2011 IEEE International Symposium on Intelligent Control (ISIC)
Part of 2011 IEEE Multi-Conference on Systems and Control
Denver, CO, USA. September 28-30, 2011

Imputing a Convex Objective Function

Arezou Keshavarz Yang Wang Stephen Boyd

Abstract— **We consider an optimizing process (or parametric optimization problem), *i.e.*, an optimization problem that depends on some parameters. We present a method for imputing or estimating the objective function, based on observations of optimal or nearly optimal choices of the variable for several values of the parameter, and prior knowledge (or assumptions) about the objective. Applications include estimation of consumer utility functions from purchasing choices, estimation of value functions in control problems, given observations of an optimal (or just good) controller, and estimation of cost functions in a flow network.**

I. Introduction

Parametric optimization is a tool for calculating the optimal solution of a problem for a given set of parameters. In this paper, we consider the reverse problem: given a set of optimal (or nearly optimal) solutions, corresponding to different parameter values, how can we impute or estimate the underlying objective function? This is a very practical problem that arises in many fields such as control, robotics, economics and societal networks. There are many scenarios where we can observe a process involving agents that behave optimally (or approximately optimally), and we want to be able to say something about what is being optimized. For instance, economists assume that consumer purchases maximize a so-called utility function, minus the price paid; the utility function describes how much satisfaction the consumer receives from purchasing a product or set of products. In reality, we do not have access to a consumer's utility function; indeed, the consumer cannot express the utility function either. But we can observe consumer purchases in response to price changes. Using the methods presented in this paper, we can impute the underlying utility function based on observations of the consumer's behavior.

As a different domain where this method can be applied, we consider controller complexity reduction for general control problems. In many cases, a good but sophisticated controller may be available, for example model predictive control [1] (or a human expert), but we might desire a lower complexity controller (or one that can be automated). We can generate optimal (or closely optimal) observations using the more sophisticated controller, and then use the paradigm described in this paper to find a controller of lower complexity.

In this paper we consider a parametric optimization problem on a convex set, but with an unknown convex objective function. Given a series of observations, consisting of actions that correspond to different parameter values, we will show that we can impute (or approximately impute) the unknown objective function by solving another convex optimization problem. In cases where the underlying objective is not convex, we can use this method to obtain a convex approximation to the objective function.

The authors are with the Information Systems Laboratory, Electrical Engineering Department, Stanford University, Stanford, CA 94305-9510, USA. Email: {arezou,yw224,boyd}@stanford.edu.

A. Prior and related work

Many disciplines, from networks to economics and robotics, have considered the problem of imputing the underlying objective function of a process based on available observations. The work of Burton, Pulleyback and Toint [2] and Burton and Toint [3] focuses on solving the inverse shortest path problem of imputing the weight of the graph edges, given a set of observations of shortest path traversals. Nielsen and Jensen [4] consider the problem of modeling a decision maker's utility function from (possibly) inconsistent behavior. Ackerberg, Benkard, Berry and Pakes [5] and Bajari, Benkard and Levin [6] study the problem of estimating demand and production functions. Ng and Russell [7] and Abbeel and Ng [8] pose the problem of inverse reinforcement learning, where the assumption is that some information about the optimal policy is available (perhaps through observing an expert's behavior) and the goal is to learn the reward function. They consider a discrete and finite problem, which becomes intractable for large state spaces or action spaces. For infinite dimensional problems, they use an affine representation of the reward function, but since the state space is discretized, the approach still suffers from the so-called 'curse of dimensionality' [9]. There are many similar works in this domain, including [10], [11], [12] and [13]. This method has been very successfully applied to helicopter robots [14].

A closely related problem is the inverse problem of optimal control. In optimal control, given the stage cost function and state dynamics, we want to calculate the control policy and the Lyapunov or Bellman functions. In inverse optimal control, we are given the control policy, but want to learn the stage cost function and the Lyapunov function (or determine that none exist that make the given policy optimal). This topic has a long history, tracing from Kalman's original paper, *When is a Control System Optimal?* [15]. For a more recent treatment, see [16, §10.6], where it is shown that for a linear quadratic control problem with stage cost $x^TQx + u^TRu$, given a linear control policy $u = Kx$, we can recover Q, R (if they exist) and calculate the Lyapunov function by solving a semidefinite program (SDP).

Another related area of research is dynamic programming (DP); see [17], [18] for an overview. The underlying idea in dynamic programming is to decompose the problem into a sequence of subproblems, and solve for a global minimizer, called the value function. The Hamilton-Jacobi-Bellman equation [19] is a fixed point equation for the value function, and the dynamic programming literature has offered various algorithms for calculating the value function, including policy iteration, value iteration, and linear programming [17], [18]. The difference between our work and dynamic programming is that we assume that we are given observations of the policy's behavior (*i.e.*, a set of decisions corresponding to different parameter values), whereas in most algorithms in dynamic programming, the policy is not provided and is estimated in each iteration of the algorithm. In cases where the exact value function cannot be determined, mostly due to the so-called curse of dimensionality [9], approximate dynamic programming algorithms aim to calculate an approximate value function; see [20], [21], [18], [9]. Usually an approximate value function is obtained and is represented as a linear combination of a set of basis functions. One such algorithm, Q-learning, which was first introduced in [22], [23], has been used as a method for learning the Q-function, which is closely related to the value function, based on observing the state and action over time. Other approaches include approximate version of value iteration, policy iteration, and the linear programming formulation of DPs, where the exact value function is replaced with a linear combination of a set of basis functions [21], [18], [24].

One main difference between our approach and the works outlined above is that we do not limit the state space to discrete and finite set of values. Furthermore, our method is based on solving a convex optimization problem, which means that we avoid the curse of dimensionality [9] (at the cost of a restricted set of problems we can handle).

II. Problem statement

We consider an optimizing process, that is, a system in which a decision x is made by optimizing an objective subject to constraints, where both objective and constraints can depend on a parameter p. (This is sometimes referred to as a parametric optimization problem.) Our goal is to learn or estimate the objective function, given a set of observations consisting of parameter values $p^{(k)}$ and associated optimal decisions $x^{(k)}$, for $k = 1, \ldots, N$. We are also given prior information, which tells us the form of the objective function and constraint functions. We refer to an objective found from observations of optimal decisions as an *imputed objective function*.

We will focus here on the case when the optimizing process involves a convex optimization problem [25],

$$\begin{array}{ll} \text{minimize} & f(x,p) \\ \text{subject to} & g_i(x,p) \le 0, \quad i = 1, \ldots, m \\ & A(p)x = b(p), \end{array} \tag{1}$$

where $x \in \mathbf{R}^n$ is the variable, f, g_i, $i = 1, \ldots, m$ are differentiable and convex in x for each value of $p \in \mathcal{P}$ (the set of allowable parameter values), $A : \mathcal{P} \to \mathbf{R}^{q \times n}$, and $b : \mathcal{P} \to \mathbf{R}^q$. We say that $x \in \mathbf{R}^n$ is optimal for $p \in \mathcal{P}$ if it is a solution of problem (1). We do not assume that for each p, there is only one solution of (1); in other words, we can have several x's that are optimal for a given p.

Optimality and approximate optimality: We will assume throughout that an appropriate constraint qualification holds, so for a given $p \in \mathcal{P}$ the necessary and sufficient (Kharush-Kuhm-Tucker or KKT) conditions for x to be optimal are the existence of $\lambda \in \mathbf{R}^m_+$ and $\nu \in \mathbf{R}^q$ that satisfy the following conditions:

$$\begin{array}{ll} 1. & g_i(x,p) \le 0, \quad i = 1, \ldots, m \\ 2. & A(p)x = b(p) \\ 3. & \nabla f(x,p) + \sum_{i=1}^m \lambda_i \nabla g_i(x,p) + A(p)^T \nu = 0 \\ 4. & \lambda_i g_i(x,p) = 0, \quad i = 1, \ldots, m, \end{array} \tag{2}$$

where the gradient is taken with respect to x. Like x, the dual variables λ and ν depend on the parameter p, but to keep the notation light, we will not explicitly show this dependence. In (2), the first and second conditions are primal feasibility, the third condition is stationarity, and the fourth condition is complementary slackness.

We use these optimality conditions to define what it means for $x \in \mathbf{R}^n$ to be *approximately optimal* for the problem (1): Given a parameter $p \in \mathbf{R}^q$, $x \in \mathbf{R}^n$ is said to be approximately optimal if the conditions in (2) hold approximately. To make this more precise, we define the residuals

$$\begin{aligned} r_{\text{ineq}} &= (g_i(x,p))_+, \quad i = 1, \ldots, m, \\ r_{\text{eq}} &= A(p)x - b(p), \\ r_{\text{stat}}(\alpha, \lambda, \nu) &= \nabla f(x,p) + \textstyle\sum_{i=1}^m \lambda_i \nabla g_i(x,p) + A(p)^T \nu, \\ r_{\text{comp}}(\lambda) &= \lambda_i g_i(x,p), \quad i = 1, \ldots, m. \end{aligned}$$

The first two residuals, r_{ineq} and r_{eq} correspond to primal feasibility, and are thus not a function of the dual variables.

Given a parameter p, if the point x is optimal (and thus feasible), there exist $\lambda \in \mathbf{R}^m_+$ and $\nu \in \mathbf{R}^q$ such that all residuals are exactly zero. We say that a point x is *approximately optimal* for the problem (1) if the primal residuals r_{ineq} and r_{eq} are close to zero, and there exist $\lambda \in \mathbf{R}^m_+$ and $\nu \in \mathbf{R}^q$ such that r_{stat} and r_{comp} are close to zero.

The imputed objective problem: We assume that the inequality constraint functions $g_1, \ldots, g_m$ are known, as are A and b (which are functions from $\mathcal{P}$ into $\mathbf{R}^{q \times n}$ and $\mathbf{R}^q$, respectively). In addition, we are given a set of N optimal (or approximately optimal) points along with the associated parameter values:

$$(x^{(k)}, p^{(k)}), \quad k = 1, \ldots, N.$$

We say that $f : \mathbf{R}^n \times \mathcal{P} \to \mathbf{R}$ is a consistent objective if $x^{(k)}$ is optimal for $p^{(k)}$, for all k.

There can be many consistent objective functions. For example, if the given observations are feasible, $f = c$, where c is any constant, is always consistent. If f is a consistent objective, so is $G \circ f$, where G is any convex increasing function. This issue will be handled by normalization and regularization, described below.

In addition to the problem of non-uniqueness, we also encounter (in applications) the problem of non-existence of (exactly) consistent objectives. This can be due to measurement or modeling error, which render the samples $x^{(k)}$ only approximately optimal. When an exactly consistent objective cannot be found, we look for an imputed objective function that is approximately consistent with our data.

We will restrict ourselves to the case where f has the finite dimensional affine parametrization

$$f = \sum_{i=0}^{K} \alpha_i f_i, \quad \alpha \in \mathcal{A},$$

where f_i are pre-selected basis functions, and $\mathcal{A}$ is a convex subset of $\mathbf{R}^{K+1}$. The set $\mathcal{A}$ collects our prior information about the objective f. For example, if the basis functions f_i are convex, then $\mathcal{A} = \mathbf{R}_+^{K+1}$ is sufficient to ensure that f is a convex function. Our goal is to find $\alpha \in \mathcal{A}$ for which f is consistent (or approximately consistent) with the data, which is a finite dimensional optimization problem with variable α.

III. Our method

Our method for computing an imputed objective function involves finding weights $\alpha \in \mathcal{A}$ so that each decision $x^{(k)}$ is approximately optimal for the associated parameter value $p^{(k)}$.

For each sample $(x^{(k)}, p^{(k)})$ we first introduce dual variables $\lambda^{(k)} \in \mathbf{R}^m$ and $\nu^{(k)} \in \mathbf{R}^q$. We let $r_{\text{ineq}}^{(k)}$, $r_{\text{eq}}^{(k)}$, $r_{\text{comp}}^{(k)}$ and $r_{\text{stat}}^{(k)}$ denote the residuals corresponding to each sample and its dual variables $(x^{(k)}, p^{(k)}, \lambda^{(k)}, \nu^{(k)})$. The primal residuals $r_{\text{ineq}}^{(k)}$ and $r_{\text{eq}}^{(k)}$ are fixed for each sample, and do not depend on α. Thus, to compute an imputed objective we solve the problem

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{N} \phi\left(r_{\text{stat}}^{(k)}, r_{\text{comp}}^{(k)}\right) \\ \text{subject to} & \lambda^{(k)} \succeq 0, \quad k = 1, \ldots, N, \quad \alpha \in \mathcal{A}, \end{array} \tag{3}$$

with variables α, $\lambda^{(k)}$, $\nu^{(k)}$, $k = 1 \ldots, N$. Here, $\phi : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}_+$ is a nonnegative convex penalty function, which satisfies

$$\phi(r_{\text{stat}}, r_{\text{comp}}) = 0 \iff r_{\text{stat}} = 0, \quad r_{\text{comp}} = 0.$$

The choice of penalty function ϕ will affect the distribution of the residuals as well as the imputed objective. As an example, we can take ϕ to be any norm on $\mathbf{R}^n \times \mathbf{R}^m$. Other choices for the penalty function include Huber, deadzone-linear, or log-barrier functions [25, §6.1].

We make a few comments about this optimization problem. First, since $r_{\text{stat}}^{(k)}$ and $r_{\text{comp}}^{(k)}$ are linear in α, $\lambda^{(k)}$ and $\nu^{(k)}$, it is easy to see that the objective function is convex. In addition, the constraints are clearly convex, so (3) is a finite-dimensional convex optimization problem, which can be efficiently solved. If we solve (3) and find that the optimal value is equal to zero, then we must have

$$r_{\text{stat}}^{(k)}(\alpha, \lambda^{(k)}, \nu^{(k)}) = 0, \quad r_{\text{comp}}^{(k)}(\lambda^{(k)}) = 0, \quad k = 1, \ldots, N.$$

Furthermore, if the samples are also primal feasible, *i.e.*, $r_{\text{ineq}}^{(k)} = 0$, $r_{\text{eq}}^{(k)} = 0$, $k = 1, \ldots, N$, then our imputed objective is exactly consistent with our data. On the other hand if we solve (3) and find that the many of the dual residuals at the optimum are very large, we can conclude that our optimizing process cannot be a good model for our data.

Trivial solutions and normalization: It is very important that the set $\mathcal{A}$ contains enough prior information about our objective f; otherwise trivial solutions may easily arise. For example, when $\mathcal{A} = \mathbf{R}_+^{K+1}$, a simple solution to (3) is $\alpha = 0$, $\lambda^{(k)} = 0$, $\nu^{(k)} = 0$, $k = 1, \ldots, N$, since $r_{\text{stat}}^{(k)}$ and $r_{\text{comp}}^{(k)}$ are homogeneous in $(\alpha, \lambda^{(k)}, \nu^{(k)})$. Another case is if we can find $\alpha \in \mathcal{A}$ for which $\sum_i \alpha_i f_i = c$, where c is a constant function of x and p. Then setting $\lambda^{(k)} = 0$ and $\nu^{(k)} = 0$ will solve (3).

In some cases this may be the desired result, but for most applications we will consider, we have prior knowledge that the underlying objective function has a non-trivial dependence on x and p. For these problems a constant imputed objective is implausible, and should be excluded from the set $\mathcal{A}$. The appropriate method for doing this depends on the specific application, but here we give a normalization approach that is applicable to a large class of problems.

We assume that $\mathcal{A} = \mathbf{R}_+^{K+1}$, and $f_0, \ldots, f_K$ are non-constant, convex functions. For many applications, we know a part of the objective ahead of time. We encode this knowledge in $\mathcal{A}$ by adding the condition $\alpha_0 = 1$, so f_0 is the part of the objective that is fixed. This is a generic normalization method; we will see other normalization methods, more appropriate for specific problems, in the examples considered later.

IV. Applications

Here we discuss potential applications of imputing an objective. One generic application is prediction, *i.e.*, guessing the decision x, given a parameter p. Another generic application is control or manipulation, *i.e.*, choosing a value of p that produces a desired x. More specific applications are discussed below.

A. Consumer behavior

We consider a set of n products with prices p_i, $i = 1, \ldots, n$. Let x_i be the consumer demand for product i. We assume that the consumer chooses x to maximize an (unknown) concave and nondecreasing utility U, minus the cost $p^T x$. The optimizing process involves solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & p^T x - U(x) \\ \text{subject to} & x \geq 0, \end{array} \tag{4}$$

with variable $x \in \mathbf{R}^n$ and parameter $p \in \mathbf{R}^n$. We are given samples $(x^{(k)}, p^{(k)})$ (for example, based on demand patterns observed in some test situation), and our goal is to impute the utility function $U : \mathbf{R}_+^n \to \mathbf{R}$.

Once we have imputed a utility function, we can use it to find a set of prices p_{des} that would achieve a target demand level x_{des}.

Imputed objective problem: For any given price p, the consumer spends a total of $p^T x$ and derives a utility of $U(x)$ from the purchase. For this application, we will consider a concave quadratic representation for U, *i.e.*, $U(x) = x^T Qx + 2r^T x$, where $Q \in \mathbf{S}^n_-$ (the set of symmetric nonpositive definite matrices), $r \in \mathbf{R}^n$. Note that we assume that the utility function has zero offset, since adding a constant to the utility function will not affect the optimal consumption levels. Furthermore, we will assume that U is nondecreasing over the range $[0, x_{\max}]$, where $x_{\max}$ is taken as the maximum demand level (which can be calculated from the training data). This means that $\nabla U \geq 0$ on that range, *i.e.*,

$$2Qx_{\max} + 2r \geq 0, \quad r \geq 0.$$

Thus, the constraint set $\mathcal{A}$ is

$$\mathcal{A} = \{(Q, r) \mid Qx_{\max} + r \geq 0,\ r \geq 0,\ Q \preceq 0\}.$$

In this problem we know the first part of the objective function is $p^T x$, so we implicitly use the normalization $\alpha_0 = 1$, given in §III. To compute an approximate imputed objective we solve (3), taking $\phi(r_{\text{stat}}, r_{\text{comp}}) = \|r_{\text{stat}}\|_2^2 + \|r_{\text{comp}}\|_2^2$.

Numerical example: As an example, consider a problem with $n = 5$ products and $N = 200$ observations of consumer demand. We generate the prices from a uniform distribution, *i.e.*, $p_i^{(k)} \sim U[p_{\min}, p_{\max}]$, with $p_{\min} = 8$ and $p_{\max} = 12$, and calculate the demand levels by minimizing (4) with

$$U_{\text{true}}(x) = \mathbf{1}^T \sqrt{Ax^{(k)} + b},$$

where $A \in \mathbf{R}^{n \times n}_+$ and $b \in \mathbf{R}^n_+$. (Note that this 'true' utility is concave and increasing, but not quadratic.)

We solve (3) and impute a quadratic utility function with coefficients Q and r. We assess the performance of the imputed utility function on the training data as well as on non-trained test data for validation. The relative training error is 5.4% and the relative test error is 6%.

Usage: We use the imputed objective function to set prices p_{des} to achieve a certain demand level x_{des}. Since the underlying demand behavior is modeled by U_{true}, the actual realized demand level, x_{real} will be different from x_{des}. Figure 1 shows a scatter plot of the desired demand level x_{des} and the realized demand level x_{real}.

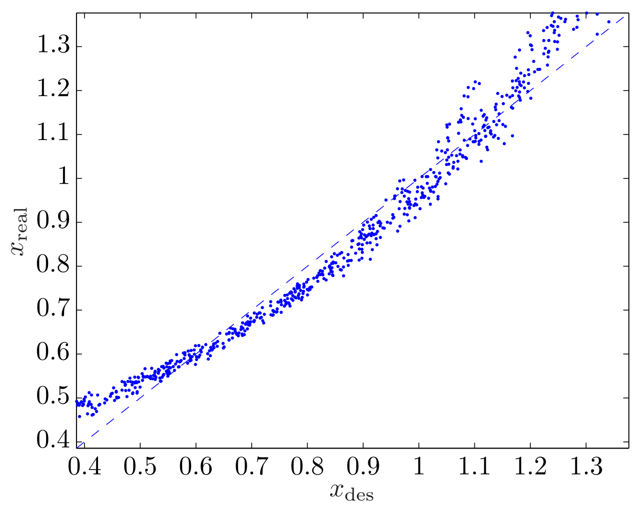

Fig. 1. Scatter plot of the desired demand level versus the realized demand level. The dashed line is the $y = x$ line.

B. Controller fitting

In this example we fit an optimizing process to samples of a control policy, to mimic the control behavior. This allows us to simplify complex control policies, such as receding horizon control (RHC) [26], or human expert control, by approximating their action with a relatively simple optimizing process.

Stochastic control: We consider a linear dynamical system,

$$x_{t+1} = Ax_t + Bu_t + w_t, \quad t = 0, 1, \ldots,$$

where $x_t \in \mathbf{R}^n$ is the state, $u_t \in \mathbf{R}^m$ is the input, and $w_t \in \mathbf{R}^n$ is the noise (or exogeneous input), at time t. We assume that $w_0, w_1, w_2, \ldots$, are zero mean IID with known distribution. The matrices $A \in \mathbf{R}^{n \times n}$ and $B \in \mathbf{R}^{n \times m}$ define the system dynamics.

We look for state feedback control policies, $u_t = \phi(x_t)$, where $\phi : \mathbf{R}^n \to \mathbf{R}^m$ is the state feedback function. For a fixed state feedback function, the state and input trajectories become stochastic processes. Our objective is

$$\limsup_{T \to \infty} \frac{1}{T} \mathbf{E} \sum_{t=1}^{T-1} \ell(x_t, u_t),$$

where $\ell : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}$ is a convex stage cost. We also have constraints on the input

$$Fu_t \leq h \ \ (\text{a.s.}), \quad t = 0, 1, \ldots,$$

where $F \in \mathbf{R}^{p \times m}$ and $h \in \mathbf{R}^p$.

The stochastic control problem is to find a control policy that minimizes the objective, among all policies that satisfy the input constraints. Here we omit technical details, such as conditions under which the expectation/limit exists. For a full technical discussion of stochastic control, see [17], [18].

Imputing a control policy: We are given samples $(u^{(k)}, x^{(k)})$ that come from running a suboptimal control policy, where $u^{(k)}$ is the input applied when the state is $x^{(k)}$. These samples can come from a policy such as receding horizon control (RHC, also known as model predictive control or MPC), or proportional-integral-derivative (PID) control, but more generally, they can be samples of an expert operator, such as an experienced pilot.

We model the action of the control policy by the optimizing process

$$\begin{array}{ll} \text{minimize} & \ell(x, u) + \mathbf{E}\, V(Ax + Bu + w_t) \\ \text{subject to} & Fu \leq h, \end{array}$$

with variable u and parameter x. Here, the function $\ell : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}$ is the given stage cost function, and $V : \mathbf{R}^n \to \mathbf{R}$ is called an approximate value function. We refer to this control policy as the approximate dynamic programming (ADP) policy [27], [18].

In this example, we look for a convex quadratic V with the form $V(z) = z^T P z$. Taking the above expectation analytically, our optimizing process simplifies to

$$\begin{array}{ll} \text{minimize} & \ell(x,u) + z^T P z \\ \text{subject to} & Fu \leq h, \quad z = Ax + Bu, \end{array} \tag{5}$$

with variables z and u, and parameter x. The imputed objective problem is to find a symmetric $P \succeq 0$ that is approximately consistent with the samples $((u^{(k)}, z^{(k)}), x^{(k)})$, $k = 1, \dots, N$, where we let $z^{(k)} = Ax^{(k)} + Bu^{(k)}$. Once we have imputed a P, we can use our optimizing process as a suboptimal controller. Thus using this method we can approximate potentially complex control behaviors with a simple optimizing process model.

Note that in this problem we implicitly use the normalization given in §III, since we know part of the objective (the stage cost function), and require $P \succeq 0$. To compute an approximate imputed objective we solve (3), taking $\phi(r_{\rm stat}, r_{\rm comp}) = \|(r_{\rm stat}, r_{\rm comp})\|_2^2$.

Numerical example: We consider a control problem with $n = 10$ states and $m = 4$ inputs, where the dynamics matrices A and B are randomly generated, with entries drawn from $\mathcal{N}(0,1)$; A is then scaled so $|\lambda_{\max}(A)| < 1$. The disturbance w_t has distribution $\mathcal{N}(0, I)$. Our stage cost is quadratic, with the form $\ell(x,u) = \|x\|_2^2 + \|u\|_2^2$, and we have box constraints $\|u\|_\infty \leq 0.1$.

To generate the data, we simulate the system under RHC (which we describe in §VI-A), for 1000 time steps. Using the states and inputs obtained in the first 100 steps we impute a quadratic approximate value function by solving (3). We then repeat the simulation (using the same disturbance trajectory), with the system running under our ADP policy. The results are presented in Figure 2, which shows histograms of stage costs for RHC (top), and ADP (bottom). The vertical lines show the average cost incurred over the 1000 step simulation. For RHC the average cost is $J_{\rm rhc} = 173.0$, for ADP the average cost is $J_{\rm adp} = 173.4$, which is almost identical. Thus, we have obtained similar control performance to RHC using an ADP policy, which requires solving a smaller optimization problem with significantly fewer variables and constraints at each time step. Indeed, when the contraints are polyhedral and the approximate value function is quadratic, the ADP policy can be computed extremely fast; see, *e.g.*, [28], [29].

C. Single commodity trans-shipment network

In this application we consider a single commodity trans-shipment network (for example, electricity), with a total of n nodes, where nodes $i = 1, \dots, p$ are producers and the remaining nodes $i = p+1, \dots, n$ are consumers. There are m edge between the nodes, and we are given the edge-node incidence matrix $R \in \mathbf{R}^{n \times m}$:

$$R(i,j) = \begin{cases} 1 & \text{if node } i \text{ is the starting node for edge } j \\ -1 & \text{if node } i \text{ is the ending node for edge } j \\ 0 & \text{otherwise.} \end{cases}$$

We assume that the edges have a maximum capacity of $f^{\max}$; furthermore, we also assume that the network obeys

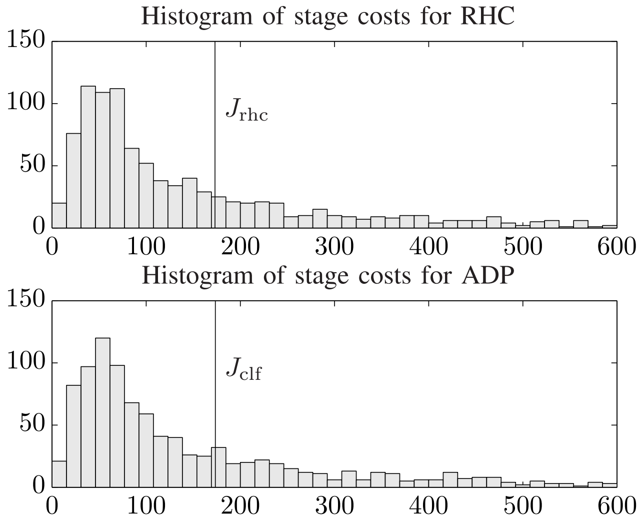

Fig. 2. Histogram of stage costs for RHC (top) and ADP (bottom). Vertical lines indicate the average cost incurred over the 1000 step simulation.

conservation of flows, *i.e.*, for a given flow on the network $f \in \mathbf{R}^m$, production level $y \in \mathbf{R}^p$ and demand level $d \in \mathbf{R}^{n-p}$, we have $Rf = (y, d)$.

A negative value for s_i means that there was an outflow at node i (which would be common in the demand nodes), and a positive s_i means that there is an injection at node i (which would be common in the producer nodes). Each producer $i = 1, \dots, p$ can produce a maximum of $y_i^{\max}$ units of commodity, and incurs a cost of $C^{(i)}(y_i)$ for producing y_i units. We also assume that there is a linear transportation cost $w_i f_i$ associated with each edge $i = 1, \dots, m$. Finally, we can write the underlying optimizing process as an optimization problem with parameter d and variable y:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{p} C^{(i)}(y_i) + \sum_{i=1}^{m} w_i f_i \\ \text{subject to} & 0 \leq y \leq y^{\max}, \\ & 0 \leq f \leq f^{\max}, \\ & Rf = \begin{bmatrix} y \\ d \end{bmatrix}. \end{array} \tag{6}$$

Imputed objective problem: We are given a set of N different demand scenarios $d^{(k)}$, the corresponding flows $f^{(k)}$ and the producer levels $y^{(k)}$, based on historical operation of the network. The variable in this problem is $x = (y, f)$ and the parameter is d. For this example, we will focus on a polynomial representation for the producer cost functions, so the objective can be written as

$$\sum_{i=1}^{p} C^{(i)}(y_i) + \sum_{i=1}^{m} w_i f_i = \sum_{i=1}^{p} \sum_{j=1}^{K} A_{i,j} y_i^j + \sum_{i=1}^{m} w_i f_i,$$

where y_i^j, $i = 1, \dots, p$, $j = 1, \dots, K$ are the basis functions for the production level y and f_i, $i = 1, \dots, m$ are basis functions for the flows f, and

$$\mathcal{A} = \{(A, w) \mid A \geq 0,\ w \geq 0,\ \mathbf{1}^T(w + A\mathbf{1}) = 1\}.$$

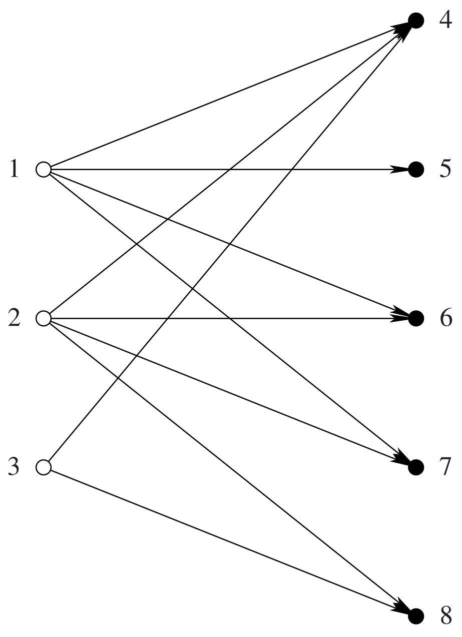

Fig. 3. Example of a single commodity network.

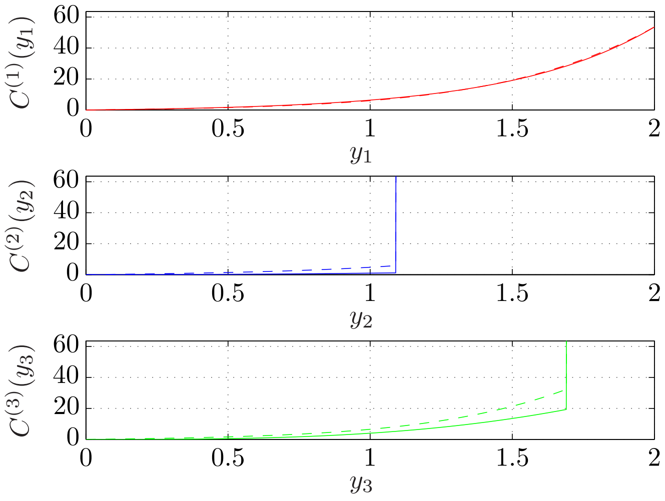

Fig. 4. Imputed cost functions (dashed) and the true cost functions (solid) for each production node.

Note that unlike the examples in §IV-A and §IV-B, in this example we do not have access to a part of the objective function f_0. Since we are not interested in a trivial objective function, we normalize the weights by requiring $\mathbf{1}^T(w + A\mathbf{1}) = 1$. Since the objective is homogenous in the variables, this normalization will result in the same solution up to a multiplicative factor.

To impute the objective function, we solve (3), taking $\phi(r_{\text{stat}}, r_{\text{comp}}) = \|r_{\text{stat}}\|_2^2 + \|r_{\text{comp}}\|_2^2$.

Numerical example: As an example, consider a problem instance with $n = 8$ nodes, $p = 3$ producers and $N = 240$ observations of network flows and production levels and $K = 5$ polynomials for basis functions. The consumers and producers are connected according to the graph in figure 3. For each consumer, we generate negative demand levels randomly from a uniform distribution $U[-1, 0]$. The production levels and edge flows are generated using the cost functions

$$C^{(1)}_{\text{true}}(y_1) = e^y - 1,$$
$$C^{(2)}_{\text{true}}(y_2) = y^2,$$
$$C^{(3)}_{\text{true}}(y_3) = 2y^3.$$

The edge weights are chosen randomly from a $U[2.5, 7.5]$ distribution, and the maximum edge capacity is set to $f^{\max} = 1.2$ for all edges. We use different production limits for each producer, $y^{\max} = (2, 1.1, 1.7)$.

As described above, we obtain a polynomial imputed objective function. The true cost functions (used to generate the data) and the imputed cost functions are plotted in figure 4. (The imputed cost functions are scaled to be on the same scale as the true cost functions for illustration purposes). To assess the performance of the imputed utility function, we used it to generate production level and edge flows on the training data as well as non-trained test data for cross validation. The average relative error in production level is around 7.8% for the training data and 9.6% for the test data.

Usage: Once we have imputed an objective function, we can use it to predict the production level and edge flows for a given demand level. Furthermore, the dual variable ν associated with the equality constraint $Rf = (y, d)$ provides a set of shadow prices. The values of $\nu_1, \ldots, \nu_p$ correspond to production prices and the values of $\nu_{p+1}, \ldots, \nu_n$ provide a set of consumer prices. Thus, for a given demand level, we can solve the convex optimization problem (6) and dynamically adjust the consumer prices by using $\nu_{p+1}, \ldots, \nu_n$. As a numerical example, consider the network in figure 3. For a demand level of $d = (1, 0.1, 0.2, 0.3, 3)$, we get $\nu_8 = 0.35$, while ν_i is on the order of 0.01 or smaller for all other demand nodes. This means that the consumer at node 8 should be charged a much higher price than other customers, since it is only connected to the producers that have a small maximum production capacity, and it is putting a very large demand on the network.

V. Conclusion

The problem of imputing an objective function is a practical problem that arises frequently, and is closely related to problems in machine learning, dynamic programming, robotics, and economics. In this paper we presented a framework for imputing a convex objective function, based on observations of the underlying optimizing process. We develop this framework by noting that if the data is optimal with respect to the imputed objective, the primal and dual variables must satisfy the KKT conditions; since the dual variables and the objective parameters enter these conditions linearly, we have a convex optimization problem. In the presence of noise or modeling errors, we minimize the residuals of the KKT conditions under a suitable regularization function.

We show the application of this method to examples in consumer behavior, control, and single commodity networks. We observe that the method works very well in these examples. Furthermore, we note that we can use this framework to reduce the complexity of a controller by observing solutions derived from a more complicated controller.

VI. Appendix

A. Receding horizon control

Here we describe RHC; for more details see, *e.g.*, [26]. RHC works as follows. At each time t we consider a time period extending T steps into the future $t, t+1, \ldots, t+T$. We solve a planning problem

$$\begin{array}{ll} \text{minimize} & \sum_{\tau=t}^{t+T-1} \ell(z_\tau, v_\tau) \\ \text{subject to} & z_{\tau+1} = Az_\tau + Bv_\tau, \quad \tau = t, \ldots, t+T-1 \\ & Fv_\tau \leq h, \quad \tau = 0, \ldots, t+T-1 \\ & z_t = x_t, \end{array}$$

with variables $z_t, \ldots, z_{t+T}$, $v_t, \ldots, v_{t+T-1}$. We let $z_t^\star, \ldots, z_{t+T}^\star$, $v_t^\star, \ldots, v_{t+T-1}^\star$ denote an optimal point; the RHC policy takes $u_t = v_t^\star$. The planning process is repeated at the next time step, using new information that have become available (in this case, the true value of x_{t+1}).

Receding horizon control has been successfully applied to many control problems, but one disadvantage is that RHC is computationally intensive, since an optimization problem (with a relatively large number of variables) must be solved at every time step. On the other hand, the ADP policy requires solving a relatively small optimization problem at each time step.

VII. Acknowledgements

This material is based upon work supported by JPL contract 1400723, by AFOSR grant FA9550-09-0130, by NASA grant NNX07AEIIA, by AFOSR grant FA9550-06-1-0312, by a National Semiconductor Corporation Stanford Graduate Fellowship, and by a Rambus Corporation Stanford Graduate Fellowship.

References

[1] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," in *Proceedings of the 17th IFAC world congress*, 2008, pp. 6974–6997.

[2] D. Burton, W. Pulleyblank, and P. Toint, "The inverse shortest path problem with upper bounds on shortest path costs," in *Network Optimization*, ser. Lecture notes in economics and mathematical systems, vol. 450. Springer, 1997, pp. 156–171.

[3] D. Burton and P. Toint, "On an instance of the inverse shortest path problem," *Mathematical Programming*, vol. 53, pp. 45–61, 1992.

[4] T. Nielsen and F. Jensen, "Learning a decision maker's utility function from (possibly) inconsistent behavior," *Artificial Intelligence*, vol. 160, pp. 53–78, 2004.

[5] D. Ackerberg, C. Benkard, S. Berry, and A. Pakes, "Econometric tools for analyzing market outcomes," in *Handbook of Econometrics*, ser. Handbook of Econometrics, J. Heckman and E. Leamer, Eds. Elsevier, 2007, vol. 6, ch. 63.

[6] P. Bajari, C. Benkard, and J. Levin, "Estimating dynamic models of imperfect competition," *Econometrica*, vol. 75, no. 5, pp. 1331–1370, 2007.

[7] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *In Proceedings of 17th International Conference on Machine Learning*. Morgan Kauffman, 2000, pp. 663–670.

[8] P. Abbeel and A. Ng, "Apprenticeship learning via inverse reinforcement learning," in *In Proceedings of 21st International Conference on Machine Learning*, ser. ICML '04. ACM, 2004.

[9] W. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics, 2007.

[10] U. Syed and R. Schapire, "A game-theoretic approach to apprenticeship learning," in *Advances in Neural Information Processing Systems 20*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds. Cambridge, MA: MIT Press, 2008, pp. 1449–1456.

[11] G. Neu and C. Szepesvari, "Apprenticeship learning using reinforcement learning and gradient methods," in *Proceedings of the 23rd Annual Conference on Uncertainty in Artificial Intelligence (UAI-07)*, 2007, pp. 295–302.

[12] N. Ratliff, J. Bagnell, and M. Zinkevich, "Maximum margin planning," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[13] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 2586–2591.

[14] P. Abbeel, A. Coates, and A. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *The International Journal of Robotics Research*, vol. 29, 2010.

[15] R. E. Kalman, "When is a linear control system optimal?" *Transactions of the ASME, Journal of Basic Engineering*, vol. 86, pp. 51–60, 1964.

[16] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in Systems and Control Theory*. Philadelphia: SIAM books, 1994.

[17] D. Bertsekas, *Dynamic Programming and Optimal Control: Volume 1*. Athena Scientific, 2005.

[18] ——, *Dynamic Programming and Optimal Control: Volume 2*. Athena Scientific, 2007.

[19] R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[20] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[21] D. De Farias and B. Van Roy, "The linear programming approach to approximate dynamic programming," *Operations Research*, vol. 51, no. 6, pp. 850–865, 2003.

[22] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Cambridge University, Cambridge, England, 1989.

[23] J. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, no. 279-292, 1992.

[24] V. Desai, V. Farias, and C. Moallemi, "Approximate dynamic programming via a smoothed linear program," 2010, manuscript.

[25] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[26] J. M. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[27] W. Powell, *Approximate dynamic programming: solving the curses of dimensionality*. John Wiley & Sons, Inc., 2007.

[28] Y. Wang and S. Boyd, "Fast evaluation of quadratic control-lyapunov policy," *IEEE Transactions on control Systems Technology*, vol. PP, no. 99, pp. 1–8, 2010.

[29] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, Jan. 2002.