ respectively. Note that these simple (expected) forms allow Monte Carlo simulations to get unbiased estimates of the gradient based on function values, without explicit computation of the gradient.

1.2. Related work

The general method for solving these problems is dynamic programming (DP) (Bellman et al., 1954; Bertsekas et al., 1995). To overcome the issues of DP such as intractability and computational cost, the common alternatives are

[1] often denoted as w_{t+1} since it is fully revealed at time $t+1$.

[2] The linear system is controllable if the matrix $[B, AB, \cdots, A^{n-1}B]$ has full column rank.

approximated dynamic programming (ADP) (Bertsekas & Tsitsiklis, 1996; Bertsekas & Shreve, 2004; Powell, 2007; De Farias & Van Roy, 2003; O'Donoghue et al., 2011) or Reinforcement Learning (RL) (Sutton et al., 1998) including policy gradient (Kakade, 2002; Schulman et al., 2017) or Q-learning based methods (Watkins & Dayan, 1992; Mnih et al., 2015).

LQR. LQR is a classical problem in control that is able to capture problems in continuous state and action space, pioneered by Kalman in the late 1950's (Kalman, 1964). Since then, many variations have been suggested and solved such as jump LQR, random LQR, (averaged) infinite horizon objective, etc (Florentin, 1961; Costa et al., 2006; Wonham, 1970). When the model is (assumed to be) known, Arithmetic Riccati Equation (ARE) (Hewer, 1971; Laub, 1979) can be used to efficiently compute the optimal value function and policy for generic LQR. Alternatively, we can use eigendecomposition (Lancaster & Rodman, 1995) or transform it into a semidefinite program (SDP) (Balakrishnan & Vandenberghe, 2003).

LQR under model uncertainty. When the model is unknown, one class is model-based approaches where the transition dynamics is attempted to be estimated. For LQR, in particular, (Abbasi-Yadkori & Szepesvári, 2011; Ibrahimi et al., 2012) developed online algorithms with regret bounds where linear dynamic (and cost function) are learned and sampled from some confidence set, but without any computational demonstrations. Another line of work is to utilize robust control with system identification (Dean et al., 2017; Recht, 2019) with sample complexity analysis (Tu & Recht, 2017). Certainty Equivalent Control for LQR is also analyzed with suboptimality gap (Mania et al., 2019). The other class is model-free approaches where policy is directly learned without estimating dynamics. Regarding discrete time LQR, in particular, Q-learning (Bradtke et al., 1994) and policy gradient (Fazel et al., 2018) were developed together with lots of mathematically machinery therein. but with little empirical demonstrations.

LQR with structured policy. For continuous time LQR, many work in control literature (Wang & Davison, 1973; Sandell et al., 1978; Lau et al., 1972) has been studied for sparse and decentralized feedback gain, supported by theoretical demonstrations. Recently, (Wytock & Kolter, 2013; Lin et al., 2012) developed fast algorithms for a large-scale system that induce the sparse feedback gain utilizing Alternating Direction Method of Multiplier (ADMM), Iterative Shrinkage Thresholding Algorithm (ISTA), Newton method, etc. And, most of these only consider sparse feedback gain under continuous time LQR.

However, to the best of our knowledge, there are few algorithms for discrete time LQR, which take various structured (linear) policies such as sparse, low-rank, or proximity to some reference policy into account. Furthermore, regarding learning such a structured policy, there is no theoretical work that demonstrates a computational complexity, sample complexity, or the dependency of stable stepsize on algorithm parameters either, even in known-model setting (and model-free setting).

2. Structured Policy for LQR

2.1. Problem statement: regularized LQR

From standard LQR in Eq. (2), we restrict policy to linear class, i.e., $u_t = Kx_t$, and add a regularizer on the policy to induce the policy structure. We formally state a *regularized LQR* problem as

$$\begin{array}{ll} \underset{K}{\text{minimize}} & \overbrace{\mathbf{E}\left(\sum_{t=0}^{\infty} x_t^T Q x_t + u_t^T R u_t\right)}^{f(K)} + \lambda r(K) \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t, \\ & u_t = Kx_t, \quad x_0 \sim \mathcal{D}, \end{array} \tag{4}$$

for a nonnegative parameter $\lambda \geq 0$. Here $f(K)$ is the (averaged) cost-to-go under policy K, and $r : \mathbf{R}^{n\times m} \to \mathbf{R}$ is a nonnegative convex regularizer inducing the structure of policy K.

Regularizer. Different regularizers induce different types of structures on the policy K. We consider lasso $r(K) = \|K\|_1 = \sum_{i,j} |K_{i,j}|$, group lasso $r(K) = \|K\|_{\mathcal{G},2} = \sum_{g\in\mathcal{G}} \|K_g\|_2$ where $K_g \in \mathbf{R}^{|g|}$ is the vector consisting of a index set g, and nuclear-norm $r(K) = \|K\|_* = \sum_i \sigma_i(K)$ where $\sigma_i(K)$ is the ith largest singular value of K. These induces sparse, block sparse, and low-rank structure respectively. For a given reference policy $K^{\mathrm{ref}} \in \mathbf{R}^{n\times m}$, we can similarly consider $\|K - K^{\mathrm{ref}}\|_1$, $\|K - K^{\mathrm{ref}}\|_{\mathcal{G},2}$, and $\|K - K^{\mathrm{ref}}\|_*$, penalizing the proximity (in different metric) to the reference policy K^{ref}.

Non-convexity. For a standard (unregularized) LQR, the objective function $f(K)$ is known to be not convex, quasi-convex, nor star-convex, but to be gradient dominant. Therefore, all the stationary points are optimal as long as $\mathbf{E}[x_0 x_0^T] \succ 0$ (see Lemma 2,3, in (Fazel et al., 2018)). However, in regularized LQR, all the stationary points of Eq. (4) may not be optimal under the existence of multiple stationary points. (See the supplementary material for the detail.)

2.2. Structured Policy Iteration (S-PI)

Eq. (4) can be simplified into

$$\text{minimize} \quad F(K) := f(K) + \lambda r(K). \tag{5}$$

Here $f(K) = \mathbf{Tr}(\Sigma_0 P)$ where $\Sigma_0 = \mathbf{E}[x_0 x_0^T]$ is the covariance matrix of initial state and P is the quadratic value matrix satisfying the following Lyapunov equation

$$(A+BK)^T P (A+BK) - P + Q + K^T R K = 0. \quad (6)$$

We introduce the *Stuctured Policy Iteration* (S-PI) in Algorithm 1 to solve Eq. (5). The S-PI algorithm consists of two parts: (1) policy evaluation and (2) policy improvement. In the policy evaluation part, we solve Lyapunov equations to compute the quadratic value matrix P and covariance matrix Σ. In the policy improvement part, we try to improve the policy while encouraging some structure, via the proximal gradient method with proper choice of an initial stepsize and a backtracking linesearch strategy.

Algorithm 1 **S**tuctured **P**olicy **I**teration (**S-PI**)

given initial stable policy K^0 and initial state covariance matrix $\Sigma_0 = \mathbf{E}[x_0 x_0^T]$, linesearch factor $\beta < 1$.
repeat
  (1) Policy (and covariance) evaluation:
  compute (P^i, Σ^i) satisfying Lyapunov equations

$$(A+BK^i)^T P^i (A+BK^i) - P^i + Q + (K^i)^T R K^i = 0,$$
$$(A+BK^i)\Sigma^i (A+BK^i)^T - \Sigma^i + \Sigma_0 = 0.$$

  return (P^i, Σ^i)
  (2) Policy improvement:
  initial stepsize $\eta_i = \mathcal{O}(1/\lambda)$.
  compute gradient at K

$$\nabla_K f(K^i) = 2\left(\left(R + B^T P^i B\right) K^i + B^T P^i A\right) \Sigma^i$$

  repeat
    $\eta_i := \beta \eta_i$.
    $K^{i+1} \leftarrow \mathrm{ProxGrad}(\nabla f(K^i), \eta_i, r, \lambda)$ (in Alg. 2).
  until linesearch (7) criterion is satisfied.
  return next iterate K^{i+1}.
until stopping criterion $\|K^{i+1} - K^i\| \le \epsilon$ is satisfied.

Algorithm 2 Subroutine: $\mathrm{ProxGrad}(\nabla f(K), \eta, r, \lambda)$

Input gradient oracle $\nabla f(K)$, stepsize η, and regularization r and its parameter λ
take gradient step

$$G \leftarrow K - \eta \nabla_K f(K)$$

take proximal step

$$K^+ \leftarrow \mathbf{prox}_{r(\cdot),\lambda\eta}(G) := \operatorname*{argmin}_K r(K) + \frac{1}{2\lambda\eta}\|K - G\|_F^2$$

return K^+

Sensitivity to initial and fixed stepsize choice. Note that we use the initial stepsize $\eta = \mathcal{O}(1/\lambda)$ as a rule of thumb whereas the typical initial stepsize (Barzilai & Borwein, 1988; Wright et al., 2009; Park et al., 2020) does not depend on the regularization parameter λ. This stepsize choice is motivated by theoretical analysis (see Lemma 3 in Section 2.3) as well as empirical demonstration (see Fig. 1 in Section 4). This order of stepsize automatically scales well when experimenting over various λs, alleviating iteration counts and leading to a faster algorithm in practice. It turns out that proximal gradient step is very sensitive to stepsizes, often leading to an unstable policy K with $\rho(A+BK) \ge 1$ or requiring a large number of iteration counts to converge. Therefore, we utilize linesearch over fixed stepsize choice.

Linesearch. We adopt a backtracking linesearch (See (Parikh et al., 2014)). Given η_i, K^i, $\nabla f(K^i)$, and the potential next iterate K^{i+1}, it check if the following criterion (the stability and the decrease of the objective) is satisfied,

$$f(K^{i+1}) \le f(K^i) - \eta_i \mathbf{Tr}(\nabla f(K^i)^T G_{\eta_i}(K^i)) + \frac{\eta_i}{2}\|G_{\eta_i}(K^i)\|_F^2, \quad (7)$$
$$\rho(A + BK^{i+1}) < 1,$$

where $G_{\eta_i}(K) = \frac{1}{\eta_i}(K - \mathbf{prox}_{r,\lambda\eta_i}(K - \eta_i \nabla f(K)))$ and $\rho(\cdot)$ is the spectral radius. Otherwise, it shrinks the stepsize η_i by a factor of $\beta < 1$ and check it iteratively until Eq. (7) is satisfied.

Stabilizing sequence of policy. We can start with a stable policy K^0, meaning $\rho(A+BK^0) < 1$. For example, under the standard assumptions on A, B, Q, R, Riccati recursion provides a stable policy, the solution of standard LQR in Eq. (2). Then, satisfying the linesearch criterion in Eq. (7) subsequently, the rest of the policies $\{K^i\}$ are a stabilizing sequence.

Computational cost. The major cost incurs when solving the Lyapunov equations in the policy (and covariance) evaluation step. Note that if $A + BK$ is stable, so does $(A+BK)^T$ since they share the same eigenvalues. Under the stability, each Lyapunov equation in Eq. (7) has a unique solution with the computational cost $O(n^3)$ (Jaimoukha & Kasenally, 1994; Li & White, 2002). Additionally, we can solve a sequence of Lyapunov equations with less cost, via using iterative methods with adopting the previous one (warm-start) or approximated one.

2.2.1. Regularizer and Proximal Operator

For various regularizers mentioned in Section 2.1, each has the closed-form solution for its proximal operator (Rockafellar, 1976; Parikh et al., 2014; Park et al., 2020). Here we only include a few representative examples and refer to the supplementary material for more examples.

Lemma 1 (Examples of proximal operator)**.**

1. ***Lasso.*** *For* $r(K) = \|K\|_1$,

$$\left(\mathbf{prox}_{r,\lambda\eta}(K)\right)_{i,j} = \mathrm{sign}(K_{i,j})(|K_{i,j}| - \lambda\eta)_+.$$

And we denote $\mathbf{prox}_{r,\lambda\eta}(K) := S_{\lambda\eta}(K)$ *as a soft-thresholding operator.*

2. ***Nuclear norm.*** *For* $r(K) = \|K\|_*$,

$$\mathbf{prox}_{r,\lambda\eta}(K) = U\mathbf{diag}(S_{\lambda\eta}(\sigma))V^T.$$

where $K = U\mathbf{diag}(\sigma)V^T$ *is singular value decomposition with singular values* $\sigma \in \mathbf{R}^{\min(n,m)}$.

3. ***Proximity to*** K^{ref}***.*** *For* $r(K) = \left\|K - K^{\mathrm{ref}}\right\|_F^2$,

$$\mathbf{prox}_{r,\lambda\eta}(K) = \frac{2\lambda\eta K^{\mathrm{ref}} + K}{2\lambda\eta + 1}.$$

2.3. Convergence analysis of S-PI

Recall that $\|\cdot\|$, $\|\cdot\|_F$, $\sigma_{\min}(\cdot)$ is ℓ_2 matrix norm, Frobenius norm, and smallest singular value, as mentioned in Section 1.1. First, we start with (local) smoothness and strong convexity around a stable policy. Here we regard a policy K is stable if $\rho(A + BK) < 1$.

Assumptions. $\rho(A + BK^0) < 1$, $\Sigma_0 = \mathbf{E}x_0x_0^T \succ 0$, $\left\|K^0 - K^\star\right\| \le \Delta$, and $\left\|K^{\mathrm{ref}} - K^\star\right\| \le \Delta$.

Lemma 2. *For stable* K, $f(K)$ *is smooth (in local) with*

$$L_K = 4\|\Sigma(K)\|\|R + B^TP(K)B\| < \infty,$$

within local ball around $K \in \mathcal{B}(K; \rho_K)$ *where the radius* ρ_K *is*

$$\rho_K = \frac{\sigma_{\min}(\Sigma_0)}{4\|\Sigma(K)\|\left(\|A + BK\| + 1\right)\|B\|} > 0.$$

And $f(K)$ *is strongly convex (in local) with*

$$m = \sigma_{\min}(\Sigma_0)\sigma_{\min}(R) > 0.$$

Then, we provide a proper stepsize that guarantees one iteration of proximal gradient step is still inside of the stable and (local) smooth ball $\mathcal{B}(K; \rho_K)$.

Lemma 3. *Let* $K^+ = \mathbf{prox}_{r,\lambda\eta}(K - \eta\nabla f(K))$. *Then*

$$K^+ \in \mathcal{B}(K; \rho_K)$$

holds for any $0 < \eta < \eta_K^{\lambda,r}$ *where* $\eta_K^{\lambda,r}$ *is given as*

$$\eta_K^{\lambda,r} = \begin{cases} \frac{\rho_K}{\|\nabla f(K)\| + \lambda nm} & r(K) = \|K\|_1 \\ \frac{\rho_K}{\|\nabla f(K)\| + \lambda \min(n,m)} & r(K) = \|K\|_* \\ \frac{\rho_K}{2\|\nabla f(K)\| + 2\lambda\|K - K^{\mathrm{ref}}\|} & r(K) = \|K - K^{\mathrm{ref}}\|_F^2 \end{cases}.$$

Next proposition describes that next iterate policy has the decrease in function value and is stable under sufficiently small stepsize.

Proposition 1. *Assume* $A + BK$ *is stable. For any stepsize* $0 < \eta \le \min(\frac{1}{L_K}, \eta_K^{\lambda,r})$ *and next iterate* $K^+ = \mathbf{prox}_{r,\eta\lambda}(K - \eta\nabla f(K))$,

$$\rho(A + BK^+) < 1 \tag{8}$$

$$F(K^+) \le F(K) - \frac{1}{2\eta}\|K - K^+\|_F^2 \tag{9}$$

holds.

We also derive the bound on stepsize in Lemma 3, not dependent on iteration numbers.

Lemma 4. *Assume that* $\{K^i\}_{i=0,\dots}$ *is a stabilizing sequence and associated* $\{f(K^i)\}_{i=0,\dots}$ *and* $\{\|K^i - K^\star\|_F\}_{i=0,\dots}$ *are decreasing sequences. Then, Lemma 3 holds for*

$$\eta^{\lambda,r} = \begin{cases} \frac{\rho^L}{\rho^f + \lambda nm} & r(K) = \|K\|_1 \\ \frac{\rho^L}{\rho^f + \lambda \min(n,m)} & r(K) = \|K\|_* \\ \frac{\rho^L}{2\rho^f + 4\lambda\Delta} & r(K) = \|K - K^{\mathrm{ref}}\|_F^2 \end{cases}. \tag{10}$$

where

$$\rho^f = 2\frac{F(K^0)}{\sigma_{\min}(Q)}\Bigg(\left\|B^T\right\|\frac{F(K^0)}{\sigma_{\min}(\Sigma_0)}\|A\| + \left(\|R\| + \left\|B^T\right\|\frac{F(K^0)}{\sigma(\Sigma_0)}\|B\|\right)(\Delta + \|K^\star\|)\Bigg),$$

$$\rho^L = \frac{\sigma_{\min}(\Sigma_0)^2}{8F(K^0)\|B\|}.$$

Now we prove that the S-PI in Algorithm 1 converges to the stationary point linearly.

Theorem 1. K^i *from Algorithm 1 converges to the stationary point* $K^\star$. *Moreover, it converges linearly, i.e., after* N *iterations,*

$$\left\|K^N - K^\star\right\|_F^2 \le \left(1 - \frac{1}{\kappa}\right)^N \left\|K^0 - K^\star\right\|_F^2.$$

Here, $\kappa = 1/\left(\eta_{\min}\sigma_{\min}(\Sigma_0)\sigma_{\min}(R)\right) > 1$ *where*

$$\eta_{\min} = h_\eta\Bigg(\sigma_{\min}(\Sigma_0), \sigma_{\min}(Q), \frac{1}{\lambda}, \frac{1}{\|A\|}, \frac{1}{\|B\|}, \frac{1}{\|R\|}, \frac{1}{\Delta}, \frac{1}{F(K^0)}\Bigg), \tag{11}$$

for some non-decreasing function h_η *on each argument.*

Note that (the global bound on) stepsize $\eta_{\min}$ is inversely proportional to λ, which motivates the initial stepsize in linesearch.

Corollary 1. *Let $K^\star$ be the stationary point from algorithm 1. Then, after N iterations*

$$N \geq 2\kappa \log\left(\frac{\left\|K^0 - K^\star\right\|_F}{\epsilon}\right),$$

$$\left\|K^N - K^\star\right\|_F \leq \epsilon$$

holds where $\kappa = 1/\left(\eta_{\min}\sigma_{\min}(\Sigma_0)\sigma_{\min}(R)\right)) > 1$ and $\eta_{\min}$ in Eq. (11).

3. Toward a Model-Free Framework

In this section, we consider the scenario where the cost function and transition dynamic are unknown. Specifically in model-free setting, policy is directly learned from (trajectory) data without explicitly estimating the cost or transition model. In this section, we extend our Structured Policy Iteration (S-PI) to the model-free setting and prove its convergence.

3.1. Model-free Structured Policy Iteration (S-PI)

Note that, in model-free setting, model parameters A, B, Q and R cannot be directly accessed, which hinders the direct computation of P, Σ, and $\nabla f(K)$ accordingly. Instead, we adopt a smoothing procedure to estimate the gradient based on samples.

Model-free S-PI in Algorithm 3 consists of two steps: (1) policy evaluation step and (2) policy improvement step. In (perturbed) policy evaluation step, perturbation U^j is uniformly drawn from the surface of the ball with radius r, $\mathbb{S}_r \subset \mathbf{R}^{n\times m}$. These data are used to estimate the gradient via a smoothing procedure for the policy improvement step. With this approximate gradient, proximal gradient subroutine tries to decrease the objective while inducing the structure of policy. Comparing to (known-model) S-PI in Algorithm 1, one important difference is its usage of a fixed stepsize η, rather than an adaptive stepsize from a backtracking linesearch that requires to access function value $f(K) = \mathbf{Tr}(\Sigma_0 P)$ explicitly.

3.2. Convergence analysis of model-free S-PI

The outline of proof is as following: We first claim that for proper parameters (perturbation, horizon number, numbers of trajectory), the gradient estimate from the smoothing procedure is close enough to actual gradient with high probability. Next we demonstrate that approximate proximal gradient still converges linearly with high probability.

Theorem 2. *Suppose $F(K^0)$ is finite, $\Sigma_0 \succ 0$, and that $x_0 \sim \mathcal{D}$ has norm bounded by D almost surly. Suppose the parameters in Algorithm 3 are chosen from*

$$(N_{\text{traj}}, H, 1/r) = h\left(n, \frac{1}{\epsilon}, \frac{1}{\sigma_{\min}(\Sigma_0)\sigma_{\min}(R)}, \frac{D^2}{\sigma_{\min}(\Sigma_0)}\right).$$

Algorithm 3 Model-free **S**tuctured **P**olicy **I**teration (Model-free **S-PI**)

given initial stable policy K^0, number of trajectories N_{traj}, roll-out horizon H, smoothing parameter r, and (fixed) stepsize η.
repeat
  (1) (Perturbed) policy evaluation:
  for $j = 1, \ldots, N_{\text{traj}}$ **do**
    sample a perturbed policy $\hat{K}^i = K^i + U^j$ where and $U^j \sim \text{Uniform}(\mathbb{S}_r)$.
    roll out $\hat{K}^i$ from sampled initial state $x_0 \sim \mathcal{D}$, over the horizon H to estimate the cost-to-go

$$\hat{f}^j = \sum_{t=0}^{H} g_t$$

    where $g_t := g(x_t, \hat{K}^i x_t)$ is the stage cost incurred at time t.
  end for
  return cost-to-go and perturbation $\{\hat{f}^j, U^j\}_{j=1}^{N_{\text{traj}}}$.
  (2) Policy improvement:
  estimate the gradient

$$\widehat{\nabla_K f(K^i)} = \frac{1}{N_{\text{traj}}} \sum_{j=1}^{N_{\text{traj}}} \frac{d}{r^2} \hat{f}^j U^j \qquad (12)$$

  $K^{i+1} \leftarrow \text{ProxGrad}(\widehat{\nabla_K f(K^i)}, \eta, r, \lambda)$ (in Alg. 2).
  return next iterate K^{i+1}.
until stopping criterion $\|K^{i+1} - K^i\| \leq \epsilon$ is satisfied.

for some polynomials h. Then, with the same stepsize in Eq. (11), there exist iteration N at most $4\kappa \log\left(\frac{\|K^0 - K^\star\|_F}{\epsilon}\right)$ such that $\left\|K^N - K^\star\right\| \leq \epsilon$ with at least $1 - o(\epsilon^{n-1})$ probability. Moreover, it converges linearly,

$$\left\|K^i - K^\star\right\|^2 \leq \left(1 - \frac{1}{2\kappa}\right)^i \left\|K^0 - K^\star\right\|^2,$$

for the iteration $i = 1, \ldots, N$, where $\kappa = \eta\sigma_{\min}(\Sigma_0)\sigma_{\min}(R) > 1$.

Remark. For (unregularized) LQR, the model-free policy gradient method (Fazel et al., 2018) is the first one that adopted a smoothing procedure to estimate the gradient. Similar to this, our model-free S-PI for regularized LQR also has several major challenges for deploying in practice. First one is that finding an initial policy with finite $F(K^0)$ is non-trivial, especially when the open loop system is unstable, i.e., $\rho(A) \geq 1$. Second one is its sensitivity to fixed stepsize η as in the known model setting (in Section 2.2), wrong choice of which easily makes it divergent. Note that these two challenges hold over most of gradient based methods for LQR including policy gradient (Fazel et al., 2018),

trust region policy optimization (TRPO) (Schulman et al., 2015), or proximal policy optimization (PPO) (Schulman et al., 2017). Last one is the joint sensitivity to multiple parameters N_{traj}, H, r, which may lead to high variance or large sub-optimality gap. On the other hand, REINFORCE (Williams, 1992) may suffer less variance, but with another potential difficulty: estimating the state-action value function $Q(x, u)$. Moreover, it is not clear how to derive a structured policy. However, here we adopted smoothing procedures that enable us to theoretically analyze the convergence rate and parameter dependency.

4. Experiments

In experiments, we consider a LQR system for the purpose of validating the theoretical results and basic properties of the S-PI algorithm. As mentioned in Section 3.2, the simple example with an unstable open loop system, i.e., $\rho(A) \geq 1$, is extremely sensitive to parameters even under known model setting, which may make it less in favor of the generic model-free RL approaches to deploy. Please note that our objective is total LQR cost-to-go in Eq. (2) more difficult than LQR cost-to-go averaged over time-horizon that some of works (Mania et al., 2018) considered. Under this difficulty, we demonstrate the properties of S-PI such as parameter sensitivity, convergence behaviors, and capability of balancing between LQR performance and policy structures. Finally, we illustrate the scalability of algorithms over various system dimensions.

4.1. Synthetic systems

In these experiments, we use the unstable Laplacian system (Recht, 2019).

Large Laplacian dynamics. $A \in \mathbf{R}^{n\times n}$ where

$$A_{ij} = \begin{cases} 1.1, & i = j \\ 0.1, & i = j+1 \text{ or } j = i+1 \\ 0, & \text{otherwise} \end{cases}$$

$B = Q = I_n \in \mathbf{R}^{n\times n}$ and $R = 1000 \times I_n \in \mathbf{R}^{n\times n}$.

Synthetic system parameters. For the Laplacian system, we regard $(n, m) = (3, 3)$, $(n, m) = (20, 20)$, and $(n, m) = (10^3, 10^3)$ dimension as small, medium, and large size of system. In addition, we experiment with Lasso regularizer over various $\lambda = 10^{-2} \sim 10^6$.

4.2. Algorithm parameters

Based on our theoretical results (Lemma 3) and sensitivity experiments that empirically show the dependency of stepsize η on λ (in Fig. 1), we set the initial stepsize $\eta = \frac{1}{\lambda}$. For the backtracking linesearch, we set $\beta = \frac{1}{2}$ and the convergence tolerance $\epsilon_{\text{tol}} = 10^{-6}$. We start with an initial policy K^0 from Riccati recursion.

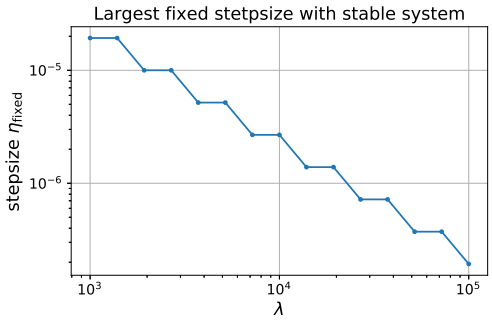

Figure 1. Largest fixed stepsize leading stable system as λ varies. This demonstrates that stepsize $\eta_{\text{fixed}} \propto \frac{1}{\lambda}$.

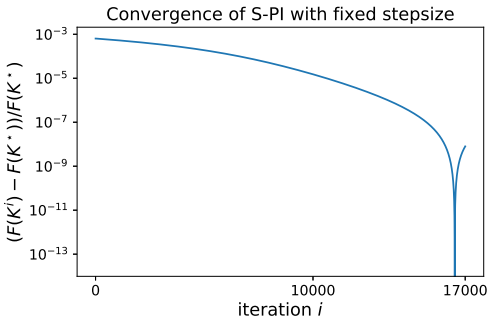

Figure 2. Convergence behavior of the Structured Policy Iteration (S-PI) with fixed stepsize for Laplacian system of $(n, m) = (3, 3)$ with $\lambda = 3000$.

4.3. Results

We denote the stationary point from S-PI (at each $\lambda > 0$) as $K^\star$ and the solution of the (unregularized) LQR as K^{lqr} .

Dependency of stepsize η on λ. Under the same problem but with different choices of weight λ, the largest fixed stepsize η that demonstrates the sequence of stable systems, i.e., $A + BK^i < 1$ actually varies, as Lemma 3 implies. Fig. 1 shows that the largest stepsize diminishes as λ increases. This motivates the choice of the initial stepsize (in linesearch) to be inversely proportional to λ.

Convergence behavior and stepsize sensitivity. S-PI with a linesearh strategy converges very fast, within 2-3 iterations for most of the Laplacian system with dimension $n = 3 \sim 10^3$ and weight $\lambda = 10^{-2} \sim 10^6$. However, S-PI with fixed stepsize may perform with subtlety even though the fixed stepsize choice is common in typical optimization problems. In Fig. 2, S-PI with fixed stepsize for the small Laplacian system $(n, m) = (3, 3)$ gets very close to optimal but begins to deviate after a certain amount of iterations. Note that it was performed over long iterations with small enough stepsize $\eta_{\text{fixed}} = 3 \times 10^{-5}$. Please refer to supplementary materials for more detailed results over various stepsizes. This illustrates that a linesearch strategy can be

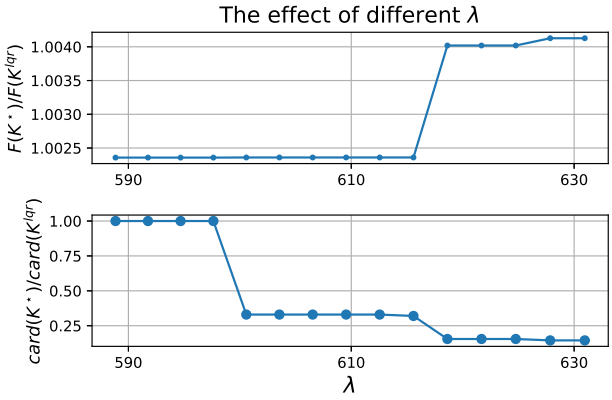

Figure 3. As λ become larger, the LQR cost slightly increases (top) within range $\lambda < 615$ whereas sparsity is significantly improved by 50% (bottom) for a Laplacian system with $(n, m) = (20, 20)$.

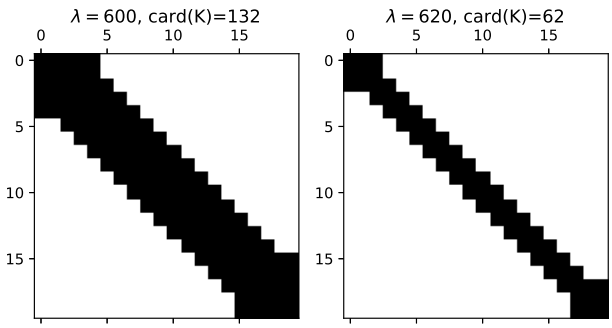

Figure 4. Sparse pattern of policy with $\lambda = 600$ and $\lambda = 620$ respectively for a Laplacian system with $n = 20$.

essential in S-PI (or even for the existing policy gradient method for LQR (Fazel et al., 2018)), even though the convergence analysis was shown with some fixed stepsize (but difficult to compute in practice).

Trade off between LQR performance and structure K. In Fig. 3 for a medium size Laplacian system, we show that as λ increases, the LQR cost $f(K^\star)$ increases whereas cardinality decreases (sparsitiy is improved). Note that the LQR performance barely changes (or is slightly worse) for $\lambda \leq 615$ but the sparsity is significantly improved by more than 50%. In Fig. 4, we show the sparsity pattern (location of non-zero elements) of the policy matrix with $\lambda = 600$ and $\lambda = 620$.

Scalability & runtime performance. In Fig. 5, we report the runtime for Laplacian system of $n = 10, \ldots, 500$. Notably, it shows the scalability of S-PI, as it takes less than 2 minutes to solve a large system with $n = 500$ dimensions, where we used a MacBook Air (with a 1.3 GHz Intel Core i5 CPU) for experiments. These results demonstrate its applicability to large-scale problems in practice.

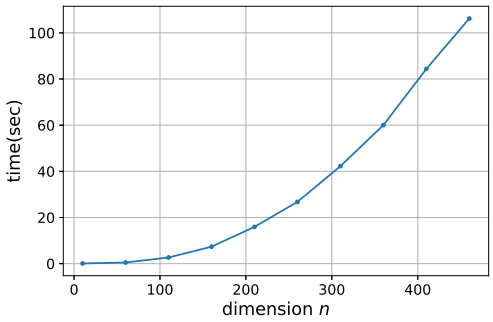

Figure 5. The elapsed time (sec.) until S-PI converges over $n = 10, \ldots, 500$ for the Laplacian system.

5. Conclusion and Discussion

In this paper, we formulated a regularized LQR problem to derive a structured linear policy and provided an efficient algorithm, Structured Policy Iteration (S-PI). We proved that S-PI guarantees to converge linearly under a proper choice of stepsize, keeping the iterate within the set of stable policy as well as decreasing the objective at each iteration. In addition, we extended S-PI for model-free setting, utilizing a smoothing procedure. We also proved its convergence guarantees with high probability under a proper choice of parameters including stepsize, horizon counts, trajectory counts, etc. In the experiments, we examined some basic properties of the S-PI such as sensitivity on stepsize and regularization parameters to convergence behaviors, which turned out to be more critical than typical optimization problems. Lastly, we demonstrated that our method is effective in terms of balancing the quality of solution and structures.

We leave for future work the practical application of other penalty functions such as low-rank and proximity regularization. There are also new extensions on using proximal Newton or proximal natural gradient method as a subroutine, beyond what was developed in S-PI in Section 2, which could be further analyzed. Even though model-free algorithm for regularized LQR was suggested with theoretical guarantees, it is extremely difficult to deploy in practice, like most of model-free approaches for LQR. Finally, developing algorithm reducing variance for (regularized) LQR, possibly like (Papini et al., 2018; Park & Ryu, 2020), as well as some practical rule of thumb on the choice of hyperparameters is another class of important problems to tackle toward model-free settings.

We described a new class of discrete time LQR problems that have yet to be studied theoretically and practically. And we discussed and demonstrated how such problems can be of practical importance despite them not being well-studied in the literature.While a few application were covered for this new class of problems, each of these contributions would open up our framework to new potential applications, providing additional benefits to future research on this topic.

References

Abbasi-Yadkori, Y. and Szepesvári, C. Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, pp. 1–26, 2011.

Al Borno, M., De Lasa, M., and Hertzmann, A. Trajectory optimization for full-body movements with complex contacts. *IEEE transactions on visualization and computer graphics*, 19(8):1405–1414, 2012.

Anderson, B. D. and Moore, J. B. *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

Balakrishnan, V. and Vandenberghe, L. Semidefinite programming duality and linear time-invariant systems. *IEEE Transactions on Automatic Control*, 48(1):30–41, 2003.

Barzilai, J. and Borwein, J. M. Two-point step size gradient methods. *IMA journal of numerical analysis*, 8(1):141–148, 1988.

Bellman, R. et al. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6): 503–515, 1954.

Bertsekas, D. P. and Shreve, S. *Stochastic optimal control: the discrete-time case*. 2004.

Bertsekas, D. P. and Tsitsiklis, J. N. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

Bertsekas, D. P., Bertsekas, D. P., Bertsekas, D. P., and Bertsekas, D. P. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

Bradtke, S. J., Ydstie, B. E., and Barto, A. G. Adaptive linear quadratic control using policy iteration. In *Proceedings of 1994 American Control Conference-ACC'94*, volume 3, pp. 3475–3479. IEEE, 1994.

Conn, A. R., Scheinberg, K., and Vicente, L. N. *Introduction to derivative-free optimization*, volume 8. Siam, 2009.

Costa, O. L. V., Fragoso, M. D., and Marques, R. P. *Discrete-time Markov jump linear systems*. Springer Science & Business Media, 2006.

De Farias, D. P. and Van Roy, B. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

Dean, S., Mania, H., Matni, N., Recht, B., and Tu, S. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, pp. 1–47, 2017.

Elmaghraby, S. E. Resource allocation via dynamic programming in activity networks. *European Journal of Operational Research*, 64(2):199–215, 1993.

Fazel, M., Ge, R., Kakade, S. M., and Mesbahi, M. Global convergence of policy gradient methods for the linear quadratic regulator. *arXiv preprint arXiv:1801.05039*, 2018.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv preprint cs/0408007*, 2004.

Florentin, J. J. Optimal control of continuous time, markov, stochastic systems. *International Journal of Electronics*, 10(6):473–488, 1961.

Hewer, G. An iterative technique for the computation of the steady state gains for the discrete optimal regulator. *IEEE Transactions on Automatic Control*, 16(4):382–384, 1971.

Ibrahimi, M., Javanmard, A., and Roy, B. V. Efficient reinforcement learning for high dimensional linear quadratic systems. In *Advances in Neural Information Processing Systems*, pp. 2636–2644, 2012.

Jaimoukha, I. M. and Kasenally, E. M. Krylov subspace methods for solving large lyapunov equations. *SIAM Journal on Numerical Analysis*, 31(1):227–251, 1994.

Kakade, S. M. A natural policy gradient. In *Advances in neural information processing systems*, pp. 1531–1538, 2002.

Kalman, R. E. When is a linear control system optimal? *Journal of Basic Engineering*, 86(1):51–60, 1964.

Kumar, V., Todorov, E., and Levine, S. Optimal control with learned local models: Application to dexterous manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 378–383. IEEE, 2016.

Lancaster, P. and Rodman, L. *Algebraic riccati equations*. Clarendon press, 1995.

Lau, R., Persiano, R., and Varaiya, P. Decentralized information and control: A network flow example. *IEEE Transactions on Automatic Control*, 17(4):466–473, 1972.

Laub, A. A schur method for solving algebraic riccati equations. *IEEE Transactions on automatic control*, 24 (6):913–921, 1979.

Lazic, N., Boutilier, C., Lu, T., Wong, E., Roy, B., Ryu, M. K., and Imwalle, G. Data center cooling using model-predictive control. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3818–3827, 2018.

Lewis, F. L., Vrabie, D., and Syrmos, V. L. *Optimal control. 3rd ed.* Hoboken, NJ: John Wiley Sons. , 2012.

Li, J.-R. and White, J. Low rank solution of lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 24(1):260–280, 2002.

Lin, F., Fardad, M., and Jovanović, M. R. Sparse feedback synthesis via the alternating direction method of multipliers. In *2012 American Control Conference (ACC)*, pp. 4765–4770. IEEE, 2012.

Mania, H., Guy, A., and Recht, B. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

Mania, H., Tu, S., and Recht, B. Certainty equivalent control of lqr is efficient. *arXiv preprint arXiv:1902.07826*, 2019.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Nerlove, M. and Arrow, K. J. Optimal advertising policy under dynamic conditions. *Economica*, pp. 129–142, 1962.

Nesterov, Y. and Spokoiny, V. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

O'Donoghue, B., Wang, Y., and Boyd, S. Min-max approximate dynamic programming. In *2011 IEEE International Symposium on Computer-Aided Control System Design (CACSD)*, pp. 424–431. IEEE, 2011.

Papini, M., Binaghi, D., Canonaco, G., Pirotta, M., and Restelli, M. Stochastic variance-reduced policy gradient. *arXiv preprint arXiv:1806.05618*, 2018.

Parikh, N., Boyd, S., et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

Park, Y. and Ryu, E. K. Linear convergence of cyclic saga. *Optimization Letters*, pp. 1–16, 2020.

Park, Y., Mahadik, K., Rossi, R. A., Wu, G., and Zhao, H. Linear quadratic regulator for resource-efficient cloud services. In *Proceedings of the ACM Symposium on Cloud Computing*, pp. 488–489, 2019.

Park, Y., Dhar, S., Boyd, S., and Shah, M. Variable metric proximal gradient method with diagonal barzilai-borwein stepsize. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3597–3601. IEEE, 2020.

Powell, W. B. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

Recht, B. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2:253–279, 2019.

Rockafellar, R. T. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976.

Sandell, N., Varaiya, P., Athans, M., and Safonov, M. Survey of decentralized control methods for large scale systems. *IEEE Transactions on automatic Control*, 23(2):108–128, 1978.

Sarimveis, H., Patrinos, P., Tarantilis, C. D., and Kiranoudis, C. T. Dynamic modeling and control of supply chain systems: A review. *Computers & Operations Research*, 35(11):3530–3561, 2008.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Sutton, R. S., Barto, A. G., et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

Tassa, Y., Erez, T., and Todorov, E. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913. IEEE, 2012.

Tu, S. and Recht, B. Least-squares temporal difference learning for the linear quadratic regulator. *arXiv preprint arXiv:1712.08642*, 2017.

Wang, S.-H. and Davison, E. On the stabilization of decentralized control systems. *IEEE Transactions on Automatic Control*, 18(5):473–478, 1973.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Wonham, W. M. Random differential equations in control theory. 1970.

Wright, S. J., Nowak, R. D., and Figueiredo, M. A. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

Wytock, M. and Kolter, J. Z. A fast algorithm for sparse controller design. *arXiv preprint arXiv:1312.4892*, 2013.