Proceedings of Machine Learning Research vol 120:1–10, 2020 2nd Annual Conference on Learning for Dynamics and Control

Fitting a Linear Control Policy to Demonstrations with a Kalman Constraint

Malayandi Palan[*†] MALAYANDI@STANFORD.EDU

Shane Barratt[*‡] SBARRATT@STANFORD.EDU

Alex McCauley[§] ALEXMCCAULEY@WAYMO.COM

Dorsa Sadigh[†‡] DORSA@STANFORD.EDU

Vikas Sindhwani[¶] SINDHWANI@GOOGLE.COM

Stephen P. Boyd[‡] BOYD@STANFORD.EDU

[†]*Department of Computer Science, Stanford University*
[‡]*Department of Electrical Engineering, Stanford University*
[§]*Waymo Research*
[¶]*Robotics at Google, New York*

[*]*Denotes equal contribution*

Editors: A. Bayen, A. Jadbabaie, G. J. Pappas, P. Parrilo, B. Recht, C. Tomlin, M. Zeilinger

Abstract

We consider the problem of learning a linear control policy for a linear dynamical system, from demonstrations of an expert regulating the system. The standard approach to this problem is (linear) policy fitting, which fits a linear policy by minimizing a loss function between the demonstrations and the policy's outputs plus a regularization function that encodes prior knowledge. Despite its simplicity, this method fails to learn policies with low or even finite cost when there are few demonstrations. We propose to add an additional constraint to the regularization function in policy fitting, that the policy is the solution to some LQR problem, *i.e.*, optimal in the stochastic control sense for some choice of quadratic cost. We refer to this constraint as a Kalman constraint. Policy fitting with a Kalman constraint requires solving an optimization problem with convex cost and bilinear constraints. We propose a heuristic method, based on the alternating direction method of multipliers (ADMM), to approximately solve this problem. An illustrative numerical experiment demonstrates that adding the Kalman constraint allows us to learn good, *i.e.*, low cost, policies even when very few data are available. An extended version of this paper is avalailable here.

Keywords: Learning from demonstrations, linear dynamical systems, convex optimization.

1. Introduction

1.1. Fitting a linear policy to demonstrations

Typically, we find a control policy for a task as follows. We first design a cost function that encodes the desired outcomes of the task; we then find a control policy that minimizes that cost function; and finally we observe or simulate the behavior of this control policy on the true system. We repeat this tuning process until we are content with the control policy performance.

This procedure, often referred to as optimization-based control, has been successfully applied to many tasks (Murray, 2009). However, for more complex tasks, it is often difficult to find a cost function that exactly captures the desired task outcomes and can be optimized effectively (Amodei and Clark, 2016; Amodei et al., 2016). For example, in autonomous driving, it is difficult, if not impossible, to construct a cost function that reliably generates "comfortable" driving behavior. For such tasks, the above procedure is expensive, time-consuming, and tedious, if it works at all.

Returning to the autonomous driving example, while it may be difficult to choose a cost function that captures "comfortable" driving behavior, it is relatively straightforward for human operators to provide demonstrations of such behavior. Similarly, for many other tasks, it is easier to collect demonstrations of desired behavior than it is to define a good cost function. This line of thought has motivated a long line of research on learning from demonstrations.

Despite this, there has been comparatively little work on learning from demonstrations in linear systems. This is surprising, since they are widely used in practice. Indeed, many systems can be modelled as linear systems, and we typically find control policies for nonlinear systems by first approximating these systems as linear. Much progress in control theory has come from studying linear systems, and we aim to continue that tradition here by considering the problem of learning a policy from demonstrations on such a system.

In this paper, we consider the problem of learning a linear policy for a known stochastic linear system from demonstrations of an expert regulating the system (*i.e.*, trying to keep the state small). The simplest method to fit a linear policy to demonstrations is (linear) policy fitting, where we minimize a loss function that measures our fit to the demonstrations plus a regularization function over linear policies. Despite its simplicity, policy fitting can lead to unstable and highly undesirable linear policies when there are few demonstrations (see §4 and Ross et al. (2011)).

Our key insight is the following: since we are trying to learn a policy for a linear system, the learned policy should be optimal for *some* quadratic cost function, *i.e.*, some linear quadratic regulator (LQR) problem. To standard policy fitting we add a *Kalman constraint*, which requires that the policy be optimal for some LQR problem. Our name for this constraint refers to the famous paper by Kalman, *"When is a control system optimal?"*, which poses the question of determining when a given linear control policy is LQR optimal for some choice of weights (Kalman, 1964). This procedure guarantees that the learned policy will retain all the desirable properies of optimal policies for LQR problems, such as stability and robustness.

We formulate policy fitting with a Kalman constraint as a bi-convex optimization problem, with a convex objective and bi-affine constraints. From this formulation we derive a heuristic, based on the alternating direction method of multipliers (ADMM), that (approximately) solves this problem efficiently. We show through an illustrative numerical experiment that this method can recover stable, low-cost policies using very few demonstrations.

1.2. Related work

Below, we discuss some relevant work from related topics, as well as the related idea of incorporating stability in system identification. (For a more complete survey, see, *e.g.*, Argall et al. (2009); Hussein et al. (2017).)

Inverse optimal control. In inverse optimal control, we are given the optimal policy and asked to find the cost function. This topic dates back to Kalman's seminal work in 1964, where he characterized a sufficient and necessary condition for a linear policy to be optimal for a given LQR problem (Kalman, 1964). (Our idea of using a Kalman constraint to regularize the policy learning procedure is directly inspired by this work.) More recently, it was shown that we can recover the cost function associated with an optimal policy for an LQR problem by solving a particular semidefinite program (SDP) (Boyd et al., 1994, §10.6). Unlike these methods, we do not assume access to the optimal policy and our focus is not on recovering the cost function but on learning an effective policy.

Inverse reinforcement learning. In inverse reinforcement learning, the goal is to learn a cost function from (noisy) demonstrations of the optimal policy. We can then find a policy by optimizing the learned cost function. (Some argue that by learning the cost function first, we get the added benefit of interpretability (Ng and Russell, 2000).) Much of the work in this space considers systems with a finite number of states and inputs (Ng and Russell, 2000; Abbeel and Ng, 2004; Ramachandran and Amir, 2007; Ziebart et al., 2008). More recent work has extended this work the continuous state and input space setting by leveraging advances in deep learning (Wulfmeier et al., 2015; Finn et al., 2016). These methods demonstrate astonishing results at times but make very few (if any) assumptions and thus typically require large numbers of demonstrations to produce sensible results. Instead, our focus here is specifically on linear systems and the few data regime.

Imitation learning. In imitation learning, which we refer to as policy fitting, the goal is to find a policy directly from (noisy) demonstrations of the optimal policy. Imitation learning (or direct policy learning or behavior cloning) typically involves learning a mapping from states to inputs via supervised learning (Pomerleau, 1989; Sammut et al., 1992; Kuniyoshi et al., 1994; Hayes and Demiris, 1994). However, standard imitation learning methods are prone to instability when only a few demonstrations are available (Ross et al., 2011) – a fact we confirm empirically in this work. Much like standard imitation learning methods, we attempt to learn a policy directly from states to inputs in this work; however, unlike prior work, we focus specifically on linear systems, leveraging their structure to add prior knowledge. Indeed, our method can be interpreted as a stability-regularized imitation learning method for linear systems.

Stable system identification. Another related problem is learning dynamics from measurements of a dynamical system. The standard approach to this problem, system identification (Ljung, 1999), frames this problem as a regression task. Recent work in this space has explored the idea of leveraging prior knowledge that the system to be identified is stable. For example, in (Lacy and Bernstein, 2003; Boots et al., 2008) the problem of fitting dynamics matrices, subject to the constraint that the dynamics are asymptotically stable, is framed as a convex optimization problem. This work has also been extended to nonlinear systems (Singh et al., 2018).

2. Linear policy fitting

Dynamics. We consider a fully-observable linear dynamical system of the form

$$x_{t+1} = Ax_t + Bu_t + \omega_t, \quad t = 0, 1, \dots, \tag{1}$$

where, at time t, $x_t \in \mathbf{R}^n$ is the system state, $u_t \in \mathbf{R}^m$ is the control input, and $\omega_t \in \mathbf{R}^n$ is a (random) disturbance. The (known) dynamics matrices are $A \in \mathbf{R}^{n\times n}$ and $B \in \mathbf{R}^{n\times m}$, and we assume that (A, B) is controllable. We assume that $\mathbf{E}[\omega_t] = 0$, $\mathbf{E}[\omega_t\omega_t^T] = W$, and that ω_t is independent of the state x_t, the control input u_t, and ω_τ for $\tau \neq t$.

Policy. A *policy* $\pi : \mathbf{R}^n \to \mathbf{R}^m$ is a function that maps the current state x_t to the control input u_t that we will apply to the system,

$$u_t = \pi(x_t), \quad t = 0, 1, \dots. \tag{2}$$

Equations (1) and (2) together define the closed-loop system. We will consider the case where π is linear, or

$$u_t = \pi(x_t) = Kx_t, \quad t = 0, 1, \dots, \tag{3}$$

where $K \in \mathbf{R}^{m \times n}$ in the gain matrix. The closed-loop dynamics are then

$$x_{t+1} = (A + BK)x_t + \omega_t, \quad t = 0, 1, \dots. \tag{4}$$

"Expert" demonstrations. We consider the case where we observe some "expert" regulating the system (1), *i.e.*, trying to keep the state x_t and input u_t small. We receive N demonstrations, that is, N pairs of states and inputs, or

$$(x^1, u^1), \dots, (x^N, u^N).$$

These pairs need not be ordered in time, optimal in any sense, or even deterministic; the expert's policy might be random. We do assume, however, that the expert is attempting in good faith to regulate the system, *i.e.*, keep x_t and u_t small, in some sense.

Linear policy fitting. The goal in linear policy fitting is to fit a linear policy K to the demonstrations. To that end, we consider the average of some loss function $l : \mathbf{R}^m \times \mathbf{R}^m \to \mathbf{R}$ (convex in its first argument) across the demonstrations:

$$L(K) = \frac{1}{N} \sum_{i=1}^{N} l(Kx^i, u^i).$$

We also assume that we have a convex regularization function $r : \mathbf{R}^{m \times n} \to \mathbf{R} \cup \{+\infty\}$ that encodes prior knowledge on K. Infinite values of r can be interpreted as constraints on K.

To fit the policy, we solve the problem

$$\text{minimize} \quad L(K) + r(K), \tag{5}$$

with variable K. This optimization problem is convex, and so can be solved efficiently (Boyd and Vandenberghe, 2004). We denote a solution to (5) by K^{pf}.

The objective function in (5) consists of two parts: the demonstration loss L and the regularization function r. The first term here encourages a good fit to the demonstrations, and the second term encourages the policy to be simpler or to be consistent with prior knowledge.

If the expert is indeed using a linear policy, *i.e.*, $x^i = K^{\text{expert}} u^i$, then we will recover K^{expert} using n demonstrations with high probability, with any reasonable choice of l, and without regularization. However, in general, policy fitting does not perform well when there are only a few demonstrations. Often, K^{pf} does not even stabilize the system, let alone mimic the expert's policy well in closed-loop simulation (see §4 and Ross et al. (2011)).

3. Policy fitting with a Kalman constraint

3.1. Linear quadratic regulator

The well-known linear quadratic regulator (LQR) problem (Bertsekas, 2017; Barratt and Boyd, 2018) chooses a policy π that minimizes the average of a quadratic cost function,

$$J(\pi) = \lim_{T \to \infty} \frac{1}{T} \mathbf{E} \sum_{t=0}^{T-1} \left(x_t^T Q x_t + u_t^T R u_t \right), \tag{6}$$

where $Q \in \mathbf{S}_+^n$ (the set of symmetric $n \times n$ positive semi-definite matrices) and $R \in \mathbf{S}_{++}^m$ (the set of symmetric $m \times m$ positive definite matrices) are the state and control weight matrices, respectively, subject to the dynamics (1) and $u_t = \pi(x_t)$. Here the expectation is taken over the initial state x_0 and disturbances $\omega_0, \omega_1, \ldots$.

It is well known that (assuming some reasonable technical conditions hold) the optimal policy π is linear, of the form

$$\pi^\star(x_t) = K^\star x_t,$$

where $K^\star \in \mathbf{R}^{m \times n}$ is the *optimal gain matrix* (Bertsekas, 2017). The optimal gain matrix $K^\star$ depends on Q and R, as well as A and B, but not W.

When is a linear policy optimal for given Q and R? Suppose we are given a linear policy K, and want to know if K is optimal for the cost (6), given the quadratic cost matrices Q and R. This is the case when

$$K = -(R + B^T PB)^{-1} B^T PA,$$

where P is the unique nonnegative solution to the algebraic Riccati equation (ARE)

$$P = Q + A^T PA - A^T PB(R + B^T PB)^{-1} B^T PA.$$

When is a linear policy optimal for *some* Q and R? Suppose we are given a linear policy K, and want to know if K is optimal for the cost (6) for *some* quadratic cost matrices Q and R, which we are free to choose. This question was addressed in (Boyd et al., 1994, §10.6), where the authors showed that we can answer this question, and get Q and R, by solving the (convex) semidefinite feasibility problem

$$\begin{array}{ll} \text{minimize} & 0, \\ \text{subject to} & Q + A^T P(A + BK) - P = 0, \\ & RK + B^T P(A + BK) = 0, \\ & P \succeq 0, \quad Q \succeq 0, \quad R \succ 0, \end{array} \tag{7}$$

with variables P, Q, and R, where $\succeq$ denotes matrix inequality, *i.e.*, with respect to the semidefinite cone. If (7) is feasible, then K is optimal for the quadratic cost matrices Q and R. On the other hand, if (7) is infeasible, then K is not optimal for any LQR problem.

Kalman constraint. We refer to the constraints in (7) as a *Kalman constraint* on K, in tribute to Kalman's seminal work on optimal control, entitled *When is a linear control system optimal?* (Kalman, 1964). A Kalman constraint on K implies that K must be optimal for some quadratic cost matrices Q and R. Policies that satisfy a Kalman constraint retain all the desirable properties of an LQR-optimal policy, such as stability and robustness.

3.2. Policy fitting with a Kalman constraint

We add a Kalman constraint to the regularization function r in the standard policy fitting problem (5). This new policy fitting problem has the form

$$\begin{array}{ll} \text{minimize} & L(K) + r(K), \\ \text{subject to} & Q + A^T P(A + BK) - P = 0, \\ & RK + B^T P(A + BK) = 0, \\ & P \succeq 0, \quad Q \succeq 0, \quad R \succeq I, \end{array} \tag{8}$$

with variables P, Q, R, and K. Note that the optimal gain matrix $K^\star$ is invariant to the relative scale of the (Q, R) matrices, *i.e.*, if K is optimal for (Q, R), it is also optimal for $(\alpha Q, \alpha R)$ for any $\alpha > 0$. Thus, we can replace the (open) constraint $R \succ 0$ in (7) with the (closed) constraint $R \succeq I$.

This problem is nonconvex and so, in general, difficult to solve exactly. However, we note that this problem is bi-convex in (P, Q, R) and K.

3.3. Solution method

ADMM. We observe that the objective in (8) is convex, and that the bi-convexity of the problem comes from the bi-affine constraints. Therefore, we propose to use the alternating direction method of multipliers (ADMM), which is guaranteed to converge to a (not necessarily optimal) stationary point for this problem, provided the penalty parameter is sufficiently large (Gao et al., 2019).

The augmented Lagrangian (Hestenes, 1969) of (8) is the extended function

$$\mathcal{L}_\rho(K, P, Q, R, Y) = L(K) + r(K) + I_C(P, Q, R) + \mathbf{Tr}(Y^T M) + \frac{\rho}{2}\|M\|_F^2,$$

where Y is the dual variable for the constraints in (8) and

$$M = \begin{bmatrix} Q + A^T P(A + BK) - P \\ RK + B^T P(BK + A) \end{bmatrix}, \quad I_C(P, Q, R) = \begin{cases} 0 & P \succeq 0, Q \succeq 0, R \succeq I, \\ +\infty & \text{otherwise.} \end{cases}$$

The ADMM algorithm alternates between minimizing the augmented Lagrangian over K, minimizing the augmented Lagrangian over (P, Q, R), and performing a dual update to Y. The full procedure is summarized in algorithm 3.1 below.

Algorithm 3.1 *Learning from demonstrations in linear systems via ADMM.*

```
given initial parameters $K^0, P^0, Q^0, R^0$, penalty parameter $\rho$, iterations $n_{\text{iter}}$.
for $k = 0, \ldots, n_{\text{iter}} - 1$
    1. $K$ step. Let $K^{k+1} = \text{argmin}_K \, \mathcal{L}_\rho(K, P^k, Q^k, R^k, Y^k)$.
    2. $(P, Q, R)$ step. Let $(P^{k+1}, Q^{k+1}, R^{k+1}) = \text{argmin}_{(P,Q,R)} \, \mathcal{L}_\rho(K^{k+1}, P, Q, R, Y^k)$.
    3. $Y$ step. Let $Y^{k+1} = Y^k + \rho \begin{bmatrix} Q^{k+1} + A^T P^{k+1}(A + BK^{k+1}) - P^{k+1} \\ R^{k+1}K^{k+1} + B^T P^{k+1}(BK^{k+1} + A) \end{bmatrix}$.
end for
return $K^{n_{\text{iter}}}$.
```

***K* step.** The update for K can be expressed as the solution to the convex optimization problem

$$\text{minimize} \qquad L(K) + r(K) + \frac{\rho}{2}\left\| \begin{bmatrix} A^T P^k BK - (P^k - Q^k - A^T P^k A - \frac{1}{\rho}Y_1^k) \\ (R^k + B^T P^k B)K - (-B^T P^k A - \frac{1}{\rho}Y_2^k) \end{bmatrix} \right\|_F^2 \qquad (9)$$

where $Y^k = (Y_1^k, Y_2^k)$. This step can be interpreted as performing policy fitting with an additional term in the regularization function that suggests that K should be approximately optimal for the LQR control problem with cost matrices Q^k, R^k, and cost-to-go matrix P^k.

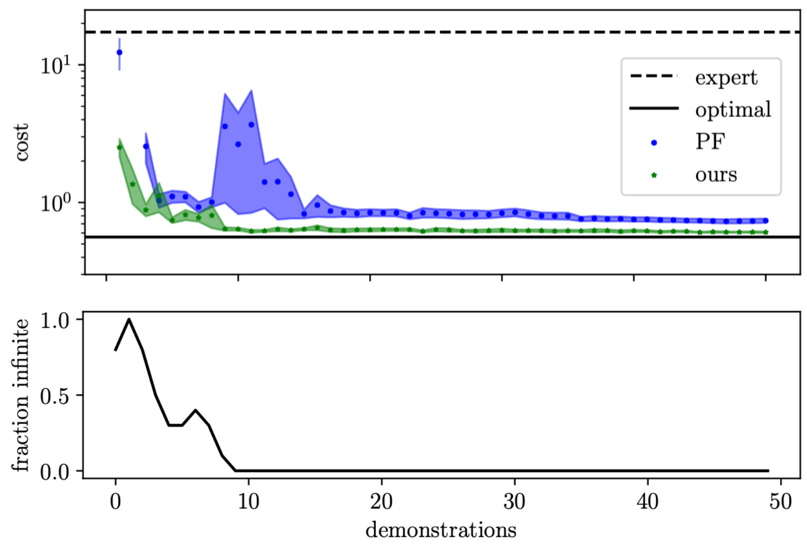

Figure 1: Example. (Top) The expected cost of policy fitting (PF) and our method for a range of numbers of demonstrations. Infinite values are ignored. (Bottom) The proportion of time that the cost for PF was infinite. Our method always attained finite cost.

(P, Q, R) step. The update for (P, Q, R) is the solution to the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \left\| \begin{bmatrix} Q + A^T P(A + BK^{k+1}) - P + \frac{1}{\rho} Y_1^k \\ RK^{k+1} + B^T P(BK^{k+1} + A) + \frac{1}{\rho} Y_2^k \end{bmatrix} \right\|_F^2 \\ \text{subject to} & P \succeq 0, Q \succeq 0, R \succeq I, \end{array} \tag{10}$$

which can be interpreted as finding the cost matrices and cost-to-go-matrix of an LQR problem such that K^{k+1} is approximately optimal for that problem.

4. Example

We illustrate our method and compare it with standard policy fitting on a simple problem. In all our experiments we use $\rho = 1$, and run ADMM with a zero initialization and 5 random initializations, keeping the K with the lowest value of $L(K) + r(K)$. We use CVXPY (Diamond and Boyd, 2016) to implement algorithm 3.1.

Imperfect LQR We first consider the case where the expert is performing imperfect regulation in an LQR problem. That is, our demonstrations have the form

$$u^i = K^\star x^i + z^i, \quad z^i \sim \mathcal{N}(0, \Sigma), \quad i = 1, \ldots, N,$$

where $K^\star$ is the solution to an LQR problem with cost matrices Q^{true} and R^{true} and $\Sigma \succ 0$. We consider loss and regularization functions

$$l(\hat{u}, u) = \|\hat{u} - u\|_2^2, \quad r(K) = (0.01)\|K\|_F^2.$$

Numerical example. We consider a system with $n = 2$ states and $m = 2$ inputs. The data is generated according to

$$A_{ij} \sim \mathcal{N}(0,1), \quad B_{ij} \sim \mathcal{N}(0,1), \quad Q^{\text{true}} = I, \quad R^{\text{true}} = I, \quad W = (0.25)I, \quad \Sigma = (0.25)I,$$

and the matrix A is scaled so that its spectral radius is one. We ran policy fitting with and without a Kalman constraint on varying numbers of demonstrations, and averaged the results over ten random seeds. In figure 1 (top) we show the expected cost (when it is finite) of policy fitting (PF) and our method versus the number of demonstrations, as well as the expected cost incurred by the expert and the optimal policy. Whereas our method never incurred infinite cost, standard policy fitting frequently did; figure 1 (bottom) shows the fraction of the time that the cost for PF was infinite.

5. Extensions and variations

General quadratic cost problem. In this paper, in the interest of clarity, we focused on the regulation of linear systems, where the goal is to keep the state and input small. However, our method can be easily adapted for a more general class of problems, such as tracking problems, where the goal is to keep the state close to a given trajectory. To do so, we simply need to define the quadratic stage cost in its more general form

$$\begin{bmatrix} x_t \\ u_t \\ 1 \end{bmatrix}^T Q \begin{bmatrix} x_t \\ u_t \\ 1 \end{bmatrix},$$

where $Q \in \mathbf{S}_+^{n+m+1}$. We can then replace the constraints in (8) with the appropriate constraints.

Finite-horizon problem. Similarly, in this paper, we only considered the time-invariant, infinite horizon problem, where the dynamics function is as given in (1). However, our method can be easily extended for time-varying, finite-horizon problems, where the dynamics function is given by

$$x_{t+1} = A_t x_t + B_t u_t + \omega_t, \quad t = 0, 1, \ldots, T-1,$$

where T is the horizon of the problem. (In this problem, the cost function is similarly truncated to T steps and is also time-varying). Here, our goal is to learn a policy for each time-step, $K_0^\star, K_1^\star, \ldots, K_T^\star$ instead of a single policy.

To solve this problem using our method, we first need replace the constraints in (8) with analogous constraints for each time-step. (There are thus $5T$ constraints as opposed to 5.) We can then adapt our algorithm to solve this problem by following the same steps outlined in Section 3.3.

6. Conclusion

In this paper, we introduced an efficient method for learning policies from demonstrations in linear systems and showed in an illustrative experiment that this method outperforms a widely-used baseline. Our method, which is based on convex optimization, is easy to implement and consistently produces reliable results, in contrast to gradient-based methods that are difficult to optimize. We believe that this method and its extensions (see, Section 5) have wide-ranging practical applications, especially in the domain of autonomous driving; indeed, a rigorous examination of this claim is the subject of future work. We are very optimistic about the potential of convex optimization to solve modern control problems and believe that this space will reemerge as a fruitful area for research in coming years.

Acknowledgements

Shane Barratt is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1656518.

References

P. Abbeel and A. Ng. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning*, page 1. ACM, 2004.

D. Amodei and J. Clark. Faulty reward functions in the wild. https://blog.openai.com/faulty-reward-functions, 2016.

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

S. Barratt and S. Boyd. Stochastic control with affine dynamics and extended quadratic costs. *arXiv preprint arXiv:1811.00168*, 2018.

D. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific, 2017.

Byron Boots, Geoffrey Gordon, and Sajid Siddiqi. A constraint generation approach to learning stable linear dynamical systems. In *Advances in Neural Information Processing Systems*, pages 1329–1336, 2008.

S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*, volume 15. SIAM, 1994.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*, pages 49–58, 2016.

W. Gao, D. Goldfarb, and F. Curtis. ADMM for multiaffine constrained optimization. *Optimization Methods and Software*, pages 1–47, 2019.

G. Hayes and J. Demiris. *A robot controller using learning by imitation*. University of Edinburgh, Department of Artificial Intelligence, 1994.

M. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, 1969.

A. Hussein, M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):21, 2017.

R. Kalman. When is a linear control system optimal? 1964.

Y. Kuniyoshi, M. Inaba, and H. Inoue. Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation*, 10(6):799–822, 1994.

Seth Lacy and Dennis Bernstein. Subspace identification with guaranteed stability using constrained optimization. *IEEE Transactions on Automatic Control*, 48(7):1259–1263, 2003.

Lennart Ljung. System identification. *Wiley Encyclopedia of Electrical and Electronics Engineering*, pages 1–19, 1999.

Richard Murray. Optimization-based control. *California Institute of Technology, CA*, pages 111–128, 2009.

A. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, volume 1, page 2, 2000.

D. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, pages 305–313, 1989.

D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *International Joint Conference on Artificial Intelligence*, volume 7, pages 2586–2591, 2007.

S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

C. Sammut, S. Hurst, D. Kedzier, and D. Michie. Learning to fly. In *Machine Learning Proceedings*, pages 385–393. Elsevier, 1992.

Sumeet Singh, Vikas Sindhwani, Jean-Jacques Slotine, and Marco Pavone. Learning stabilizable dynamical systems via control contraction metrics. *arXiv preprint arXiv:1808.00113*, 2018.

M. Wulfmeier, P. Ondruska, and I. Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

B. Ziebart, A. Maas, J. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. 2008.