TU Delft

Delft University of Technology

Fast Approximate Dynamic Programming for Input-Affine Dynamics

Sharifi Kolarijani, M.A.; Esfahani, P. M.

DOI
10.1109/TAC.2022.3232637

Publication date
2023

Document Version
Final published version

Published in
IEEE Transactions on Automatic Control

Citation (APA)
Sharifi Kolarijani, M. A., & Esfahani, P. M. (2023). Fast Approximate Dynamic Programming for Input-Affine Dynamics. *IEEE Transactions on Automatic Control*, *68*(10), 6315-6322. https://doi.org/10.1109/TAC.2022.3232637

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

https://www.openaccess.nl/en/you-share-we-take-care

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

Fast Approximate Dynamic Programming for Input-Affine Dynamics

Mohamad Amin Sharifi Kolarijani and Peyman Mohajerin Esfahani

***Abstract*—We propose two novel numerical schemes for the approximate implementation of the dynamic programming (DP) operation concerned with finite-horizon optimal control of discrete-time systems with input-affine dynamics. The proposed algorithms involve discretization of the state and input spaces and are based on an alternative path that solves the dual problem corresponding to the DP operation. We provide error bounds for the proposed algorithms, along with a detailed analysis of their computational complexity. In particular, for a specific class of problems with separable data in the state and input variables, the proposed approach can reduce the typical time complexity of the DP operation from $\mathcal{O}(XU)$ to $\mathcal{O}(X+U)$, where X and U denote the size of the discrete state and input spaces, respectively. This reduction in complexity is achieved by an algorithmic transformation of the minimization in DP operation to an addition via discrete conjugation.**

***Index Terms*—Computational complexity, conjugate duality, dynamic programming, input-affine dynamics.**

I. Introduction

Dynamic programming (DP) is one of the most common tools used for tackling sequential decision problems with applications in, e.g., optimal control, operation research, and reinforcement learning. The basic idea of DP is to solve the Bellman equation

$$J_t(x_t) = \min_{u_t} \{C(x_t, u_t) + J_{t+1}(x_{t+1})\} \tag{1}$$

backward in time t for the costs-to-go J_t, where $C(x_t, u_t)$ is the cost of taking the control action u_t at the state x_t (value iteration). Arguably, the most important drawback of DP is its high computational cost in solving problems with a large scale *finite* state space. For problems with a *continuous* state space, which is commonly the case in engineering applications, solving the Bellman equation requires solving an infinite number of optimization problems. This usually renders the exact implementation of the DP operation impossible, except for a few cases with an available closed-form solution, e.g., linear quadratic regulator [1, Sec. 4.1]. To address this issue, various schemes have been introduced, commonly known as *approximate* dynamic programming; see, e.g., [2], [3]. A common scheme is to use a sample-based approach accompanied by some form of function approximation. This usually amounts to deploying a brute force search over a discretization/abstraction of the state and input spaces, leading to a time complexity of at least $\mathcal{O}(XU)$, where X and U are the cardinalities of the discrete state and input spaces, respectively.

Manuscript received 16 March 2022; revised 9 September 2022; accepted 14 December 2022. Date of publication 28 December 2022; date of current version 27 September 2023. This research is part of a project that has received funding from the European Research Council (ERC) under the Grant TRUST-949796. Recommended by Associate Editor S.-J. Chung. *(Corresponding author: Mohamad Amin Sharifi Kolarijani.)*

The authors are with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: m.a.sharifikolarijani@tudelft.nl; p.mohajerinesfahani@tudelft.nl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2022.3232637.

Digital Object Identifier 10.1109/TAC.2022.3232637

For some DP problems, it is possible to reduce this complexity by using duality, i.e., approaching the minimization problem in (1) in the conjugate domain. For instance, for the deterministic linear dynamics $x_{t+1} = Ax_t + Bu_t$ with the separable cost $C(x_t, u_t) = C_{\mathrm{s}}(x_t) + C_{\mathrm{i}}(u_t)$, we have

$$J_t(x_t) \geq C_{\mathrm{s}}(x_t) + \left[C_{\mathrm{i}}^*(-B^\top \cdot) + J_{t+1}^*\right]^*(Ax_t) \tag{2}$$

where the operator $[\cdot]^*$ denotes the Legendre–Fenchel transform, also known as the (convex) conjugate transform. Under some technical assumptions (including, among others, convexity of the functions C_{i} and J_{t+1}), we have equality in (2); see [4, Prop. 5.3.1]. Notice how the minimization operator in (1) transforms to a simple addition in (2). This observation signals the possibility of a significant reduction in the time complexity of solving the Bellman equation, at least for particular classes of DP problems.

Approaching the DP problem through the lens of the conjugate duality goes back to Bellman [5]. Further applications of this idea for complexity reduction were later explored in [6], [7]. Fundamentally, these approaches exploit the operational duality of *infimal convolution* and *addition* with respect to (w.r.t.) the conjugate transform $[\cdot]^*$ [8]: For two functions $f_1, f_2 : \mathbb{R}^n \to [-\infty, +\infty]$, we have $(f_1 \square f_2)^* = f_1^* + f_2^*$, where

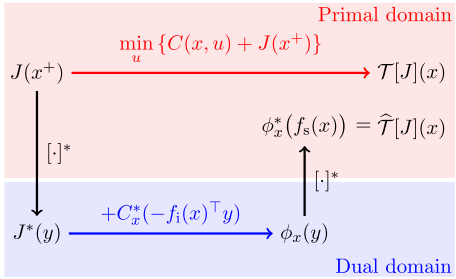
$$f_1 \square f_2(w) := \inf\{f_1(w_1) + f_2(w_2) : w_1 + w_2 = w\} \tag{3}$$

is the infimal convolution of f_1 and f_2. This is analogous to the well-known operational duality of convolution and multiplication w.r.t. the Fourier transform. Actually, the conjugate transform plays a similar role as the Fourier transform when the underlying algebra is the *max-plus* algebra, as opposed to the conventional plus-times algebra. Much like the extensive application of the latter operational duality upon the introduction of the fast Fourier transform, "fast" numerical algorithms for (discrete) conjugate transform can facilitate efficient applications of the former one. Fortunately, such numerical algorithms for computing conjugate functions are already available [9], [10], [11]. One of the first and most widespread applications of these fast algorithms is in solving the Hamilton–Jacobi equation [9], [12], [13]. Another interesting area of application is image processing, where the Legendre–Fenchel transform is commonly known as "distance transform" [14], [15]. However, surprisingly, the application of these fast algorithms in solving discrete-time optimal control problems seems to remain largely unexplored. An exception is [16], where the authors propose the "fast value iteration" algorithm for computing the fixed-point of the Bellman operator arising from a specific class of infinite-horizon, discrete-time DP problems (with state-independent stage cost $C(x, u) = C(u)$ and linear dynamics $x^+ = Ax + Bu$, where A is invertible and monotone).

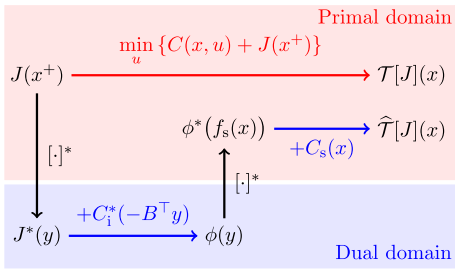
Another related line of work involves algorithms that utilize max-plus algebra in solving continuous-time, continuous-space, deterministic optimal control problems; see, e.g., [17], [18], [19]. These works exploit the compatibility of the Bellman operation with max-plus operations and approximate the value function as a max-plus linear combination. In particular, in [20], the authors use this idea to propose an approximate value iteration algorithm for deterministic, continuous-state Markov decision processes. In this regard, we note

0018-9286 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.



(a)



(b)

Fig. 1. Sketch of the proposed algorithms – the standard DP operation in the primal domain (upper red paths) and the CDP operation through the dual domain (bottom blue paths): (a) Setting 1 with dynamics $x^+ = f_s(x) + f_i(x)\cdot u$ and generic cost $C(x,u)$. (b) Setting 2 with dynamics $x^+ = f_s(x) + B\cdot u$ and separable cost $C(x,u) = C_s(x) + C_i(u)$.

that the proposed algorithms in the current study also implicitly involve representing cost functions as max-plus linear combinations.

In this study, we consider the approximate implementation of the DP operation arising in the finite-horizon optimal control of discrete-time systems with continuous state and input spaces. The proposed approach involves discretization of the state space and is based on an alternative path that solves the dual problem corresponding to the DP operation by utilizing the *Linear-time* Legendre transform (LLT) algorithm [11] for discrete conjugation. The **main contributions** of this work are as follows:

1) *From minimization to addition:* For *input-affine* dynamics, we propose an algorithm that transforms the minimization in the DP operation to a simple addition at the expense of three conjugate transforms; see Fig. 1(a). This leads to transferring the computational cost from the input domain $\mathbb{U}$ to the dual state domain $\mathbb{Y}$ (Theorem 3.4).
2) *From quadratic to linear complexity:* For problems with *separable* data in the state and input variables, we propose another algorithm that further reduces the complexity from quadratic to linear; see Fig. 1(b). To be precise, for this class, the time complexity of solving the optimal control problem reduces to $\mathcal{O}(X+U)$, compared to the standard $\mathcal{O}(XU)$ (Theorem 3.9).
3) *Construction of dual domain:* We provide specific guidelines for construction of a *dynamic* discrete dual space based on our error analysis (Theorems 3.6 and 3.10).
4) *Towards quantum DP:* The proposed algorithms are developed such that any reduction in the complexity of conjugate transform immediately translates to a reduced computational cost for these algorithms. Particularly, motivated by the recent quantum speedup for discrete conjugate transform [21], we envision that the proposed method paves the way for developing a quantum DP algorithm.
5) *Software package:* We provide the discrete CDP (d-CDP) MATLAB package [22] for the algorithms presented in this study.

We note that the extended version of this article is available in [23].

Notations: We use $\mathbb{R}$ to denote the real line and $\overline{\mathbb{R}} = \mathbb{R}\cup\{+\infty\}$ to denote its extension. $\|\cdot\|$ and $\langle\cdot,\cdot\rangle$ denote the two-norm and the dot product in $\mathbb{R}^n$, respectively. For $A\in\mathbb{R}^{m\times n}$, $A^\top$ denotes its transpose, and $\|A\| = \sup_x\{\|Ax\| : \|x\| = 1\}$ denotes it induced two-norm. We use the common convention in optimization whereby the optimal value of an infeasible minimization (resp. maximization) problem is set to $+\infty$ (resp. $-\infty$). Continuous sets (infinite, uncountable) are denoted as $\mathbb{X},\mathbb{Y},\ldots$. For *discrete* (finite) sets, we use the superscript d as in $\mathbb{X}^\mathrm{d},\mathbb{Y}^\mathrm{d},\ldots$. Moreover, we use the superscript g to differentiate *grid-like* (i.e., factorized) finite sets as in $\mathbb{X}^\mathrm{g} = \Pi_{i=1}^n \mathbb{X}_i^\mathrm{g} = \mathbb{X}_1^\mathrm{g}\times\ldots\times\mathbb{X}_n^\mathrm{g}\subset\mathbb{R}^n$, where $\mathbb{X}_i^\mathrm{g}$ is a finite subset of $\mathbb{R}$. The cardinality of a finite set $\mathbb{X}^\mathrm{d}$ (or $\mathbb{X}^\mathrm{g}$) is denoted by X. Let $\mathbb{X},\mathbb{Y}$ be two arbitrary sets in $\mathbb{R}^n$. The convex hull of $\mathbb{X}$ is denoted by $\mathrm{co}(\mathbb{X})$. The diameter of $\mathbb{X}$ is defined as $\Delta_\mathbb{X} := \sup_{x,y\in\mathbb{X}}\|x-y\|$. We use $\mathrm{d}(\mathbb{X},\mathbb{Y}) := \inf_{x\in\mathbb{X},y\in\mathbb{Y}}\|x-y\|$ to denote the distance between $\mathbb{X}$ and $\mathbb{Y}$. The one-sided Hausdorff distance *from* $\mathbb{X}$ *to* $\mathbb{Y}$ is defined as $\mathrm{d_H}(\mathbb{X},\mathbb{Y}) := \sup_{x\in\mathbb{X}}\inf_{y\in\mathbb{Y}}\|x-y\|$. Let $h:\mathbb{R}^n\to\overline{\mathbb{R}}$ be an extended real-valued function with a nonempty effective domain $\mathrm{dom}(h) = \mathbb{X} := \{x\in\mathbb{R}^n : h(x) < +\infty\}$. We use $h^\mathrm{d}:\mathbb{X}^\mathrm{d}\to\overline{\mathbb{R}}$ to denote the discretization of h, where $\mathbb{X}^\mathrm{d}$ is a finite subset of $\mathbb{R}^n$. Whether a function is discrete is usually also clarified by providing its domain explicitly. We particularly use this notation in combination with a second operation to emphasize that the second operation is applied on the discretized version of the operand; e.g., we use $\widetilde{h^\mathrm{d}}:\mathbb{R}^n\to\overline{\mathbb{R}}$ to denote the extension of the discretization h^d. If the domain $\mathbb{X}^\mathrm{d} = \mathbb{X}^\mathrm{g}$ of h^d is grid-like, we then use $\widehat{h^\mathrm{d}}$ (as opposed to $\widetilde{h^\mathrm{d}}$) for the extension using *multilinear interpolation and extrapolation (LERP)*; see, e.g., [24, Appendix D]. The Lipschitz constant of h over a set $\mathbb{Y}\subset\mathbb{X}$ is denoted by $\mathrm{L}(h;\mathbb{Y}) := \sup_{x,y\in\mathbb{Y}}|h(x)-h(y)|/\|x-y\|$. We also denote $\mathrm{L}(h) := \mathrm{L}(h;\mathbb{X})$. The subdifferential of h at a point $x\in\mathbb{X}$ is defined as $\partial h(x) := \{y\in\mathbb{R}^n : h(\tilde{x})\geq h(x)+\langle y,\tilde{x}-x\rangle, \forall\tilde{x}\in\mathbb{X}\}$. The Legendre–Fenchel transform (convex conjugate) of h is the function $h^*:\mathbb{R}^n\to\overline{\mathbb{R}}$, defined by $h^*(y) = \sup_x\{\langle y,x\rangle - h(x)\}$. We note that the conjugate function h^* is convex by construction. We again use the notation $h^{\mathrm{d}*}$ to emphasize the fact that the domain of the underlying function is finite, i.e., $h^{\mathrm{d}*}(y) = \sup_{x\in\mathbb{X}^\mathrm{d}}\{\langle y,x\rangle - h^\mathrm{d}(x)\}$. The biconjugate and discrete biconjugate operators are defined accordingly and denoted by $[\cdot]^{**} = [[\cdot]^*]^*$ and $[\cdot]^{\mathrm{d}*\mathrm{d}*} = [[\cdot]^{\mathrm{d}*}]^{\mathrm{d}*}$, respectively. We report the complexities using the standard big O notations $\mathcal{O}$ and $\widetilde{\mathcal{O}}$, where the latter hides the logarithmic factors. In this study, we are mainly concerned with the dependence of the computational complexities on *the size of the finite sets* involved (discretization of the primal and dual domains). In particular, we ignore the possible dependence of the computational complexities on the dimension of the variables, unless they appear in the power of the size of those discrete sets; e.g., the complexity of a single evaluation of an analytically available function is taken to be of $\mathcal{O}(1)$, regardless of the dimension of its input and output arguments.

II. Problem Statement and Standard Solution

We consider the optimal control of discrete-time systems

$$x_{t+1} = f(x_t,u_t),\quad t=0,\ldots,T-1 \tag{4}$$

where $f:\mathbb{R}^n\times\mathbb{R}^m\to\mathbb{R}^n$ describes the dynamics, and $T\in\mathbb{N}$ is the finite horizon. In this study, we focus on deterministic dynamics; we refer the reader to [23, Appendix C] for the extension of the proposed algorithms for stochastic dynamics with additive noise. We also consider state and input constraints

$$\begin{cases} x_t\in\mathbb{X}\subset\mathbb{R}^n & \text{for}\quad t\in\{0,\ldots,T\}\\ u_t\in\mathbb{U}\subset\mathbb{R}^m & \text{for}\quad t\in\{0,\ldots,T-1\}.\end{cases} \tag{5}$$

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

Let $C : \mathbb{X} \times \mathbb{U} \to \overline{\mathbb{R}}$ and $C_T : \mathbb{X} \to \mathbb{R}$ be the stage and terminal cost functions, respectively. In particular, notice that we let the stage cost C take $+\infty$ for $(x, u) \in \mathbb{X} \times \mathbb{U}$ so that it can embed the *state-dependent input constraints*. For an initial state $x_0 \in \mathbb{X}$, the cost incurred by the state trajectory $\mathbf{x} = (x_0, \dots, x_T)$ in response to the input sequence $\mathbf{u} = (u_0, \dots, u_{T-1})$ is

$$J(x_0, \mathbf{u}) = \sum_{t=0}^{T-1} C(x_t, u_t) + C_T(x_T).$$

The problem of interest is then to find the optimal control sequence $\mathbf{u}^\star(x_0)$, i.e., a solution to the problem

$$J^\star(x_0) = \min_{\mathbf{u}} \{J(x_0, \mathbf{u}) : (4)\&(5)\}. \tag{6}$$

Throughout this study, we assume that the problem data satisfy the following conditions.

Assumption 2.1 (Problem data): the dynamics $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is locally Lipschitz continuous. The constraint sets $\mathbb{X}$ and $\mathbb{U}$ are compact. The set of admissible inputs $\mathbb{U}(x) := \{u \in \mathbb{U} : C(x, u) < +\infty,\ f(x, u) \in \mathbb{X}\}$ is nonempty for all $x \in \mathbb{X}$. The stage cost $C : \mathbb{X} \times \mathbb{U} \to \overline{\mathbb{R}}$ has a compact effective domain. Moreover, C and C_T are Lipschitz continuous.

The properties laid out in Assumption 2.1 imply that the set $\mathbb{U}(x)$ of admissible inputs is nonempty and compact (because of compactness of $\mathrm{dom}(C)$ and $\mathbb{X}$, and continuity of f), and the objective in (6) is continuous. Hence, the optimal value in (6) is achieved.

The DP algorithm solves (6) by solving the Bellman equation

$$J_t(x_t) = \min_{u_t} \{C(x_t, u_t) + J_{t+1}(x_{t+1}) : (4)\ \&\ (5)\}$$

for each $x_t \in \mathbb{X}$, backward in time $t = T-1, \dots, 0$, initialized by $J_T = C_T$. The iteration finally outputs $J_0 = J^\star$ [1, Prop. 1.3.1]. In order to simplify the exposition, let us embed the state and input constraints in the cost functions (C and J_t) by extending them to $+\infty$ outside their effective domain. Let us also drop the time subscript t and focus on a single step of the recursion by defining the DP operator

$$\mathcal{T}[J](x) := \min_u \{C(x, u) + J(f(x, u))\}, \quad x \in \mathbb{X} \tag{7}$$

so that $J_t = \mathcal{T}[J_{t+1}] = \mathcal{T}^{(T-t)}[J_T]$ for $t = T-1, \dots, 0$.

Notice that the DP operation (7) requires solving an infinite number of optimization problems for the continuous state space $\mathbb{X}$. Except for a few cases with an available closed-form solution, the exact implementation of DP operation is impossible. A standard approximation scheme uses a sample-based approach with a function approximation technique: In each iteration, we solve (7) for a finite number of points $x \in \mathbb{X}^{\mathrm{g}}$, for a *grid-like* discretization $\mathbb{X}^{\mathrm{g}}$ of the state space, to derive $[\mathcal{T}[J]]^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$ as the output of the current iteration, and then use these data points to form an extension $\widetilde{[\mathcal{T}[J]]^{\mathrm{d}}} : \mathbb{X} \to \mathbb{R}$ to be used for the next iteration. The extension can be realized, e.g., as a parametric approximation, where the parameters are determined via regression using the data points of the sampled state space.

Next to be addressed is the issue of solving the minimization

$$\min_{u \in \mathbb{U}} \left\{C(x, u) + \widetilde{J^{\mathrm{d}}}(f(x, u))\right\}$$

for each $x \in \mathbb{X}^{\mathrm{g}}$, where the next step cost-to-go is approximated by the extension $\widetilde{J^{\mathrm{d}}}$. Often, this problem is a difficult, nonconvex minimization problem. Here, again, a common approximation involves enumeration over a proper discretization $\mathbb{U}^{\mathrm{d}} \subset \mathbb{U}$ of the input space. We assume that the joint discretization of the state-input space is such that $\mathbb{U}^{\mathrm{d}}(x) := \mathbb{U}(x) \cap \mathbb{U}^{\mathrm{d}}$ is nonempty for all $x \in \mathbb{X}^{\mathrm{g}}$.

These approximations introduce an error which, under some regularity assumptions, depends on the discretization of the state and input spaces and the extension operation; see [23, Prop. A.1]. Incorporating these approximations, we can introduce the *discrete* DP (d-DP) operator, corresponding to a generic sample-based value iteration scheme described above, as follows:

$$\mathcal{T}^{\mathrm{d}}[J^{\mathrm{d}}](x) := \min_{u \in \mathbb{U}^{\mathrm{d}}} \left\{C(x, u) + \widetilde{J^{\mathrm{d}}}(f(x, u))\right\}, \ x \in \mathbb{X}^{\mathrm{g}}. \tag{8}$$

This generic d-DP operator/algorithm will be our benchmark for evaluating the performance of the alternative algorithms developed in this study. To this end, we discuss the time complexity of the d-DP operation in the following remark.

Remark 2.2 (Complexity of d-DP): Let the time complexity of a single evaluation of the extension operator $\widetilde{[\cdot]}$ in (8) be of $\mathcal{O}(E)$. Then, the time complexity of the d-DP operation (8) is of $\mathcal{O}(XUE)$.

III. Alternative Solution in Dual Domain

In this section, we identify two classes of problems that allow us to effectively employ conjugate duality for the DP problem. For these classes of problems, we propose an alternative path that solves a discretized version of the corresponding dual problem.

A. From Minimization to Addition

We now show that the linearity of dynamics in the input is the key property in developing the alternative solution. In particular, we propose an algorithm that transforms the minimization in the primal domain into an addition in the dual domain at the expense of three conjugate transforms for input-affine dynamics:

Setting 1: The dynamics is input-affine, i.e., $f(x, u) = f_{\mathrm{s}}(x) + f_{\mathrm{i}}(x) \cdot u$, where $f_{\mathrm{s}} : \mathbb{R}^n \to \mathbb{R}^n$ and $f_{\mathrm{i}} : \mathbb{R}^n \to \mathbb{R}^{n \times m}$.

1) d-CDP Algorithm: Alternatively, we can approach the optimization problem in the DP operation (7) in the dual domain. Let us fix $x \in \mathbb{X}$, and consider the following reformulation of (7):

$$\mathcal{T}[J](x) = \min_{u,z} \{C(x, u) + J(z) : z = f(x, u)\}. \tag{9}$$

Notice how for input-affine dynamics of Setting 1, the formulation (9) resembles the infimal convolution (3). In this regard, consider the corresponding dual problem

$$\widehat{\mathcal{T}}[J](x) := \max_y \min_{u,z} \{C(x, u) + J(z) + \langle y, f(x, u) - z\rangle\} \tag{10}$$

where $y \in \mathbb{R}^n$ is the dual variable. For input-affine dynamics, we can derive the following equivalent reformulation of (10):

$$\begin{aligned}&\widehat{\mathcal{T}}[J](x)\\ &= \max_y \left\{\langle y, f_{\mathrm{s}}(x)\rangle - \max_u \left[\langle -f_{\mathrm{i}}(x)^\top y, u\rangle - C(x, u)\right] - J^*(y)\right\}\end{aligned}$$

where $J^*(y) = \max_z \langle y, z\rangle - J(z)$. Now, let us define the partial conjugate of the stage cost w.r.t. the input variable u as follows:

$$C_x^*(v) := \max_u \{\langle v, u\rangle - C(x, u)\}, \quad v \in \mathbb{R}^m \tag{11}$$

so that

$$\widehat{\mathcal{T}}[J](x) = \max_y \{\langle y, f_{\mathrm{s}}(x)\rangle - C_x^*(-f_{\mathrm{i}}(x)^\top y) - J^*(y)\}.$$

Equivalently, we have

$$\phi_x(y) := C_x^*(-f_{\mathrm{i}}(x)^\top y) + J^*(y), \qquad y \in \mathbb{R}^n \tag{12a}$$

$$\widehat{\mathcal{T}}[J](x) = \phi_x^*(f_{\mathrm{s}}(x)), \qquad x \in \mathbb{X}. \tag{12b}$$

The construction above suggests an alternative path for computing the output of the DP operator through the conjugate domain. We call this alternative approach *conjugate* DP (CDP), and the corresponding

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: For d-CDP Operator (13) and Setting 1.

Input: dynamics $f_{\mathrm{s}} : \mathbb{R}^n \to \mathbb{R}^n$, $f_{\mathrm{i}} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$;
discrete cost-to-go (at $t+1$) $J^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$;
conjugate of stage cost $C_x^* : \mathbb{R}^m \to \mathbb{R}$ for $x \in \mathbb{X}^{\mathrm{g}}$;
grid $\mathbb{Y}^{\mathrm{g}} \subset \mathbb{R}^n$.
Output: discrete cost-to-go (at t) $\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$.
use LLT to compute $J^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ from $J^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$;
for each $x \in \mathbb{X}^{\mathrm{g}}$ **do**
  $\psi_x^{\mathrm{d}}(y) \leftarrow C_x^*(-f_{\mathrm{i}}(x)^\top y) + J^{\mathrm{d}*\mathrm{d}}(y)$ for $y \in \mathbb{Y}^{\mathrm{g}}$;
  $\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \leftarrow \max_{y\in\mathbb{Y}^{\mathrm{g}}}\{\langle f_{\mathrm{s}}(x), y\rangle - \psi_x^{\mathrm{d}}(y)\}$.
end for

operator (12) the CDP operator. Fig. 1(a) characterizes this alternative path schematically. The numerical implementation of CDP operation requires the computation of conjugate functions. In particular, CDP operation involves three conjugate transforms. In this article, we assume that the partial conjugate C_x^* of the stage cost in (11) is analytically available. We note however that one can also consider a numerical scheme to approximate this conjugation; see [23, Appendix C] for further details.

Assumption 3.1 (Conjugate of stage cost): The conjugate function C_x^* (11) is analytically available. That is, the time complexity of evaluating $C_x^*(v)$ for each $v \in \mathbb{R}^m$ is of $\mathcal{O}(1)$.

The two remaining conjugate operations of the CDP path in Fig. 1(a) are handled numerically. To be precise, we again take a sample-based approach and compute $\widehat{\mathcal{T}}[J]$ for a finite number of states $x \in \mathbb{X}^{\mathrm{g}}$. More importantly, we use the linear-time LLT algorithm [11] for the first conjugation.[1] That is, for computing the conjugate of J in Fig. 1(a), we employ LLT to compute $J^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ using the data points $J^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$, for a *grid-like* discretization $\mathbb{Y}^{\mathrm{g}}$ of the dual domain. Proper construction of $\mathbb{Y}^{\mathrm{g}}$ will be discussed in the next subsection. Now, let

$$\psi_x^{\mathrm{d}}(y) := C_x^*(-f_{\mathrm{i}}(x)^\top y) + J^{\mathrm{d}*\mathrm{d}}(y), \quad y \in \mathbb{Y}^{\mathrm{g}}$$

be a discrete *approximation* of ϕ_x in (12a). The approximation stems from the fact that we used the discrete conjugate $J^{\mathrm{d}*}$ instead of the conjugate J^*. Then, we can also handle the last conjugation in Fig. 1(a) numerically, and approximate $\phi_x^*(f_{\mathrm{s}}(x))$ in (12b) by

$$\psi_x^{\mathrm{d}*}(f_{\mathrm{s}}(x)) = \max_{y\in\mathbb{Y}^{\mathrm{g}}}\{\langle f_{\mathrm{s}}(x), y\rangle - \psi_x^{\mathrm{d}}(y)\}$$

computed via enumeration over $y \in \mathbb{Y}^{\mathrm{g}}$. Hence, we can introduce the d-CDP operator as follows:

$$J^{\mathrm{d}*\mathrm{d}}(y) = \max_{x\in\mathbb{X}^{\mathrm{g}}}\left\{\langle y, x\rangle - J^{\mathrm{d}}(x)\right\}, \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{13a}$$

$$\psi_x^{\mathrm{d}}(y) = C_x^*(-f_{\mathrm{i}}(x)^\top y) + J^{\mathrm{d}*\mathrm{d}}(y), \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{13b}$$

$$\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) := \psi_x^{\mathrm{d}*}(f_{\mathrm{s}}(x)), \qquad x \in \mathbb{X}^{\mathrm{g}} \tag{13c}$$

with C_x^* being the partial conjugate of stage cost defined in (11). Algorithm 1 provides the pseudo-code for the numerical implementation of the d-CDP operation (13). We next analyze the performance of Algorithm 1 by considering its complexity and error.

2) Analysis of d-CDP Algorithm: We begin with the computational complexity of Algorithm 1. We will use the following result in the analysis of the algorithms developed in this study.

Remark 3.2 (Complexity of LLT): Consider a function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ and its discretization over a grid-like set $\mathbb{X}^{\mathrm{g}} \subset \mathbb{R}^n$ such that $\mathbb{X}^{\mathrm{g}} \cap \mathrm{dom}(h) \neq \emptyset$. LLT computes the discrete conjugate function $h^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ using the data points $h^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \overline{\mathbb{R}}$, with a time complexity of $\mathcal{O}(\Pi_{i=1}^n (X_i + Y_i))$, where X_i (resp. Y_i) is the cardinality of the ith dimension of the grid $\mathbb{X}^{\mathrm{g}}$ (resp. $\mathbb{Y}^{\mathrm{g}}$). In particular, if the grids $\mathbb{X}^{\mathrm{g}}$ and $\mathbb{Y}^{\mathrm{g}}$ have approximately the same cardinality in each dimension, then the complexity of LLT is of $\mathcal{O}(X + Y)$ [11, Corollary 5].

Hereafter, in order to simplify the exposition, we consider the following assumption.

Assumption 3.3 (Grid sizes in LLT): The primal and dual grids used for LLT have approximately the same size in each dimension.

Theorem 3.4 (Complexity of Algorithm 1): Let Assumptions 3.1 and 3.3 hold. The implementation of the d-CDP operator (13) via Algorithm 1 requires $\mathcal{O}(XY)$ operations.

Proof: By the assumptions and considering the complexity of LLT given in Remark 3.2, the result follows by taking into account the time complexity of each line of Algorithm 1.

Recall that the time complexity of the d-DP operator (8) is of $\mathcal{O}(XUE)$; see Remark 2.2. Comparing this complexity to the one reported in Theorem 3.4, points to a basic characteristic of CDP w.r.t. DP: CDP avoids the minimization over the control input in DP and casts it as a simple addition in the dual domain at the expense of three conjugate transforms. Consequently, the time complexity is transferred from the primal input domain $\mathbb{U}^{\mathrm{d}}$ to the dual state domain $\mathbb{Y}^{\mathrm{g}}$. This observation implies that if $Y < UE$, then d-CDP is expected to computationally outperform d-DP.

We now consider the error introduced by Algorithm 1 w.r.t. the DP operator (7). We begin with an alternative representation of the d-CDP operator that sheds some light on the main sources of error in the d-CDP operation.

Proposition 3.5 (d-CDP reformulation): Let Assumption 2.1 hold. Also, assume that $C : \mathbb{X} \times \mathbb{U} \to \overline{\mathbb{R}}$ is convex in the input variable. Then, the d-CDP operator (13) equivalently reads as

$$\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) = \min_u \left\{C(x,u) + J^{\mathrm{d}*\mathrm{d}*}(f(x,u))\right\}, \quad x \in \mathbb{X}^{\mathrm{g}} \tag{14}$$

where $J^{\mathrm{d}*\mathrm{d}*}$ is the discrete biconjugate of J, using the primal grid $\mathbb{X}^{\mathrm{g}}$ and the dual grid $\mathbb{Y}^{\mathrm{g}}$.

Proof: We can use (11) and (13) to write

$$\begin{aligned}
\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) &= \max_{y\in\mathbb{Y}^{\mathrm{g}}}\{\langle f_{\mathrm{s}}(x), y\rangle - \psi_x^{\mathrm{d}}(y)\} \\
&= \max_{y\in\mathbb{Y}^{\mathrm{g}}}\left\{\langle f_{\mathrm{s}}(x), y\rangle - \max_{u\in\mathrm{dom}\, C(x,\cdot)}\left[\langle -f_{\mathrm{i}}(x)^\top y, u\rangle \right.\right. \\
&\qquad \left.\left. - C(x,u)\right] - J^{\mathrm{d}*\mathrm{d}}(y)\right\} \\
&= \max_{y\in\mathbb{Y}^{\mathrm{g}}} \min_{u\in\mathrm{dom}\, C(x,\cdot)}\left\{C(x,u) + \langle y, f(x,u)\rangle - J^{\mathrm{d}*\mathrm{d}}(y)\right\}.
\end{aligned}$$

Since C is convex in u and the mapping f is affine in u, the objective function of this maximin problem is convex in u, with $\mathrm{dom}(C(x,\cdot))$ being compact. Also, the objective function is Ky Fan concave in y, which follows from the convexity of $J^{\mathrm{d}*}$. Then, by the Ky Fan's Minimax Theorem [25, Th. A], we can swap the maximization and minimization operators to obtain

$$\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) = \min_u \left\{C(x,u) + J^{\mathrm{d}*\mathrm{d}*}(f(x,u))\right\}.$$

■

Comparing (7) and (14), we note that the d-CDP operator $\widehat{\mathcal{T}}^{\mathrm{d}}$ differs from the DP operator $\mathcal{T}$ in that it uses $J^{\mathrm{d}*\mathrm{d}*}$ as an approximation of J. This observation points to two main sources of error in the proposed approach, namely, dualization and discretization. Indeed, $\widehat{\mathcal{T}}^{\mathrm{d}}$ is a discretized version of the dual problem (10). Regarding the dualization error, we note that the d-CDP operator is "blind" to nonconvexity; i.e., it essentially replaces the cost-to-go J by its convex envelope (the greatest

[1] LLT is an efficient algorithm for computing the discrete conjugate over a finite *grid-like* dual domain. Precisely, to compute the conjugate of the function $h : \mathbb{X} \to \mathbb{R}$, LLT takes its discretization $h^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$ as an input, and outputs $h^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$, for the grid-like dual domain $\mathbb{Y}^{\mathrm{g}}$. That is, LLT is equivalent to the operation $[\cdot]^{\mathrm{d}*\mathrm{d}}$. We refer the reader to [11] for a detailed description of LLT.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

convex function that supports J from below). The discretization error, on the other hand, depends on the choice of the finite primal and dual domains $\mathbb{X}^{\mathrm{g}}$ and $\mathbb{Y}^{\mathrm{g}}$. The following result captures this dependence.

Theorem 3.6 (Error of Algorithm 1): Let Assumption 2.1 hold. Consider the DP operator $\mathcal{T}$ (7) and the implementation of the d-CDP operator $\widehat{\mathcal{T}}^{\mathrm{d}}$ (13) via Algorithm 1. Assume that $C : \mathbb{X} \times \mathbb{U} \to \overline{\mathbb{R}}$ is convex in the input variable. Also, assume that $J : \mathbb{X} \to \mathbb{R}$ is a Lipschitz continuous, convex function. Then, we have

$$-e_2 \leq \mathcal{T}[J](x) - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \leq e_1(x), \quad \forall x \in \mathbb{X}^{\mathrm{g}} \tag{15}$$

where

$$\begin{aligned} e_1(x) &= c_1(x) \cdot \mathrm{d}\left(\partial\mathcal{T}[J](x), \mathbb{Y}^{\mathrm{g}}\right) \\ e_2 &= \left[\Delta_{\mathbb{Y}^{\mathrm{g}}} + \mathrm{L}(J)\right] \cdot \mathrm{d}_{\mathrm{H}}(\mathbb{X}, \mathbb{X}^{\mathrm{g}}) \end{aligned} \tag{16}$$

and $c_1(x) = \|f_{\mathrm{s}}(x)\| + \|f_{\mathrm{i}}(x)\| \cdot \Delta_{\mathbb{U}} + \Delta_{\mathbb{X}}$.

Proof: Fix $x \in \mathbb{X}^{\mathrm{g}}$ and observe that

$$\begin{aligned} &\mathcal{T}[J](x) - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \\ &\quad = \left[\mathcal{T}[J](x) - \widehat{\mathcal{T}}[J](x)\right] + \left[\widehat{\mathcal{T}}[J](x) - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x)\right]. \end{aligned} \tag{17}$$

First note that the convexity of C (in u) and J implies that the dualization error $\mathcal{T}[J] - \widehat{\mathcal{T}}[J]$ in (17) is zero. Indeed, following a similar argument to the one provided in the proof of Proposition 3.5 and using Sion's Minimax Theorem [26, Th. 3], we have

$$\widehat{\mathcal{T}}[J](x) = \min_u \left\{C(x,u) + J^{**}\left(f(x,u)\right)\right\}, \quad x \in \mathbb{X}.$$

Since J is proper, closed, and convex, we have $J^{**} = J$, and hence $\widehat{\mathcal{T}}[J] = \mathcal{T}[J]$. We next consider the discretization error $\widehat{\mathcal{T}}[J] - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}]$ in (17). From (12b) and (13c), we have ($z = f_{\mathrm{s}}(x)$)

$$\begin{aligned} \widehat{\mathcal{T}}[J](x) - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) &= \phi_x^*(z) - \psi_x^{\mathrm{d}*}(z) \\ &= \left[\phi_x^*(z) - \phi_x^{\mathrm{d}*}(z)\right] + \left[\phi_x^{\mathrm{d}*}(z) - \psi_x^{\mathrm{d}*}(z)\right] \end{aligned} \tag{18}$$

with $\phi_x^{\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ being the discretization of $\phi_x : \mathbb{R}^n \to \mathbb{R}$. For $\phi_x^* - \phi_x^{\mathrm{d}*}$ in (18), we can use Lemma 1.1 in Appendix to obtain

$$\begin{aligned} &0 \leq \phi_x^*\left(f_{\mathrm{s}}(x)\right) - \phi_x^{\mathrm{d}*}\left(f_{\mathrm{s}}(x)\right) \\ &\leq \min_{y \in \partial\phi_x^*(f_{\mathrm{s}}(x))} \left\{ \left[\|f_{\mathrm{s}}(x)\| + \mathrm{L}\left(\phi_x; \{y\} \cup \mathbb{Y}^{\mathrm{g}}\right)\right] \cdot \mathrm{d}(y, \mathbb{Y}^{\mathrm{g}}) \right\} \\ &\leq \min_{y \in \partial\mathcal{T}[J](x)} \left\{ \left[\|f_{\mathrm{s}}(x)\| + \|f_{\mathrm{i}}(x)\| \cdot \Delta_{\mathbb{U}} + \Delta_{\mathbb{X}}\right] \cdot \mathrm{d}(y, \mathbb{Y}^{\mathrm{g}}) \right\} \\ &= \left[\|f_{\mathrm{s}}(x)\| + \|f_{\mathrm{i}}(x)\| \cdot \Delta_{\mathbb{U}} + \Delta_{\mathbb{X}}\right] \cdot \min_{y \in \partial\mathcal{T}[J](x)} \mathrm{d}(y, \mathbb{Y}^{\mathrm{g}}) \\ &= c_1(x) \cdot \mathrm{d}\left(\partial\mathcal{T}[J](x), \mathbb{Y}^{\mathrm{g}}\right) = e_1(x) \end{aligned} \tag{19}$$

where for the last inequality, we used the fact that $\phi_x^*(f_{\mathrm{s}}(\cdot)) = \widehat{\mathcal{T}}[J](\cdot) = \mathcal{T}[J](\cdot)$, and

$$\begin{aligned} \mathrm{L}\left(\phi_x(\cdot)\right) &\leq \mathrm{L}\left(C_x^*(-f_{\mathrm{i}}(x)^\top \cdot)\right) + \mathrm{L}\left(J^*(\cdot)\right) \\ &\leq \|f_{\mathrm{i}}(x)\| \cdot \mathrm{L}(C_x^*) + \mathrm{L}(J^*) \\ &\leq \|f_{\mathrm{i}}(x)\| \cdot \Delta_{\mathrm{dom}(C(x,\cdot))} + \Delta_{\mathrm{dom}(J)} \\ &\leq \|f_{\mathrm{i}}(x)\| \cdot \Delta_{\mathbb{U}} + \Delta_{\mathbb{X}}. \end{aligned}$$

For $\phi_x^{\mathrm{d}*} - \psi_x^{\mathrm{d}*}$ in (18), first observe that for each $y \in \mathbb{Y}^{\mathrm{g}}$, we have

$$\phi_x^{\mathrm{d}}(y) - \psi_x^{\mathrm{d}}(y) = J^{*\mathrm{d}}(y) - J^{\mathrm{d}*\mathrm{d}}(y) = J^*(y) - J^{\mathrm{d}*}(y)$$

[see (12a) and (13b), and recall that h^{d} is simply a sampled version of h]. Moreover, since $\mathrm{dom}(J) = \mathbb{X}$ is compact, we can again use Lemma 1.1 in Appendix to write for each $y \in \mathbb{Y}^{\mathrm{g}}$

$$\begin{aligned} 0 \leq J^*(y) - J^{\mathrm{d}*}(y) &\leq \left[\|y\| + \mathrm{L}(J)\right] \cdot \mathrm{d}_{\mathrm{H}}(\mathbb{X}, \mathbb{X}^{\mathrm{g}}) \\ &\leq \left[\Delta_{\mathbb{Y}^{\mathrm{g}}} + \mathrm{L}(J)\right] \cdot \mathrm{d}_{\mathrm{H}}(\mathbb{X}, \mathbb{X}^{\mathrm{g}}) = e_2. \end{aligned}$$

That is, $0 \leq \phi_x^{\mathrm{d}}(y) - \psi_x^{\mathrm{d}}(y) \leq e_2$ for all $y \in \mathbb{Y}^{\mathrm{g}}$. Then, using the definition of the discrete conjugate, we have

$$0 \leq \psi_x^{\mathrm{d}*}\left(f_{\mathrm{s}}(x)\right) - \phi_x^{\mathrm{d}*}\left(f_{\mathrm{s}}(x)\right) \leq e_2, , \quad \forall x \in \mathbb{X}^{\mathrm{g}}. \tag{20}$$

Combining the inequalities (19) and (20) completes the proof. ■

Notice how the two terms e_1 and e_2 capture the errors due to the discretization of the dual state space ($\mathbb{Y}$) and the primal state space ($\mathbb{X}$), respectively. In particular, the first error term suggests that we choose $\mathbb{Y}^{\mathrm{g}}$ such that $\mathbb{Y}^{\mathrm{g}} \cap \partial\mathcal{T}[J](x) \neq \emptyset$ for all $x \in \mathbb{X}^{\mathrm{g}}$. However, even if we had access to $\mathcal{T}[J]$, satisfying such a condition could lead to dual grids of size $Y = \mathcal{O}(X^n)$. A more realistic objective is then to choose $\mathbb{Y}^{\mathrm{g}}$ such that $\mathrm{co}(\mathbb{Y}^{\mathrm{g}}) \cap \partial\mathcal{T}[J](x) \neq \emptyset$ for all $x \in \mathbb{X}^{\mathrm{g}}$. With such a construction, the distance $\mathrm{d}(\partial\mathcal{T}[J](x), \mathbb{Y}^{\mathrm{g}})$ and hence e_1 decrease by using finer grids for the dual domain. To this end, we need to approximate "the range of slopes" of the function $\mathcal{T}[J]$ for $x \in \mathbb{X}^{\mathrm{g}}$. Notice, however, that we do not have access to $\mathcal{T}[J]$ since it is the *output* of the d-CDP operation in Algorithm 1. What we have at our disposal as *inputs* are the stage cost C and the next step (discrete) cost-to-go J^{d}. A coarse way to approximate the range of slopes of $\mathcal{T}[J]$ is then to use the extrema of the functions C and J^{d}, and the diameter of $\mathbb{X}^{\mathrm{g}}$ in each dimension. The following remark explains such an approximation for the construction of $\mathbb{Y}^{\mathrm{g}}$.

Remark 3.7 (Construction of $\mathbb{Y}^{\mathrm{g}}$): Let $C^M = \max_{x,u} C(x,u)$ and $C^m = \min_{x,u} C(x,u)$. Compute $J^M = \max_{x \in \mathbb{X}^{\mathrm{g}}} J^{\mathrm{d}}(x)$ and $J^m = \min_{x \in \mathbb{X}^{\mathrm{g}}} J^{\mathrm{d}}(x)$, and then choose $\mathbb{Y}^{\mathrm{g}} = \Pi_{i=1}^n \mathbb{Y}_i^{\mathrm{g}} \subset \mathbb{R}^n$ such that for each dimension $i = 1, \ldots, n$, we have

$$\pm\alpha(C^M + J^M - C^m - J^m)/\Delta_{\mathbb{X}_i^{\mathrm{g}}} \in \mathrm{co}(\mathbb{Y}_i^{\mathrm{g}}).$$

Here, $\alpha > 0$ is a scaling factor mainly depending on the dimension n of the state space. Construction of $\mathbb{Y}^{\mathrm{g}}$ as described above requires $\mathcal{O}(X)$ operations *per iteration* for computing J^M and J^m via enumeration over $x \in \mathbb{X}^{\mathrm{g}}$.

B. From Quadratic to Linear Complexity

We now focus on a specific subclass of the considered optimal control problems and exploit the problem structure in this subclass to reduce the computational cost of the d-CDP algorithm. In this regard, a closer look to Algorithm 1 reveals a computational bottleneck in its numerical implementation: The computation of the objects $\psi_x^{\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$, $x \in \mathbb{X}^{\mathrm{g}}$, and their conjugates which requires working in the product space $\mathbb{X}^{\mathrm{g}} \times \mathbb{Y}^{\mathrm{g}}$. Hence, if the structure of the problem allows for the complete decomposition of these objects, then a significant reduction in the time complexity is achievable. This is indeed possible for problems with separable data in the state and input variables:

Setting 2: 1) The dynamics is input-affine with state-independent input dynamics, i.e., $f(x,u) = f_{\mathrm{s}}(x) + B \cdot u$, where $f_{\mathrm{s}} : \mathbb{R}^n \to \mathbb{R}^n$ and $B \in \mathbb{R}^{n \times m}$. 2) The stage cost is separable in state and input, i.e., $C(x,u) = C_{\mathrm{s}}(x) + C_{\mathrm{i}}(u)$, where $C_{\mathrm{s}} : \mathbb{X} \to \mathbb{R}$ and $C_{\mathrm{i}} : \mathbb{U} \to \mathbb{R}$.

Note that the separability of the stage cost implies that there are no state-dependent input constraints.

1) Modified d-CDP Algorithm: Under the conditions of Setting 2, the state cost (C_{s}) can be taken out of the minimization in the DP operator (7), i.e.,

$$\mathcal{T}[J](x) = C_{\mathrm{s}}(x) + \min_u \left\{C_{\mathrm{i}}(u) + J\left(f(x,u)\right)\right\}, \quad x \in \mathbb{X}. \tag{21}$$

Following the same dualization and then discretization procedure as in Section III-A1, we can derive the corresponding d-CDP operator

$$J^{\mathrm{d}*\mathrm{d}}(y) = \max_{x \in \mathbb{X}^{\mathrm{g}}} \left\{\langle y, x\rangle - J^{\mathrm{d}}(x)\right\}, \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{22a}$$

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2: For Modified d-CDP Operator (23) and Setting 2.

Input: dynamics $f_{\mathrm{s}} : \mathbb{R}^n \to \mathbb{R}^n$, $B \in \mathbb{R}^{n\times m}$;
discrete cost-to-go (at $t+1$) $J^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$;
state cost $C_{\mathrm{s}}(x) : \mathbb{X} \to \mathbb{R}$;
conjugate of input cost $C_{\mathrm{i}}^* : \mathbb{R}^m \to \mathbb{R}$;
grids $\mathbb{Y}^{\mathrm{g}}, \mathbb{Z}^{\mathrm{g}} \subset \mathbb{R}^n$.
Output: discrete cost-to-go (at t) $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}[J^{\mathrm{d}}](x) : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$.
1: use LLT to compute $J^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ from $J^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$;
2: $\psi^{\mathrm{d}}(y) \leftarrow C_{\mathrm{i}}^*(-B^\top y) + J^{\mathrm{d}*\mathrm{d}}(y)$ for $y \in \mathbb{Y}^{\mathrm{g}}$;
3: use LLT to compute $\psi^{\mathrm{d}*\mathrm{d}} : \mathbb{Z}^{\mathrm{g}} \to \mathbb{R}$ from $\psi^{\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$;
4: **for** each $x \in \mathbb{X}^{\mathrm{g}}$ **do**
5: use LERP to compute $\overline{\psi^{\mathrm{d}*\mathrm{d}}}(f_{\mathrm{s}}(x))$ from $\psi^{\mathrm{d}*\mathrm{d}} : \mathbb{Z}^{\mathrm{g}} \to \mathbb{R}$;
6: $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \leftarrow C_{\mathrm{s}}(x) + \overline{\psi^{\mathrm{d}*\mathrm{d}}}(f_{\mathrm{s}}(x))$;
7: **end for**

$$\psi^{\mathrm{d}}(y) := C_{\mathrm{i}}^*(-B^\top y) + J^{\mathrm{d}*\mathrm{d}}(y), \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{22b}$$

$$\widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) = C_{\mathrm{s}}(x) + \psi^{\mathrm{d}*}\left(f_{\mathrm{s}}(x)\right), \qquad x \in \mathbb{X}^{\mathrm{g}}. \tag{22c}$$

Here, again, we assume that the conjugate of the input cost is analytically available (similar to Assumption 3.1, now in the context posed by Setting 2).

Assumption 3.8 (Conjugate of input cost): The conjugate function $C_{\mathrm{i}}^*(v) = \max_u\{\langle v, u\rangle - C_{\mathrm{i}}(u)\}$ is analytically available; i.e., the complexity of evaluating $C_{\mathrm{i}}^*(v)$ for each $v \in \mathbb{R}^m$ is of $\mathcal{O}(1)$.

Notice that the function ψ^{d} in (22b) is now independent of the state variable x. Hence, the computation of ψ^{d} requires $\mathcal{O}(X+Y)$ operations, as opposed to $\mathcal{O}(XY)$ for the computation of ψ_x^{d}, in Algorithm 1. Next is the computation of the conjugate function $\psi^{\mathrm{d}*}(f_{\mathrm{s}}(x)) = \max_{y\in\mathbb{Y}^{\mathrm{g}}}\{\langle f_{\mathrm{s}}(x), y\rangle - \psi(y)\}$ for $x \in \mathbb{X}^{\mathrm{g}}$ in (22c). The straightforward maximization via enumeration over $y \in \mathbb{Y}^{\mathrm{g}}$ for each $x \in \mathbb{X}^{\mathrm{g}}$ (as in Algorithm 1) again leads to a time complexity of $\mathcal{O}(XY)$. The key idea for complexity reduction is to use *approximate discrete conjugation as follows:*

1) Use LLT to compute $\psi^{\mathrm{d}*\mathrm{d}} : \mathbb{Z}^{\mathrm{g}} \to \mathbb{R}$ from the data points $\psi^{\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ for a grid $\mathbb{Z}^{\mathrm{g}}$;
2) For each $x \in \mathbb{X}^{\mathrm{g}}$, use LERP to compute $\overline{\psi^{\mathrm{d}*\mathrm{d}}}(f_{\mathrm{s}}(x))$ from the data points $\psi^{\mathrm{d}*\mathrm{d}} : \mathbb{Z}^{\mathrm{g}} \to \mathbb{R}$.

Proper construction of the grid $\mathbb{Z}^{\mathrm{g}}$ will be discussed shortly. With such an approximation, the d-CDP operator (22) *modifies* to

$$J^{\mathrm{d}*\mathrm{d}}(y) = \max_{x\in\mathbb{X}^{\mathrm{g}}}\left\{\langle y, x\rangle - J^{\mathrm{d}}(x)\right\}, \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{23a}$$

$$\psi^{\mathrm{d}}(y) = C_{\mathrm{i}}^*(-B^\top y) + J^{\mathrm{d}*\mathrm{d}}(y), \qquad y \in \mathbb{Y}^{\mathrm{g}} \tag{23b}$$

$$\psi^{\mathrm{d}*\mathrm{d}}(z) = \max_{y\in\mathbb{Y}^{\mathrm{g}}}\left\{\langle z, y\rangle - \psi^{\mathrm{d}}(y)\right\}, \qquad z \in \mathbb{Z}^{\mathrm{g}} \tag{23c}$$

$$\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}[J^{\mathrm{d}}](x) := C_{\mathrm{s}}(x) + \overline{\psi^{\mathrm{d}*\mathrm{d}}}\left(f_{\mathrm{s}}(x)\right), \qquad x \in \mathbb{X}^{\mathrm{g}} \tag{23d}$$

with C_{i}^* being the conjugate of input cost defined in Assumption 3.8. Algorithm 2 provides the pseudo-code of the preceding scheme.

2) Analysis of Modified d-CDP Algorithm: We again begin with the time complexity of the proposed algorithm.

Theorem 3.9 (Complexity of Algorithm 2): Let Assumptions 3.3 and 3.8 hold. Then, the computation of the modified d-CDP operator (23) via Algorithm 2 has a time complexity of $\widetilde{\mathcal{O}}(X+Y+Z)$.

Proof: The results follow by considering the complexity of each line of Algorithm 2, while taking into account Remark 3.2 and the fact that LERP has a logarithmic complexity [23, Remark 2.2]. ■

Comparing the time complexity of the d-CDP Algorithm 2 with that of the d-CDP Algorithm 1, we can observe the reduction from quadratic to (log-) linear complexity. In particular, assuming that all of the involved grids ($\mathbb{X}^{\mathrm{g}}, \mathbb{Y}^{\mathrm{g}}, \mathbb{Z}^{\mathrm{g}}$) are of the same size, i.e., $Y, Z = X$ (this is also consistent with Assumption 3.3), the complexity of the d-CDP Algorithm 1 is of $\mathcal{O}(X^2)$, while that of the d-CDP Algorithm 2 is of $\widetilde{\mathcal{O}}(X)$. We next consider the error of the proposed algorithm.

Theorem 3.10 (Error of Algorithm 2): Let Assumption 2.1 hold. Consider the DP operator $\mathcal{T}$ (21) and the implementation of the modified d-CDP operator $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}$ (23) via Algorithm 2. Assume that the input cost $C_{\mathrm{i}} : \mathbb{U} \to \mathbb{R}$ is convex, and the function $J : \mathbb{X} \to \mathbb{R}$ is a Lipschitz continuous, convex function. Also, assume that the grid $\mathbb{Z}^{\mathrm{g}}$ in Algorithm 2 is such that $\mathrm{co}(\mathbb{Z}^{\mathrm{g}}) \supseteq f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}})$. Then

$$-(e_2+e_3) \le \mathcal{T}[J](x) - \widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \le e_1^m(x), \quad \forall x \in \mathbb{X}^{\mathrm{g}} \tag{24}$$

where

$$\begin{aligned} e_1^m(x) &:= c_1^m(x) \cdot \mathrm{d}\left(\partial\left(\mathcal{T}[J] - C_{\mathrm{s}}\right)(x), \mathbb{Y}^{\mathrm{g}}\right) \\ e_2 &= \left[\Delta_{\mathbb{Y}^{\mathrm{g}}} + \mathrm{L}(J)\right] \cdot \mathrm{d}_{\mathrm{H}}(\mathbb{X}, \mathbb{X}^{\mathrm{g}}) \\ e_3 &:= \Delta_{\mathbb{Y}^{\mathrm{g}}} \cdot \mathrm{d}_{\mathrm{H}}\left(f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}}), \mathbb{Z}^{\mathrm{g}}\right) \end{aligned} \tag{25}$$

and $c_1^m(x) = \|f_{\mathrm{s}}(x)\| + \|B\| \cdot \Delta_{\mathbb{U}} + \Delta_{\mathbb{X}}$.

Proof: Let $\widehat{\mathcal{T}}^{\mathrm{d}}$ denote the output of the implementation of the d-CDP operator (22) via Algorithm 1. Note that the computation of the modified d-CDP operator $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}$ (23) via Algorithm 2 differs form that of the d-CDP operator $\widehat{\mathcal{T}}^{\mathrm{d}}$ (22) via Algorithm 1 only in the last step: $\widehat{\mathcal{T}}^{\mathrm{d}}$ *exactly* computes $\psi^{\mathrm{d}*}(f_{\mathrm{s}}(x))$ for $x \in \mathbb{X}^{\mathrm{g}}$ (see Algorithm 1:4), however, in $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}$, the *approximation* $\overline{\psi^{\mathrm{d}*\mathrm{d}}}(f_{\mathrm{s}}(x))$ is used (see Algorithm 2:5), where the approximation uses LERP over the data points $\psi^{\mathrm{d}*\mathrm{d}} : \mathbb{Z}^{\mathrm{g}} \to \mathbb{R}$. By Lemma 1.2 in Appendix, this leads to an over-approximation of $\psi^{\mathrm{d}*}$ with the upper bound

$$e_3 = \Delta_{\mathbb{Y}^{\mathrm{g}}} \cdot \max_{x\in\mathbb{X}^{\mathrm{g}}} \mathrm{d}\left(f_{\mathrm{s}}(x), \mathbb{Z}^{\mathrm{g}}\right) = \Delta_{\mathbb{Y}^{\mathrm{g}}} \cdot \mathrm{d}_{\mathrm{H}}\left(f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}}), \mathbb{Z}^{\mathrm{g}}\right).$$

Hence, compared to $\widehat{\mathcal{T}}^{\mathrm{d}}$, the operator $\widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}$ is an over-approximation with the difference bounded by e_3. That is

$$0 \le \widehat{\mathcal{T}}_{\mathrm{m}}^{\mathrm{d}}[J^{\mathrm{d}}](x) - \widehat{\mathcal{T}}^{\mathrm{d}}[J^{\mathrm{d}}](x) \le e_3, \quad \forall x \in \mathbb{X}^{\mathrm{g}}.$$

The result then follows from Theorem 3.6 by taking into account the effect of the state cost C_{s}. ■

Once again, the terms e_1^m, e_2, and e_3 capture the errors due to the discretization of y, x, and z, respectively. We now use this result to provide some guidelines on the construction of the required grids. Concerning the grid $\mathbb{Y}^{\mathrm{g}}$, the error term e_1^m in (25) implies similar guidelines to the ones provided in Section III-A2. In particular, notice that e_1^m now depends on $\mathrm{d}(\partial(\mathcal{T}[J] - C_{\mathrm{s}})(x), \mathbb{Y}^{\mathrm{g}})$, and hence in the construction of $\mathbb{Y}^{\mathrm{g}}$, we need to consider the range of slopes of $\mathcal{T}[J] - C_{\mathrm{s}}$. This essentially means using $C_{\mathrm{i}}^M = \max_{u\in\mathbb{U}} C_{\mathrm{i}}$ and $C_{\mathrm{i}}^m = \min_{u\in\mathbb{U}} C_{\mathrm{i}}$ instead of C^M and C^m, respectively, in Remark 3.7. Next to be addressed is the construction of the grid $\mathbb{Z}^{\mathrm{g}}$. Here, we are dealing with the issue of constructing the dual grid for approximate discrete conjugation. In particular, the condition $\mathrm{co}(\mathbb{Z}^{\mathrm{g}}) \supseteq f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}})$ in Theorem 3.10 follows from Lemma 1.2 in Appendix. The following remark summarizes this discussion.

Remark 3.11 (Construction of $\mathbb{Z}^{\mathrm{g}}$): Construct the grid $\mathbb{Z}^{\mathrm{g}}$ such that $\mathrm{co}(\mathbb{Z}^{\mathrm{g}}) \supseteq f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}})$. This can be done by finding the vertices of the smallest hyper-rectangle that contains the set $f_{\mathrm{s}}(\mathbb{X}^{\mathrm{g}})$. Constructing $\mathbb{Z}^{\mathrm{g}}$ in this way has a *one-time* cost of $\mathcal{O}(X)$.

We finish this section with some remarks on using the output of the backward value iteration for finding a suboptimal control sequence $\mathbf{u}^\star(x_0)$ for a given instance of the optimal control problem with initial state x_0. Having the discrete costs-to-go $J_t^{\mathrm{d}} : \mathbb{X}^{\mathrm{g}} \to \mathbb{R}$, $t = 0, 1, \ldots, T-1$, at our disposal (the output of the d-DP or d-CDP algorithm), at each time step, we can use the greedy policy w.r.t. the next step's cost-to-go, i.e.,

$$u_t^\star \in \mathrm{argmin}_{u_t\in\mathbb{U}^{\mathrm{d}}}\left\{C(x_t, u_t) + \widetilde{J_{t+1}^{\mathrm{d}}}\left(f(x_t, u_t)\right)\right\} \tag{26}$$

for a proper discrete input space $\mathbb{U}^{\mathrm{d}}$. Assuming these minimization problems are handled via enumeration, they lead to an additional

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

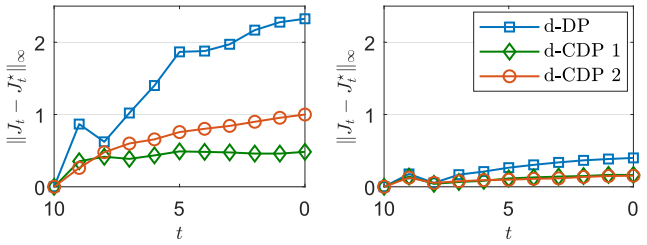
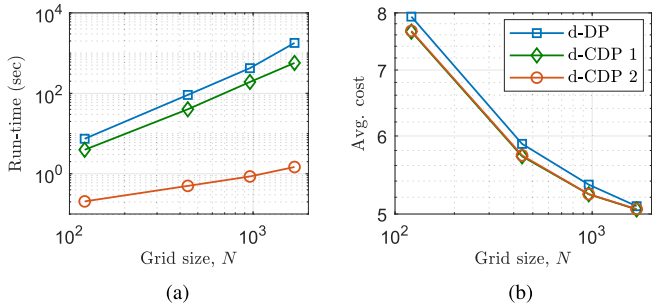


Fig. 2. Error of the computed discrete costs-to-go for $N = 11^2$ (left) $N = 41^2$ (right). Notice that the time axis is backward.



(a) (b)

Fig. 3. Performance for different sizes N of the grids $\mathbb{X}^{\mathrm{g}}, \mathbb{U}^{\mathrm{g}}, \mathbb{Y}^{\mathrm{g}}, \mathbb{Z}^{\mathrm{g}}$. (a) Total running time for solving a random problem instance. (b) Average cost of controlled trajectories for 100 random initial states.

computational burden of $\mathcal{O}(UE)$ per iteration, where E represents the complexity of the extension operation in (26). Then, the *total* time complexity of solving a T-step problem (i.e., the time requirement of backward value iteration for finding J_t^{d}, $t = 0, 1, \ldots, T-1$, plus the time requirement of forward iteration for finding $u_t^\star$, $t = 0, 1, \ldots, T-1$) of the three algorithms can be summarized as follows.

Remark 3.12 (Comparison of total time complexities): The total time complexity of solving a T-step optimal control problem for a given initial state, where the control input is generated using the greedy policy (26), is as follows: 1) $\mathcal{O}(TXUE)$ for the d-DP algorithm; 2) $\mathcal{O}(T(XY + UE))$ for the d-CDP Algorithm 1; 3) $\widetilde{\mathcal{O}}(T(X + Y + Z + UE))$ for the d-CDP Algorithm 2, where E represents the complexity of the operation $\widetilde{[\cdot]}$ in (8) and (26).

IV. Numerical Experiments

We now examine the performance of the proposed d-CDP algorithms (referred to as d-CDP 1 and d-CDP 2) in comparison with the generic d-DP algorithm (referred to as d-DP) through a synthetic numerical example. In particular, we use these numerical simulations to verify our theoretical results on the complexity and error of the proposed algorithms. We refer the reader to [23, Sec. 6 and Appendix C] for more numerical simulations of these algorithms (and their extensions). Finally, we note that all the simulations presented in this article were implemented via MATLAB version R2017b, on a PC with an Intel Xeon 3.60 GHz processor and 16 GB RAM.

Throughout this section, we consider a linear system $f(x, u) = Ax + Bu$ with two states and two inputs, where $A = [-0.5\ 2;\ 1\ 3]$ and $B = [1\ \ 0.5;\ \ 1\ \ 1]$, over the finite horizon $T = 10$, with the state and input constraints $x_t \in \mathbb{X} = [-1, 1]^2 \subset \mathbb{R}^2$ and $u_t \in \mathbb{U} = [-2, 2]^2 \subset \mathbb{R}^2$, respectively. Moreover, we consider *quadratic* state cost $C_{\mathrm{s}}(x) = C_T(x) = \|x\|^2$ and *exponential* input cost $C_{\mathrm{i}}(u) = e^{|u_1|} + e^{|u_2|} - 2$. Note that the conjugate of the input cost is given by $C_{\mathrm{i}}^*(v) = 1 + \langle \hat{u}, v\rangle - e^{|\hat{u}_1|} - e^{|\hat{u}_2|}$, $v \in \mathbb{R}^2$, where $\hat{u}_i = \max\{-2,\ \min\{2,\ \mathrm{sgn}(v_i) \ln |v_i|\}\}$ if $v_i \neq 0$, and $= 0$ otherwise, in each dimension $i = 1, 2$. Moreover, corresponding to the notation of Section III-A, we have $C_x^*(v) = C_{\mathrm{i}}^*(v) - \|x\|^2$.

We use *uniform* grid-like discretizations $\mathbb{X}^{\mathrm{g}}$ and $\mathbb{U}^{\mathrm{g}}$ for the state and input spaces, such that $\mathrm{co}(\mathbb{X}^{\mathrm{g}}) = \mathbb{X}$ and $\mathrm{co}(\mathbb{U}^{\mathrm{g}}) = \mathbb{U}$. The grids $\mathbb{Y}^{\mathrm{g}}$ and $\mathbb{Z}^{\mathrm{g}}$ in d-CDP algorithms are also constructed *uniformly*, following the guidelines of Remarks 3.7 and 3.11 (with $\alpha = 1$). We note that the extension of the discrete cost functions [in the d-DP operation (8) and for generating greedy control actions in (26)] is also handled via LERP. Moreover, we take all of the involved grids to be of the same size $N = X, U, Y, Z$. We are interested in the performance of d-CDP algorithms in comparison with d-DP, as the size N of these discrete sets increases.

We begin with examining the error in d-DP and d-CDP algorithms. Since the problem does not have a closed-form solution, our "reference" is the costs-to-go $J_t^\star : \mathbb{X} \to \mathbb{R}$ computed numerically via a high-resolution application of d-DP with $X, U = 81^2$. Fig. 2 depicts the maximum absolute error in the discrete costs computed using these algorithms over the horizon. As expected and in line with our error analysis (Theorems 3.6 and 3.10), using a finer discretization scheme with larger N, leads to a smaller error. Moreover, over the time steps in the backward iteration, due to the accumulation of error, a general increase is seen in the error.

TABLE I
Coarse Dual Grid $\mathbb{Y}^{\mathrm{g}}$ in d-CDP 1

Algorithm	Run-time (sec)	Avg. cost
d-DP with $X, U = 41^2$	1790	5.09
d-CDP 1 with $X, Y = 41^2$	570	5.05
d-CDP 1 with $X = 41^2, Y = 21^2$	187	5.05

The first two rows correspond to the rightmost data points in Figure 3.

We next compare the performance of the three algorithms in solving instances of the optimal control problem, using the cost functions derived from the backward value iteration. To this end, we apply the greedy control input (26) w.r.t. the computed discrete costs-to-go J_t^{d} via d-DP and d-CDP algorithms, using the same discretization $\mathbb{U}^{\mathrm{g}}$ of the input space as in d-DP. Fig. 3(b) reports the average cost of the controlled trajectories over 100 instances of the optimal control problem with random initial conditions, chosen uniformly from $\mathbb{X} = [-1, 1]^2$. As shown, d-CDP algorithms have effectively the same performance as our benchmark d-DP when it comes to the quality of greedy control actions.

Let us now consider the complexity of d-DP and d-CDP algorithms. Fig. 3(a) reports the *total* run-time of a random problem instance for different grid sizes. Regarding the reported running times, first, note that they correspond to the given complexities in Theorems 3.4 and 3.9 and Remark 3.12: For our numerical example, the running time is of $\mathcal{O}(TN^2)$ for d-DP and d-CDP 1, and of $\mathcal{O}(TN)$ for d-CDP 2. The difference can be readily seen in the slope of the corresponding lines in Fig. 3(a) as N increases. Hence, as expected, the application of d-CDP 2 leads to a significant reduction in the running time. In particular, notice how the lower complexity of d-CDP 2 allows us to increase the size of the grids to $N = 41^2$, while keeping the running time at the same order as d-DP with $N = 11^2$. Second, we note that d-CDP 1 gives us an extra degree of freedom for the size Y of the dual grid. In particular, if the cost functions are "compactly representable" in the dual domain (i.e., via their slopes), we can reduce the time complexity of d-CDP 1 by using a more coarse grid $\mathbb{Y}^{\mathrm{g}}$, with a limited effect on the "quality" of computed cost functions. This effect is illustrated in Table I: For solving the same optimal control problem, we can reduce the size of the dual grid (and hence reduce the running time of d-CDP 1) by a factor of 4, while achieving the same average cost in the controlled trajectories.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.

V. Final Remarks

In this article, we proposed an alternative scheme for the approximate implementation of the DP algorithm for solving discrete-time, optimal control problems. The proposed d-CDP algorithms use the linear-time LLT algorithm for solving the discretized version of the corresponding dual problem. Due to this dualization, the d-CDP algorithms may suffer from duality gap for problems with nonconvex costs-to-go. In particular, we focused on input-affine dynamics with separable data in the state and input variables: We first used the linearity of the dynamics in the input to transform the minimization in DP operation into an addition in the dual domain and then used the separability of the dynamics and costs in the state and input to reduce the standard quadratic complexity of DP operation to a linear one. Nevertheless, the proposed scheme still suffers from the infamous "curse of dimensionality" in the sense that the computational cost increases exponentially with the dimension of the state and input spaces. However, in the d-CDP Algorithm 2, the rate of exponential increase is $\max\{n, m\}$ (corresponding to $\mathcal{O}(X + U)$ complexity), compared to the rate $m + n$ for the d-DP algorithm (corresponding to $\mathcal{O}(XU)$ complexity).

We finish this section with some remarks on the extensions of the proposed algorithms. First, the proposed algorithm, with some modifications, can handle stochastic dynamics of the form $x_{t+1} = f(x_t, u_t) + w_t$ with an *independent, additive* disturbance w_t, while achieving the same reduction in the time complexity. We note however that for a generic disturbance, the expectation operation has a complexity of $\mathcal{O}(XU)$, which dominates the complexity of the proposed algorithm. Second, the assumption on the availability of the conjugate of the stage cost can be restrictive. While for some functions such as quadratic, the conjugate is available in closed form, for generic functions, this is not the case. To address this issue, we also developed a scheme for the numerical computation of these conjugate functions with a (log-) linear complexity in the size U of the discrete input space. The proposed scheme also introduces some error that mainly depends on the grids that are used for the discretization of the input space and its dual domain. We note that the provided MATLAB package already includes these extensions; see [23, Appendix C] for more details. Moreover, the application of the extended d-CDP Algorithm 2 for solving infinite-horizon, discounted cost problems, along with a detailed error and complexity analysis, is available in [27].

Appendix

We refer the reader to [23, Sec. 2.4] for the proofs of the following lemmas on the error of (approximate) discrete conjugation. Consider the function $h : \mathbb{X} \to \mathbb{R}$ and its discretization $h^{\mathrm{d}} : \mathbb{X}^{\mathrm{d}} \to \mathbb{R}$ with finite domain $\mathbb{X}^{\mathrm{d}} \subset \mathbb{X} \neq \emptyset$. Let $h^{\mathrm{d}*\mathrm{d}} : \mathbb{Y}^{\mathrm{g}} \to \mathbb{R}$ be the discretization of $h^{\mathrm{d}*}$ with $\mathbb{Y}^{\mathrm{g}} \subset \mathbb{R}^n$, and $\overline{h^{\mathrm{d}*\mathrm{d}}} : \mathbb{R}^n \to \mathbb{R}$ be the LERP extension of $h^{\mathrm{d}*\mathrm{d}}$.

Lemma 1.1 (Conjugate versus discrete conjugate): Let h be closed and convex. Then, $0 \leq h^*(y) - h^{\mathrm{d}*}(y) \leq \widetilde{e}_1$, $\forall y \in \mathbb{R}^n$, where

$$\widetilde{e}_1 := \min_{x \in \partial h^*(y)} \left\{ \left[\|y\| + \mathrm{L}\left(h; \{x\} \cup \mathbb{X}^{\mathrm{d}}\right) \right] \cdot \mathrm{d}(x, \mathbb{X}^{\mathrm{d}}) \right\}.$$

If, moreover, $\mathbb{X}$ is compact and h is Lipschitz continuous, then $0 \leq h^*(y) - h^{\mathrm{d}*}(y) \leq \widetilde{e}_2 := [\|y\| + \mathrm{L}(h)] \cdot \mathrm{d}_{\mathrm{H}}(\mathbb{X}, \mathbb{X}^{\mathrm{d}})$, $\forall y \in \mathbb{R}^n$.

Lemma 1.2 (Approximate discrete conjugate): We have

$$0 \leq \overline{h^{\mathrm{d}*\mathrm{d}}}(y) - h^{\mathrm{d}*}(y) \leq \Delta_{\mathbb{X}^{\mathrm{d}}} \cdot \mathrm{d}(y, \mathbb{Y}^{\mathrm{g}}), \quad \forall y \in \mathrm{co}(\mathbb{Y}^{\mathrm{g}}).$$

Acknowledgement

The authors would like to thank G. F. Max for several fruitful discussions.

References

[1] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. I, 3rd ed. Belmont, MA, USA: Athena Scientific, 2005.
[2] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*. Belmont, MA, USA: Athena Scientific, 2019.
[3] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. Hoboken, NJ, USA: Wiley, 2011.
[4] D. P. Bertsekas, *Convex Optimization Theory*. Belmont, MA, USA: Athena Scientific, 2009.
[5] R. Bellman and W. Karush, "Mathematical programming and the maximum transform," *J. Soc. Ind. Appl. Math.*, vol. 10, no. 3, pp. 550–567, 1962.
[6] A. O. Esogbue and C. W. Ahn, "Computational experiments with a class of dynamic programming algorithms of higher dimensions," *Comput. Math. Appl.*, vol. 19, no. 11, pp. 3–23, 1990.
[7] C. M. Klein and T. L. Morin, "Conjugate duality and the curse of dimensionality," *Eur. J. Oper. Res.*, vol. 50, no. 2, pp. 220–228, 1991.
[8] R. Rockafellar, *Conjugate Duality and Optimization*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1974.
[9] L. Corrias, "Fast Legendre-Fenchel transform and applications to Hamilton–Jacobi equations and conservation laws," *SIAM J. Numer. Anal.*, vol. 33, no. 4, pp. 1534–1558, 1996.
[10] Y. Lucet, "A fast computational algorithm for the Legendre-Fenchel transform," *Comput. Optim. Appl.*, vol. 6, no. 1, pp. 27–57, 1996.
[11] Y. Lucet, "Faster than the fast legendre transform, the linear-time Legendre transform," *Numer. Algorithms*, vol. 16, no. 2, pp. 171–185, 1997.
[12] Y. Achdou, F. Camilli, and L. Corrias, "On numerical approximation of the Hamilton-Jacobi-transport system arising in high frequency approximations," *Discrete Continuous Dyn. Syst. Ser. B*, vol. 19, no. 3, pp. 629–650, 2014.
[13] G. Costeseque and J.-P. Lebacque, "A variational formulation for higher order macroscopic traffic flow models: Numerical investigation," *Transp. Res. Part B: Methodol.*, vol. 70, pp. 112–133, 2014.
[14] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," *Theory Comput.*, vol. 8, no. 1, pp. 415–428, 2012.
[15] Y. Lucet, "New sequential exact Euclidean distance transform algorithms based on convex analysis," *Image Vis. Comput.*, vol. 27, no. 1, pp. 37–44, 2009.
[16] R. Carpio and T. Kamihigashi, "Fast value iteration: An application of legendre-fenchel duality to a class of deterministic dynamic programming problems in discrete time," *J. Difference Equ. Appl.*, vol. 26, no. 2, pp. 209–222, 2020.
[17] W. M. McEneaney, "Max-plus eigenvector representations for solution of nonlinear H_∞ problems: Basic concepts," *IEEE Trans. Autom. Control*, vol. 48, no. 7, pp. 1150–1163, Jul. 2003.
[18] W. M. McEneaney, *Max-Plus Methods for Nonlinear Control and Estimation*. Boston, MA, USA: Birkhäuser, 2006.
[19] M. Akian, S. Gaubert, and A. Lakhoua, "The max-plus finite element method for solving deterministic optimal control problems: Basic properties and convergence analysis," *SIAM J. Control Optim.*, vol. 47, no. 2, pp. 817–848, 2008.
[20] E. Berthier and F. Bach, "Max-plus linear approximations for deterministic continuous-state Markov decision processes," *IEEE Control Syst. Lett.*, vol. 4, no. 3, pp. 767–772, Jul. 2020.
[21] D. Sutter, G. Nannicini, T. Sutter, and S. Woerner, "Quantum Legendre-Fenchel transform," 2021, *arXiv:2006.04823*.
[22] M. A. S. Kolarijani and P. Mohajerin Esfahani, "d-CDP MATLAB package," 2022. [Online]. Available: https://github.com/AminKolarijani/d-CDP
[23] M. A. S. Kolarijani and P. Mohajerin Esfahani, "Fast approximate dynamic programming for input-affine dynamics," 2022, *arXiv:2008.10362*.
[24] E. J. Kirkland, *Advanced Computing in Electron Microscopy*. Boston, MA, USA: Springer US, 2010.
[25] I. Joó and L. L. Stachó, "A note on Ky Fan's minimax theorem," *Acta Math. Academiae Scientiarum Hungarica*, vol. 39, no. 4, pp. 401–407, 1982.
[26] S. Simons, "Minimax theorems and their proofs," in *Minimax and Applications*, D.-Z. Du and P. M. Pardalos, Eds. Boston, MA, USA: Springer US, 1995, pp. 1–23.
[27] M. A. S. Kolarijani, G. Max, and P. M. Esfahani, "Fast approximate dynamic programming for infinite-horizon Markov decision processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 23652–23663.

Authorized licensed use limited to: TU Delft Library. Downloaded on October 30,2023 at 07:56:31 UTC from IEEE Xplore. Restrictions apply.