A Lagrangian Method for Inverse Problems in Reinforcement Learning

Pierre-Luc Bacon
Department of Computer Science
Stanford University
plbacon@cs.stanford.edu

Florian Schäfer
Computing and Mathematical Sciences
California Institute of Technology
florian.schaefer@caltech.edu

Clement Gehring
Electrical Engineering and Computer Sciences
Massachusetts Institute of Technology
gehring@csail.mit.edu

Animashree Anandkumar
Computing and Mathematical Sciences
California Institute of Technology
anima@caltech.edu

Emma Brunskill
Department of Computer Science
Stanford University
ebrun@cs.stanford.edu

Abstract

We cast inverse problems in reinforcement learning as nonlinear equality-constrained programs and propose a new game-theoretic solution method. Our approach is based on the saddle-point problem arising in the Lagrangian formulation and applies more broadly to other problems involving equilibrium constraints. As opposed to implicit differentiation, our Lagrangian method need not solve a fixed-point problem at every step. We demonstrate our approach in the context of imitation learning and in a new problem which we call the *Optimal Model Design Problem*: that of finding a Markov Decision Process model leading to policies which also perform well once evaluated under the true MDP. We show experiments in both discrete MDPs and under the continuous LQR setting.

1 Introduction

In its prototypical form, inverse reinforcement learning (Russell, 1998) is the problem of estimating a reward function for a Markov Decision Process (Puterman, 1994) consistent with the observed behavior of a rational decision maker. In econometrics, this problem has been studied by Rust (1988) under the umbrella of *structural estimation of Markov Decision Processes*. In this framework, the estimation problem goes beyond that of the reward function only and applies to other *structural* parameters such as the discount factor or the transition function. In this paper, we study the general optimization problem arising from the inverse reinforcement learning problem or its *structural estimation* counterpart with a nonlinear program of the form:

$$\text{(ECP)} \qquad \begin{aligned} &\text{maximize } \; J(\boldsymbol{x}, \boldsymbol{\theta}) \\ &\text{subject to } \; \boldsymbol{x} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) \; . \end{aligned} \tag{1}$$

where in Rust (1988) for example, J is the log-likelihood function for the MDP parameters $\boldsymbol{\theta}$ and $\boldsymbol{x}$ is the fixed-point solution to a *smooth* variant of the Bellman optimality equations. Hence, we want to find a model of an MDP such that when solving for the corresponding optimal value function, the

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

resulting optimal policy behaves similarly – in terms of log-likelihood – to a given set of demonstrated trajectories. A close relative to inverse reinforcement learning (IRL) is also obtained by replacing the log-likelihood objective with the expected return. This problem, termed *optimal reward design* problem by Sorg et al. (2010), then consists in finding parameters for a synthetic reward function such that the policies derived from it also perform well under the true objective. In the same way that the structural estimation problem is a generalization of the typical inverse RL setting, we can also extend the scope of optimal reward design to what we call the *optimal model design* (OMD) problem in which we seek a full MDP model within a designated parametric family.

While conceptually different at first glance, both IRL and OMD share the same problem structure: the maximization of a scalar objective of the model parameters subject to a fixed-point constraint. Fundamentally, both problems are *inverse control problems* in which the model parameters are *hidden* but differ in how their *empirical validity* (Rust, 1988) is established: with the log-likelihood in the structural estimation setting and via the expected true return for OMD. Just as with the IRL assumption that an agent's *internal* reward function is subjective, the OMD problem also posits that an agent ought to form models in accordance with its own belief about how the environment behaves. This principle is reminiscent of early ideas on predictive representation of states (Littman et al., 2001) or *subjective* localization and mapping (Bowling et al., 2005). However, due to its clear formulation as an nonlinear program, there is no ambiguity regarding how to address *discovery* (Sutton & Barto, 2018) – how to entice an agent to find the right predictions for itself – as it happens *implicitly* by virtue of solving the optimization problem itself.

The implicit differentiation (Griewank & Walther, 2008) approach underlying many IRL formulations (Rust, 1988; Neu & Szepesvári, 2007; Amos et al., 2018) involves solving a fixed-point problem at every step in what amounts to a projection onto the feasible set. In this paper, we propose an alternative which decouples the problem of maximizing J with that of satisfying the fixed-point constraint. We achieve this goal by finding a saddle-point solution to the Lagrangian problem by adapting the game-theoretic approach of Schäfer & Anandkumar (2019). Compared to implicit differentiation, our *competitive differentiation* approach does not require the fixed-point constraint to be satisfied at every step to make progress towards the overall solution. Furthermore, it retains the desirable memory characteristic of implicit differentiation while avoiding its computational overhead. Due to its ties to constrained optimization, competitive differentiation applies naturally to other forms of control methods such as LQR (Anderson & Moore, 1990) and can accommodate additional constraints (eg. safety, robustness, energy, etc.) seamlessly.

2 Problem Formulation

Per Rust (1988), the structural estimation problem for Markov Decision Processes consists in finding model parameters for the reward function, transition probability function and discount factor such that a policy derived from the resulting MDP maximizes the likelihood of a given set of trajectories. In order to make this problem continuously differentiable, Rust (1988) uses a smooth variant [1] of the Bellman optimality equations (Bellman, 1957) in which the optimal smooth value function satisfies:

$$\tilde{\boldsymbol{v}}^\star := \operatorname*{lse}_{\pi \in \text{MD}} \boldsymbol{r}_\pi + \gamma \boldsymbol{P}_\pi \tilde{\boldsymbol{v}}_\pi ,$$

where "lse" stands for log-sum-exp, $\boldsymbol{P}_\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}, [\boldsymbol{P}_\pi]_{ij} := P(j|i,\pi(i))$ and $\boldsymbol{r}_\pi \in \mathbb{R}^{|\mathcal{S}|}, [\boldsymbol{r}]_i := r(i,\pi(i))$. As usual (Puterman, 1994), the soft maximization is performed component-wise rather than over the space of stationary Markov deterministic policies "MD". It follows that the smooth greedy policy is a stochastic policy, which we denote in the context of our problem as $\pi_{\boldsymbol{x},\boldsymbol{\theta}} : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ to highlight its dependence on $\boldsymbol{\theta}$ via a composition of the form: $\boldsymbol{\theta} \overset{\boldsymbol{\phi}}{\mapsto} \boldsymbol{x} \overset{\boldsymbol{\psi}}{\mapsto} \pi_{\boldsymbol{x},\boldsymbol{\theta}}$. Here, $\boldsymbol{\phi}$ is an implicit function of the model parameters $\boldsymbol{\theta}$ to the optimal smooth "action-value" function $\boldsymbol{x}$ and $\boldsymbol{\psi}$ is the soft-argmax function. Our structural estimation problem can then be written as:

$$\text{(SEP)} \qquad \begin{aligned} &\text{maximize} \;\; \mathbb{E}\left[\log P_0(S_0) + \sum_{t=0}^{T-1} \log \pi_{\boldsymbol{x},\boldsymbol{\theta}}(A_t|S_t) P(S_{t+1}|S_t, A_t)\right] \\ &\text{subject to} \;\; \boldsymbol{x} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) \;\; , \end{aligned}$$

[1] The smooth Bellman operator of Rust (1988) is the same one appearing in maximum entropy reinforcement learning (Ziebart, 2010; Fox et al., 2016; Haarnoja et al., 2017).

where $\mathbf{f}$ is the smooth Bellman mapping. The expression inside the expectation – taken over the distribution of trajectories under the *true* MDP – is the log-likelihood for $Z_t(\omega) \coloneqq (s_0, a_0, \ldots, s_t)$ with probability mass function $P_{\boldsymbol{x},\boldsymbol{\theta}}(Z_t = s_0, a_0, \ldots, s_T) \coloneqq P_0(s_0) \prod_{t=0}^{T-1} \pi_{\boldsymbol{x},\boldsymbol{\theta}}(a_t|s_t) P(s_{t+1}|s_t, a_t)$. Hence, the expected log-likelihood objective can be conceptualized as a return-maximization problem (subject to constraints) where the "reward" function is defined as: $y(s_t, a_t, s_{t+1}) \coloneqq \log \pi(s_t|a_t) P(s_{t+1}|s_t, a_t)$. Viewing the expected log-likelihood objective in this form is helpful to appreciate the similarity with the optimal model design problem formulated as:

$$\text{(OMD)} \qquad \begin{aligned} &\text{maximize } \mathbb{E}_{\boldsymbol{x},\boldsymbol{\theta}}\left[\sum_{t=0}^{T} \gamma^t r(S_t, A_t)\right] \\ &\text{subject to } \boldsymbol{x} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) \ , \end{aligned}$$

and where the infinite horizon setting obtained by taking $t \to \infty$. As opposed to (SEP), the expectation in OMD is taken with respect to the dynamics induced by the policy $\pi_{\boldsymbol{x},\boldsymbol{\theta}}$ under the true MDP: the decision variables $(\boldsymbol{x}, \boldsymbol{\theta})$ appear outside the expectation and not inside. Because of this difference, a gradient estimator (L'Ecuyer, 1991) such as REINFORCE (Williams, 1992) is required in the OMD setting while a sample analogue of the expected log-likelihood is enough for the structural estimation problem.

2.1 A Lagrangian Perspective

We take the Lagrangian function corresponding to general equality-constrained problem (ECP) as the starting point of our discussion:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}) \coloneqq J(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{\lambda}^\top (\boldsymbol{x} - \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})) \ .$$

Hence, if $(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)$ is a local maximum for (ECP) then there must also be a unique $\boldsymbol{\lambda}^\star \in \mathbb{R}^m$ such that $\nabla \mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star, \boldsymbol{\lambda}^\star) = \mathbf{0}$. By solving for this $\boldsymbol{\lambda}^\star$, we find that when $(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)$ is a local maximum of (ECP) then:

$$\frac{\partial J(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)}{\partial \boldsymbol{\theta}} + \frac{\partial J(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)}{\partial \boldsymbol{x}} \left(\boldsymbol{I} - \frac{\partial \mathbf{f}(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)}{\partial \boldsymbol{x}}\right)^{-1} \frac{\partial \mathbf{f}(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)}{\partial \boldsymbol{\theta}} = \mathbf{0} \ . \tag{2}$$

given that $\rho(\frac{\partial \mathbf{f}(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)}{\partial \boldsymbol{x}}) < 1$, which is satisfied if $\mathbf{f}$ is a contraction mapping. Equation 2 can then be read as the first-order optimality condition for an unconstrained problem. This unconstrained form follows from the *implicit* relationship between the parameters and the fixed-point $\boldsymbol{x}^\star$ which depends on $\boldsymbol{\theta}$ via $\mathbf{f}$ only in the limit of the corresponding fixed-point iteration procedure. If we assume that there exists an unique fixed-point $\boldsymbol{x}^\star$ for every $\boldsymbol{\theta}$ and that the Jacobian of $\mathbf{F}(\boldsymbol{x}, \boldsymbol{\theta}) \coloneqq \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{x}$ exists and is invertible for every pair $(\boldsymbol{x}^\star, \boldsymbol{\theta})$, then the implicit function theorem (Bertsekas, 1999, A.25) tells us that there exists a continuous function $\boldsymbol{\phi} : \mathbb{R}^n \to \mathbb{R}^m$ with the property that $\boldsymbol{\phi}(\boldsymbol{\theta}) = \boldsymbol{x}^\star$ such that (ECP) can now be written as:

$$\text{maximize } \ J(\boldsymbol{\phi}(\boldsymbol{\theta}), \boldsymbol{\theta}), \ \ \boldsymbol{\theta} \in \mathbb{R}^n \ .$$

Furthermore, the total derivative of $\boldsymbol{\phi}$ is

$$\frac{d\boldsymbol{\phi}(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \left(\boldsymbol{I} - \frac{\partial \mathbf{f}(\boldsymbol{\phi}(\boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{x}}\right)^{-1} \frac{\partial \mathbf{f}(\boldsymbol{\phi}(\boldsymbol{\theta}), \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \ , \tag{3}$$

which also appears in equation 2. Hence, implicit differentiation (Griewank & Walther, 2008) can be seen as a transformation of the nonlinear constrained problem (ECP) into an unconstrained one. The idea of eliminating the constraints also underlies what we may call *process-oriented* methods [2]: methods which *differentiate through* the dynamics of the underlying iterative process (Sutton, 1992; Andrychowicz et al., 2016; Duan et al., 2016; Tamar et al., 2016; Finn et al., 2017; Ravi & Larochelle, 2017; Xu et al., 2018). The process-oriented approximation to (ECP) is:

$$\begin{aligned} &\text{maximize } \ J(\boldsymbol{x}_T, \boldsymbol{\theta}) \\ &\text{subject to } \ \boldsymbol{x}_{t+1} = \mathbf{f}(\boldsymbol{x}_t, \boldsymbol{\theta}), \ \ t = 0, \ldots, T-1 \\ &\text{given } \ \boldsymbol{x}_0 \ \text{ and } \ T \in \mathbb{Z}^+, T < \infty \ . \end{aligned}$$

[2]We can also show (see appendix) that process-oriented methods are a subcase of discrete-time optimal control (Bertsekas, 1999). The recursive equation 4 is related to the so-called *adjoint equation* in control theory.

The unconstrained counterpart is defined using the set of functions (Gilbert, 1992, equation 6) $\{\boldsymbol{\phi}_t : \boldsymbol{\phi}_t(\boldsymbol{\theta}) = \mathbf{f}^t(\boldsymbol{x}_0, \boldsymbol{\theta})\}_{t=0}^{T}$ for the problem:

$$\text{maximize} \;\; J(\boldsymbol{\phi}_T(\boldsymbol{\theta}), \boldsymbol{\theta}), \;\; \boldsymbol{\theta} \in \mathbb{R}^n \;\; .$$

When applying the chain rule for the total derivative of $\boldsymbol{\phi}_T$, we get the recursion:

$$\frac{d\boldsymbol{\phi}_T(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \frac{\partial \mathbf{f}(\boldsymbol{x}_{T-1}, \boldsymbol{\theta})}{\partial \boldsymbol{x}} \frac{d\boldsymbol{\phi}_{T-1}(\boldsymbol{\theta})}{d\boldsymbol{\theta}} + \frac{\partial \mathbf{f}(\boldsymbol{x}_{T-1}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \;\; . \tag{4}$$

Under some assumptions, it can be shown (Gilbert, 1992, proposition 1) that as $t \to \infty$ and $\boldsymbol{\phi}_t(\boldsymbol{\theta}) \to \boldsymbol{\phi}(\boldsymbol{\theta}) = \boldsymbol{x}^\star$, it also follows that $\nabla\boldsymbol{\phi}_t(\boldsymbol{\theta}) \to \nabla\boldsymbol{\phi}(\boldsymbol{\theta})$. That is, the convergence of the inner fixed-point procedure also implies that of the adjoint fixed-point recursion (Christianson, 1994). Correspondingly, the solution to the process-oriented program only coincides with the original problem (ECP) in the limit of $t \to \infty$.

2.2 A "Competitive Differentiation" Approach

The Lagrangian perspective allowed us to elucidate the origins of implicit differentiation and its process-oriented approximation. Rather than going through a transformation of our original constrained problem into an unconstrained one, we propose to tackle (ECP) directly using Lagrangian methods (Bertsekas, 1999). Algorithms of this kind can be obtained for example by seeking for a solution to the stationary conditions $\nabla\mathcal{L}(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star, \boldsymbol{\lambda}^\star) = \mathbf{0}$ using a root-finding algorithm such as Newton's method [3]. More simply, we could also use fixed-point iterations with the mapping $\mathbf{F}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}) \coloneqq \nabla\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\lambda})$ and leading to the following primal-dual updates:

$$\Delta(\boldsymbol{x}, \boldsymbol{\theta}) \coloneqq \nabla_{\boldsymbol{x},\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}), \;\; \text{and} \;\; \Delta\boldsymbol{\lambda} \coloneqq -\nabla_{\boldsymbol{\lambda}}\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}) \;\; .$$

This algorithm can be shown (Bertsekas, 1982, p. 232) to converge given an initial estimate in the neighborhood of the optimal values $(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star, \boldsymbol{\lambda}^\star)$. In our experience, the local nature of this algorithm makes it difficult to use in practice due its tendency to diverge. This instability is closely related to the oscillatory behavior of alternating gradient descent in the training of Generative Adversarial Networks (Goodfellow et al., 2014). In this paper, we leverage the synergy between saddle-point optimization and the Lagrangian formulation to develop a stable method based on the following problem:

$$\max_{\boldsymbol{x},\boldsymbol{\theta}} \min_{\boldsymbol{\lambda}} J(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{\lambda}^\top(\boldsymbol{x} - \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})) \;\; . \tag{5}$$

If a candidate solution for the decision variables $(\boldsymbol{x}, \boldsymbol{\theta})$ does not satisfy the fixed-point equality constraint, the inner "min opponent" can choose $\boldsymbol{\lambda} \to \infty$ to defeat the "max player" over the performance measure J; if the constraint is satisfied, the "max player" is free to maximize J. Hence, this formulation preserves the structure of the original problem: that of maximizing the performance measure without violating the constraints. Equipped with this game-theoretic perspective on (ECP), we apply the competitive gradient ascent (CGA) method of Schäfer & Anandkumar (2019) to find an equilibrium solution to our two-player game. The application of competitive gradient ascent[4] to the Lagrangian game in equation 5 leads to the following updates:

$$\begin{pmatrix} \Delta(\boldsymbol{\theta}, \boldsymbol{x}) \\ \Delta\boldsymbol{\lambda} \end{pmatrix} \coloneqq \begin{pmatrix} \boldsymbol{I} & -\eta\boldsymbol{A} \\ \eta\boldsymbol{A}^\top & \boldsymbol{I} \end{pmatrix}^{-1} \begin{pmatrix} \nabla J(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{\lambda}^\top(\boldsymbol{I} - \nabla\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})) \\ \mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{x} \end{pmatrix} \tag{6}$$

where $\boldsymbol{A} \in \mathbb{R}^{(m+n)\times(m+n)}$, $\boldsymbol{A} \coloneqq \boldsymbol{I} - \nabla\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})$ and $\eta \in \mathbb{R}$ is a step size parameter. By using Schur complementation, the resulting update can be decoupled as

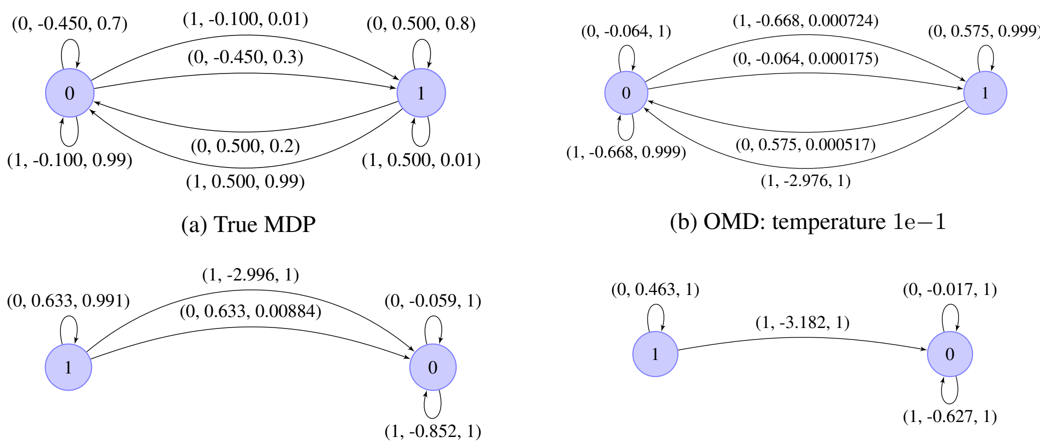
$$\Delta(\boldsymbol{\theta}, \boldsymbol{x}) \coloneqq \eta\left(\boldsymbol{I} + \eta^2\boldsymbol{A}\boldsymbol{A}^\top\right)^{-1}\left(\nabla J(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{\lambda}^\top(\boldsymbol{I} - \nabla\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})) + \eta\boldsymbol{A}(\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{x})\right) \tag{7}$$

$$\Delta\boldsymbol{\lambda} \coloneqq \eta\left(\boldsymbol{I} + \eta^2\boldsymbol{A}^\top\boldsymbol{A}\right)^{-1}\left(\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta}) - \boldsymbol{x} + \eta\boldsymbol{A}^\top(\nabla J(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{\lambda}^\top(\boldsymbol{I} - \nabla\mathbf{f}(\boldsymbol{x}, \boldsymbol{\theta})))\right). \tag{8}$$

In practice, the primal-dual updates can be obtained efficiently by solving the corresponding linear system with a matrix-free solver: using basic linear iterations (Varga, 1962) or via conjugate gradient methods (Hestenes & Stiefel, 1952) for example.

[3]This idea leads to the so-called Sequential Quadratic Programming (SQP) methods (Bertsekas, 1999).

[4]See equation 3 of Schäfer & Anandkumar (2019) for the general form of CGA



(a) True MDP

(b) OMD: temperature $1\mathrm{e}{-1}$

(c) OMD: temperature $1\mathrm{e}{-2}$

(d) OMD: temperature $1\mathrm{e}{-3}$

Figure 1: Optimal Design Problem: the transition and reward functions are estimated indirectly through the performance of the policy derived from them. By varying the temperature of the logits, we obtain optimal solutions with different levels of sparsity. Edges are labeled by (action, reward, probability) and those with transition probability less than $1\mathrm{e}-5$ are omitted
.

3 Demonstration

We apply our Lagrangian approach to the MDP of Dadashi et al. (2019, figure 2d) under the optimal model design problem. The reward and transition probability functions are specified in figure 1a where the edge labels are triples of the form: action, reward, transition probability. We provide the true discount factor (0.9) but attempt to recover a reward model and transition model consistent with the desired objective under the OMD formulation.

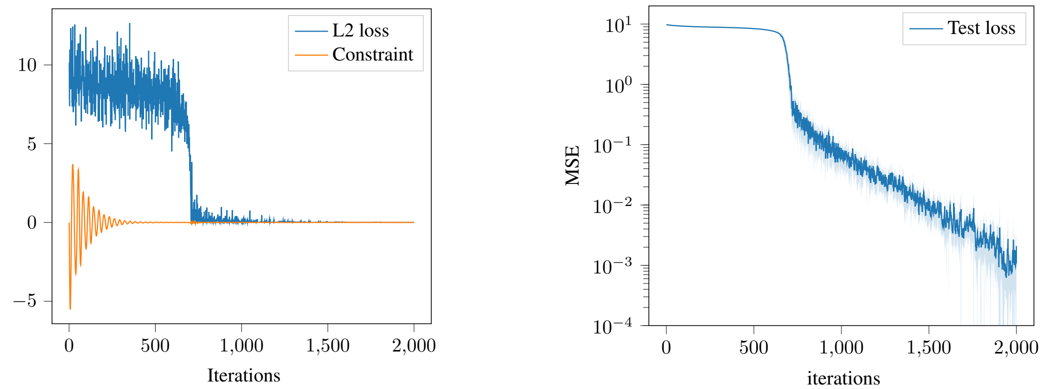
We use a tabular representation for the reward model and for the logits of the transition model which we pass through the soft-argmax function to obtain a proper conditional probability distribution. Furthermore, we scale the logits by a temperature parameter to control the desired level of sparsity in the transition model. All model parameters are initialized to zero. In the OMD experiment, we compute the optimal expected return under a uniform initial distribution (≈ 1.0272725) and optimize our solution until it reaches this level of performance within six digits of accuracy. For qualitative purposes, we compute the performance measures exactly rather than by sampling, thereby eliminating randomness as a confounder in our results. Being an inverse problem, there may be multiple OMD solutions consistent with our desire to obtain optimal policies under the true MDP. This is what we observe in practice with the reward and transition models found under the OMD setting being different from the true MDP (figure 1a) but still preserving optimality under the original MDP. By decreasing the temperature for the logits of the transition model, we can also control the level of sparsity of the final solution as shown in figures 1b, 1c and 1d. This suggests that the OMD formulation may also provide a basis for state aggregation or model compression.

3.1 LQR Experiment

We apply our competitive differentiation approach in the context of imitation learning under the linear quadratic assumption (Anderson & Moore, 1990). Rather than using a log-likelihood objective as in Rust (1988), we aim to minimize the Euclidean distance between the actions of an optimal LQR controller derived from the discrete time algebraic Riccati equation and the demonstrated actions. Our constrained optimization problem is:

$$\begin{aligned} &\text{minimize } \ \mathbb{E}\left[\|\boldsymbol{a}_i - \pi_{\boldsymbol{X}}(\boldsymbol{s}_i)\|_2\right] \\ &\text{subject to } \ \boldsymbol{A}^\top \boldsymbol{X}\boldsymbol{A} - \left(\boldsymbol{A}^\top \boldsymbol{X}\boldsymbol{B}\right)\left(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{X}\boldsymbol{B}\right)^{-1}(\boldsymbol{B}\boldsymbol{X}\boldsymbol{A}) + \boldsymbol{Q} = \boldsymbol{0} \end{aligned}$$

where $\pi_{\boldsymbol{X}}(\boldsymbol{s}_i) := -(\boldsymbol{R} + \boldsymbol{B}^\top \boldsymbol{X}\boldsymbol{B})^{-1}(\boldsymbol{B}\boldsymbol{X}\boldsymbol{A})\boldsymbol{s}_i$ and the expectation is taken with respect to a distribution over demonstrations. We estimate this expectation by querying an optimal LQR policy



(a) A training run (1 seed) (b) Generalization (40 seeds)

.

Figure 2: Imitation learning experiment in the cartpole domain

at 1000 random states sampled around the equilibrium. Figure 2a shows the joint evolution of the player behind the imitation loss (J in our previous notation from section 2.2) and its opponent trying to satisfy the constraint. We attribute the initial plateauing of the "imitation loss player" to the need of first roughly satisfying the constraint before the loss can be systematically improved. Once the feasible set has been approached, the loss drops quickly as the direction of improvement becomes easier to identify. Figure 2b measures the generalization loss over an independent dataset of 500 states sampled at random around the equilibrium. We report the test performance across 40 random seeds and compute 99% confidence intervals. By viewing the generalization plot on a log scale, we see that our competitive differentiation converges linearly to a solution once it overcomes the initial plateau.

4 Conclusion

We propose a new Lagrange method for solving inverse problems in reinforcement in which an outer objective depends on the solution to a fixed-point constraint. While our paper focuses on inverse problems in reinforcement learning, our competitive differentiation approach applies more broadly to problems involving equilibrium constraints such as meta-learning (Bellman, 1967; Sutton, 1992; Do et al., 2007; Domke, 2010; Rajeswaran et al., 2019), or hierarchical reinforcement learning (Parr & Russell, 1998; Sutton et al., 1999; Dietterich, 2000) for example. We demonstrate our algorithm in an inverse problem that we call the Optimal Model Design Problem which extends the Optimal Reward Design problem of Sorg et al. (2010) by estimating both the rewards and dynamics.

A constrained optimization approach to structural estimation of Markov Decision Processes can be found in the field of econometrics with Su & Judd (2012). The authors propose an interior-point method (Waltz et al., 2006) to solve a problem of the same form as (ECP). Su & Judd (2012) also highlights the similarities between (ECP) and Mathematical Program with Equilibrium Constraints (Harker & Pang, 1988; Luo et al., 1996) which often use Lagrangian methods (section 2.2) such as Sequential Quadratic Programming (Luo et al., 1996, 6.4). The idea of representing the fixed point $\boldsymbol{x}$ as an implicit function of $\boldsymbol{\theta}$ is well-known in the literature on MPECs (Luo et al., 1996, sections 1.3.4, 5.4, 6.3.1) and bilevel programming (Kolstad & Lasdon, 1990; Savard & Gauvin, 1994; Colson et al., 2007). The idea of "relaxing" the automatic differentiation problem via a Lagrangian formulation is also at the core of Taylor et al. (2016) who use the Alternating Direction Method of Multipliers (Powell, 1978; Bertsekas, 1982) as an alternative to back-propagation in neural networks. The connection between reverse-mode automatic differentiation and the Lagrangian formulation finds its roots in the control literature (Kelley, 1960; Bryson, 1961; Pontryagin et al., 1962; Dreyfus, 1990) ; its introduction into the AI literature is often credited to Lecun (1988); Dreyfus (1990). The form of the optimization problem studied in this paper can also be found in control theory (Lefkowitz, 1966; Bauman, 1968; Donoghue & Lefkowitz, 1972; A. Benveniste & Cohen, 1976; Forestier & Varaiya, 1978; Wilson, 1979; White & Schlussel, 1981; Wheeler & Narendra, 1986; Haurie, 1995), process engineering (Brosilow & Nunez, 1968; Hendry et al., 1973; Uronen, 1980; Newell, 1980; Nishida et al., 1981) and more broadly in hierarchical optimization (Lasdon, 1968; Mesarović et al., 1970; Anandalingam, 1988; Anandalingam & Friesz, 1992).

References

P. Bernhard A. Benveniste and A. Cohen. On the decomposition of stochastic control problems. *IFAC Proceedings Volumes*, 9(3):651–660, June 1976.

Brandon Amos, Ivan Dario Jimenez Rodriguez, Jacob Sacks, Byron Boots, and J. Zico Kolter. Differentiable MPC for end-to-end planning and control. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 8299–8310, 2018.

G. Anandalingam. A mathematical programming model of decentralized multi-level systems. *Journal of the Operational Research Society*, 39(11):1021–1033, Nov 1988.

G. Anandalingam and T. L. Friesz. Hierarchical optimization: An introduction. *Annals of Operations Research*, 34(1):1–11, Dec 1992. ISSN 1572-9338.

Brian D. O. Anderson and John B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1990. ISBN 0-13-638560-5.

Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 3981–3989, 2016.

Edward James Bauman. Multilevel optimization techniques with application to trajectory decomposition. In *Advances in Control Systems*, pp. 159–220. Elsevier, 1968.

R. Bellman. Adaptive processes and intelligent machines. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 4: Biology and Problems of Health*, pp. 11–14, Berkeley, Calif., 1967. University of California Press.

Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

Dimitri P. Bertsekas. Chapter 4 - exact penalty methods and lagrangian methods. In *Constrained Optimization and Lagrange Multiplier Methods*, pp. 179 – 301. Academic Press, 1982. ISBN 978-0-12-093480-5.

D.P. Bertsekas. *Nonlinear programming*. Athena Scientific optimization and computation series. Athena Scientific, 1999. ISBN 9781886529007.

Michael Bowling, Ali Ghodsi, and Dana Wilkinson. Action respecting embedding. In *Proceedings of the 22nd international conference on Machine learning - ICML 2005*. ACM Press, 2005.

C. Brosilow and E. Nunez. Multi/level optimisation applied to a catalytic cracking plant. *The Canadian Journal of Chemical Engineering*, 46(3):205–212, June 1968.

A. E. Bryson. A gradient method for optimizing multi-stage allocation processes. In *Proc. Harvard Univ. Symposium on digital computers and their applications*, 1961.

Bruce Christianson. Reverse accumulation and attractive fixed points. *Optimization Methods and Software*, 3(4):311–326, jan 1994.

Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, Sep 2007. ISSN 1572-9338.

Robert Dadashi, Adrien Ali Taïga, Nicolas Le Roux, Dale Schuurmans, and Marc G. Bellemare. The value function polytope in reinforcement learning. *CoRR*, abs/1901.11524, 2019.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res.*, 13:227–303, 2000.

Chuong B. Do, Chuan-Sheng Foo, and Andrew Y. Ng. Efficient multiple hyperparameter learning for log-linear models. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pp. 377–384, 2007.

Justin Domke. Implicit differentiation by perturbation. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pp. 523–531, 2010.

J. Donoghue and I. Lefkowitz. Economic tradeoffs associated with a multilayer control strategy for a class of static systems. *IEEE Transactions on Automatic Control*, 17(1):7–15, February 1972.

Stuart E. Dreyfus. Artificial neural networks, back propagation, and the kelley-bryson gradient procedure. *Journal of Guidance, Control, and Dynamics*, 13(5):926–928, September 1990.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl$ˆ2$: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pp. 1126–1135, 2017.

J.-P. Forestier and P. Varaiya. Multilayer control of large markov chains. *IEEE Transactions on Automatic Control*, 23(2):298–305, April 1978.

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence, UAI 2016, June 25-29, 2016, New York City, NY, USA*, 2016.

Jean Charles Gilbert. Automatic differentiation and iterative processes. *Optimization Methods and Software*, 1(1):13–21, January 1992.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Andreas Griewank and Andrea Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 2008. ISBN 0898716594, 9780898716597.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1352–1361. JMLR. org, 2017.

Patrick T Harker and Jong-Shi Pang. Existence of optimal solutions to mathematical programs with equilibrium constraints. *Operations Research Letters*, 7(2):61 – 64, 1988. ISSN 0167-6377.

A. Haurie. Time scale decomposition in production planning for unreliable flexible manufacturing systems. *European Journal of Operational Research*, 82(2):339–358, April 1995.

J. E. Hendry, D. F. Rudd, and J. D. Seader. Synthesis in the design of chemical processes. *AIChE Journal*, 19(1):1–15, January 1973.

M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409, December 1952.

Henry J. Kelley. Gradient theory of optimal flight paths. *ARS Journal*, 30(10):947–954, October 1960.

C. D. Kolstad and L. S. Lasdon. Derivative evaluation and computational experience with large bilevel mathematical programs. *Journal of Optimization Theory and Applications*, 65(3):485–499, June 1990.

Leon Lasdon. Duality and decomposition in mathematical programming. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):86–100, 1968.

Yann Lecun. A theoretical framework for back-propagation. In D. Touretzky, G. Hinton, and T. Sejnowski (eds.), *Proceedings of the 1988 Connectionist Models Summer School, CMU, Pittsburg, PA*, pp. 21–28. Morgan Kaufmann, 1988.

Pierre L'Ecuyer. An overview of derivative estimation. In *1991 Winter Simulation Conference Proceedings.*, pp. 207–217, Dec 1991.

I. Lefkowitz. Multilevel approach applied to control system design. *Journal of Basic Engineering*, 88 (2):392, 1966.

Michael L. Littman, Richard S. Sutton, and Satinder P. Singh. Predictive representations of state. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pp. 1555–1561, 2001.

Zhi-Quan Luo, Jong-Shi Pang, and Daniel Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, 1996.

M.D. Mesarović, A. Torokhti, D. Macko, D. Macko, Y. Takahara, and G.A. Bürger. *Theory of Hierarchical, Multilevel, Systems*. Mathematics in Science and Engineering : a series of monographs and textbooks. Academic Press, 1970.

Gergely Neu and Csaba Szepesvári. Apprenticeship learning using inverse reinforcement learning and gradient methods. In *UAI 2007, Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22, 2007*, pp. 295–302, 2007.

R. B. Newell. A comparative study of model and goal coordination in the multilevel optimization of a double-effect evaporator. *The Canadian Journal of Chemical Engineering*, 58(2):275–278, April 1980.

Naonori Nishida, George Stephanopoulos, and A. W. Westerberg. A review of process synthesis. *AIChE Journal*, 27(3):321–351, May 1981.

James M. Ortega and Werner C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. Computer science and applied mathematics. Academic Press, 1970.

Ronald Parr and Stuart J. Russell. Reinforcement learning with hierarchies of machines. In M. I. Jordan, M. J. Kearns, and S. A. Solla (eds.), *Advances in Neural Information Processing Systems 10*, pp. 1043–1049. MIT Press, 1998.

Lev Semenovich Pontryagin, V G Boltyanskii, R V Gamkrelidze, and E F Mishchenko. *The mathematical theory of optimal processes*. Wiley, New York, NY, 1962.

M. J. D. Powell. Algorithms for nonlinear constraints that use lagrangian functions. *Mathematical Programming*, 14(1):224–248, Dec 1978.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994. ISBN 0471619779.

Aravind Rajeswaran, Chelsea Finn, Sham M. Kakade, and Sergey Levine. Meta-learning with implicit gradients. *CoRR*, abs/1909.04630, 2019.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Stuart J. Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998.*, pp. 101–103, 1998.

John Rust. Maximum likelihood estimation of discrete control processes. *SIAM journal on control and optimization*, 26(5):1006–1024, 1988.

Gilles Savard and Jacques Gauvin. The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, 15(5):265–272, June 1994.

Florian Schäfer and Anima Anandkumar. Competitive gradient descent. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, Canada*, 2019.

Jonathan Sorg, Richard L Lewis, and Satinder P Singh. Reward design via online gradient ascent. In *Advances in Neural Information Processing Systems*, pp. 2190–2198, 2010.

Che-Lin Su and Kenneth L. Judd. Constrained optimization approaches to estimation of structural models. *Econometrica*, 80(5):2213–2230, 2012.

Richard S. Sutton. Adapting bias by gradient descent: An incremental version of delta-bar-delta. In *Proceedings of the 10th National Conference on Artificial Intelligence, San Jose, CA, USA, July 12-16, 1992.*, pp. 171–176, 1992.

Richard S. Sutton, Doina Precup, and Satinder P. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1-2):181–211, 1999.

R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press, 2018. ISBN 9780262039246.

Aviv Tamar, Sergey Levine, Pieter Abbeel, Yi Wu, and Garrett Thomas. Value iteration networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 2146–2154, 2016.

Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable ADMM approach. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 2722–2731, 2016.

P. Uronen. Hierarchical production control for integrated pulp and paper mills: a survey. *IFAC Proceedings Volumes*, 13(12):575–586, 1980.

Richard S. Varga. *Matrix iterative analysis*. Prentice-Hall, Englewood Cliffs, 1962.

R.A. Waltz, J.L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107(3):391–408, Jul 2006.

R. Wheeler and K. Narendra. Decentralized learning in finite markov chains. *IEEE Transactions on Automatic Control*, 31(6):519–526, June 1986.

Chelsea C. White and Kent Schlussel. Suboptimal design for large scale, multimodule systems. *Operations Research*, 29(5):865–875, October 1981.

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

I. D. Wilson. Foundations of hierarchical control. *International Journal of Control*, 29(6):899–933, June 1979.

Zhongwen Xu, Hado P. van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pp. 2402–2413, 2018.

Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, 2010.

5 Appendix

5.1 Time-Varying Process-Oriented Formulation

We can generalize the basic process-oriented formulation to one which allows for nonstationary iterative methods (Ortega & Rheinboldt, 1970), ubiquitous in deep learning (Goodfellow et al., 2016), using stage-dependent parameters $\{\boldsymbol{\theta}_t\}_{t=0}^{T}$ and operators $\{\mathbf{f}\}_{t=0}^{T-1}$. The resulting problem can then be formulated as the following nonlinear program with equality constraints:

$$\begin{aligned}&\text{maximize } \; J(\boldsymbol{x}_T)\\ &\text{subject to } \; \boldsymbol{x}_{t+1} = \mathbf{f}_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t), \;\; t = 0, \dots, T-1\\ &\text{given } \; \boldsymbol{x}_0 \; \text{ and } \; T \in \mathbb{Z}^+, T < \infty \;\; .\end{aligned}$$

Once again, we can convert (Bertsekas, 1999, p. 212, sect. 2.6) the equality constrained problem into an unconstrained one by expressing the iterates $\{\boldsymbol{x}_t\}_{t=0}^{T}$ via a function $\boldsymbol{\phi}_t$ of all the parameters $\{\boldsymbol{\theta}_t\}_{t=0}^{T}$ applied during the inner optimization procedure:

$$\boldsymbol{\phi}_t(\boldsymbol{\theta}_0, \dots, \boldsymbol{\theta}_T) \coloneqq \mathbf{f}_{t-1}(\dots \mathbf{f}_0(\boldsymbol{x}_0, \boldsymbol{\theta}_0), \boldsymbol{\theta}_{t-1}) = \boldsymbol{x}_t \;\; .$$

The resulting time-varying, unconstrained, and process-oriented formulation counterpart to (ECP) is:

$$\text{maximize } \; J(\phi_T(\boldsymbol{\theta}_{0:T})) \;\; \text{given } \; \boldsymbol{x}_0 \; \text{ and } \; T \in \mathbb{Z}^+, T < \infty \;\; .$$

With a change of variables through $\boldsymbol{\phi}_t$, we can apply the chain rule and obtain:

$$\frac{\partial J}{\partial \boldsymbol{\theta}_t} = \frac{\partial J}{\partial \boldsymbol{x}_T}\frac{\partial \boldsymbol{\phi}_T}{\partial \boldsymbol{\theta}_t} = \frac{\partial J}{\partial \boldsymbol{x}_T}\frac{\partial \mathbf{f}_{t-1}}{\partial \boldsymbol{x}_t}\dots\frac{\partial \mathbf{f}_t}{\partial \boldsymbol{\theta}_t} \;\; .$$

By accumulating the terms from right to left (future to past), we can also write this expression recursively as:

$$\frac{\partial J}{\partial \boldsymbol{\theta}_t} = \boldsymbol{\lambda}_{t+1}^{\top}\frac{\partial \mathbf{f}_t}{\partial \boldsymbol{\theta}_t}, \;\text{where}\; \boldsymbol{\lambda}_t^{\top} = \boldsymbol{\lambda}_{t+1}^{\top}\frac{\partial \mathbf{f}_t}{\partial \boldsymbol{x}_t} \;\text{ and }\; \boldsymbol{\lambda}_T^{\top} = \frac{\partial J}{\partial \boldsymbol{x}_T} \;\; . \tag{9}$$

In control theory, the row vector $\boldsymbol{\lambda}_t^{\top}$ is called the *costate* or *adjoint* vector and is recursively updated using the *adjoint equation* (Bertsekas, 1999, 2.174). The adjoint equation coincides exactly with the computation taking place during reverse mode automatic differentiation (Griewank & Walther, 2008).

While we have assumed so far that $\{\mathbf{f}_t\}_{t=0}^{T}$ and $\{\mathbf{f}_t\}_{t=0}^{T}$ describe the dynamics of the inner iterative process, we could also consider a formulation which involves a *process model* (an *optimizer model*). This approach would amount to a *model-based* (Sutton & Barto, 2018) approach, which may be beneficial for certain class of models, such as the LQR formulation (Bertsekas, 1999). In this case, it would be interesting to quantify the effect of using an approximate inner optimization model on the overall performance of the optimization procedure.

5.2 Process-Oriented Formulation as a Discrete-Time Control Problem

A full generalization of the time-varying formulation to a discrete-time control problem can also be developed. In this case, we see $\mathbf{f}_t$ as a time-varying *transition function*, $\boldsymbol{x}_t \in \mathbb{R}^m$ as a *state vector* and $\boldsymbol{\theta}_t \in \mathbb{R}^n$ as a *control vector*. We also define J as a sum of immediate performance measures (which play the role of immediate *rewards*) of the form $g_t : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}, \; (\boldsymbol{x}, \boldsymbol{\theta}) \mapsto g_t(\boldsymbol{x}, \boldsymbol{\theta}), t = 0, \dots, T-1$ and final immediate performance $g_T : \mathbb{R}^m \to \mathbb{R}, \boldsymbol{x}_T \mapsto g_T(\boldsymbol{x}_T)$. The generalization of the time-varying formulation to the discrete-time optimal control setting can be described as:

(OCP)

$$\begin{aligned}&\text{maximize } \; J(\boldsymbol{x}_0, ..., \boldsymbol{x}_T, \boldsymbol{\theta}_0, \dots, \boldsymbol{\theta}_T) = g_T(\boldsymbol{x}_T) + \sum_{t=0}^{T-1} g(\boldsymbol{x}_t, \boldsymbol{\theta}_t)\\ &\text{subject to } \; \boldsymbol{x}_{t+1} = \mathbf{f}_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t)\\ &\text{given } \; \boldsymbol{x}_0 \; \text{ and } \; T \in \mathbb{Z}^+, T < \infty \;\; .\end{aligned} \tag{10}$$

Note that if the non-terminal immediate performance measures $g_t, t = 0, \dots, T-1$ were to be absent from (OCP), then the resulting problem would be equivalent to the time-varying formulation derived in the previous section. Bertsekas (1999, p. 213) shows that a reduction of the full (OCP) problem to the terminal case can accomplished by viewing the sum of immediate performance measures *so far* (the return so far) as a state variable. The return so far also obeys a deterministic recursive update of the form:

$$z_{t+1} = g_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + z_t, \;\; z_0 = 0 \;\; ,$$

It follows that the *total return* objective in (OCP) can be written as:

$$\begin{aligned}&\text{maximize} \;\; J(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_T, \boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_T) = g_T(\boldsymbol{x}_T) + z_T \\ &\text{subject to} \;\; \boldsymbol{x}_{t+1} = \mathbf{f}(\boldsymbol{x}_t, \boldsymbol{\theta}_t) \;\; \text{and} \;\; z_{t+1} = g_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + z_t \\ &\text{given} \;\; \boldsymbol{x}_0 \;\; \text{and} \;\; z_0 = 0 \;\; .\end{aligned}$$

We can then augment the state vector with z_t and define the transition functions and immediate performance measures as functions of both components:

$$\begin{aligned}\tilde{\boldsymbol{x}}_{t+1} = \tilde{\mathbf{f}}_t(\tilde{\boldsymbol{x}}_t, \boldsymbol{\theta}) &\coloneqq \left[\mathbf{f}_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) \quad g_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + z_t\right]^\top \\ \tilde{g}_T(\tilde{\boldsymbol{x}}_T) &\coloneqq g_T(\boldsymbol{x}_t) + z_t \;\; .\end{aligned}$$

Using equation 9, the adjoint equation over the augmented state vectors is:

$$\tilde{\boldsymbol{\lambda}}_t^\top = \tilde{\boldsymbol{\lambda}}_{t+1}^\top \frac{\partial \tilde{\mathbf{f}}_t}{\partial \tilde{\boldsymbol{x}}_t}, \;\; \tilde{\boldsymbol{\lambda}}_T^\top = \frac{\partial \tilde{\boldsymbol{g}}_T}{\partial \boldsymbol{x}_T} \;\; .$$

The output of $\tilde{\mathbf{f}}_t$ comprising of both $\boldsymbol{x}_{t+1}$ and z_{t+1}, we now have a 2×2 block Jacobian and the augmented adjoint equation is:

$$\tilde{\boldsymbol{\lambda}}_t^\top = \tilde{\boldsymbol{\lambda}}_{t+1}^\top \begin{bmatrix} \frac{\partial \mathbf{f}_t}{\boldsymbol{x}_t} & 0 \\ \frac{\partial g_t}{\partial \boldsymbol{x}_t} & 1 \end{bmatrix}, \;\; \tilde{\boldsymbol{\lambda}}_T^\top = \begin{bmatrix} \frac{\partial g_T}{\partial \boldsymbol{x}_T} & 1 \end{bmatrix} \;\; .$$

Note that the total return with respect to the augmented state is also a block vector with two components: the first one quantifying the variation of the total return for a change in $\boldsymbol{x}_T$ whereas the second one pertains to the effect of a perturbation of the return so far on the total return – a linear relationship with slope 1. It follows that the generalization of the adjoint equation equation 9 to (OCP) with non-terminal immediate performance measures is:

$$\frac{\partial J}{\partial \boldsymbol{\theta}_t} = \frac{\partial g_t}{\partial \boldsymbol{\theta}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t}{\partial \boldsymbol{\theta}_t}, \;\; \text{where} \;\; \boldsymbol{\lambda}_t^\top = \frac{\partial g_t}{\partial \boldsymbol{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t}{\partial \boldsymbol{x}_t} \;\; \text{and} \;\; \boldsymbol{\lambda}_T^\top = \frac{\partial g_T}{\partial \boldsymbol{x}_T} \;\; . \tag{11}$$

The high-level structure of this adjoint equation is similar to the one in Christianson's two-phase algorithm Christianson (1994). Due to the general formulation of (OCP), this equation however involves a non-stationary *intercept* term $\partial g_t / \partial \boldsymbol{x}_t$ and time-varying $\partial \mathbf{f}_t / \partial \boldsymbol{x}_t$. The adjoint equation derived in this section is also closely related to the Pontryagin's Maximum Principle (Pontryagin et al., 1962) in discrete-time. This connection becomes clearer (Bertsekas, 1999, proposition 2.6.1) when expressing the first-order stationary conditions for (OCP) in terms of the Hamiltonian function H_t, central in Pontryagin's formulation:

$$H_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t, \boldsymbol{\lambda}_{t+1}) \coloneqq g_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + \boldsymbol{\lambda}_{t+1}^\top \mathbf{f}_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t) \;\; .$$

Taking the gradient of the Hamiltonian with respect to each control vector, we recover equation 11 and have that for optimal parameters $\{\boldsymbol{\theta}_t^\star\}_0^k$ and $t = 0, \ldots, T$:

$$\frac{\partial H_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t^\star, \boldsymbol{\lambda}_{t+1})}{\partial \boldsymbol{\theta}_t} = \frac{\partial J(\boldsymbol{\theta}_t^\star)}{\partial \boldsymbol{\theta}_t} = 0 \;\; .$$