HJB Optimal Feedback Control with Deep Differential Value Functions and Action Constraints

Michael Lutter[1], Boris Belousov[1], Kim Listmann[2], Debora Clever[1,2], Jan Peters[1,3]

[1] Department of Computer Science, Technische Universität Darmstadt, Germany
[2] ABB Corporate Research Center at Ladenburg, Germany
[3]Robot Learning Group, Max Planck Institute for Intelligent Systems,Tübingen, Germany
`{lutter, belousov, peters}@ias.informatik.tu-darmstadt.de`
`{kim.listmann, debora.clever}@de.abb.com`

Abstract: Learning optimal feedback control laws capable of executing optimal trajectories is essential for many robotic applications. Such policies can be learned using reinforcement learning or planned using optimal control. While reinforcement learning is sample inefficient, optimal control only plans an optimal trajectory from a specific starting configuration. In this paper we propose HJB control to learn an optimal feedback policy rather than a single trajectory using principles from optimal control. By exploiting the inherent structure of the robot dynamics and strictly convex action cost, we derive principled cost functions such that the optimal policy naturally obeys the action limits, is globally optimal and stable on the training domain given the optimal value function. The corresponding optimal value function is learned end-to-end by embedding a deep differential network in the Hamilton-Jacobi-Bellmann differential equation and minimizing the error of this equality while simultaneously decreasing the discounting from short- to far-sighted to enable the learning. Our proposed approach enables us to learn an optimal feedback control law in continuous time, that in contrast to existing approaches generates an optimal trajectory from any point in state-space without the need of replanning. The resulting approach is evaluated on non-linear systems and achieves optimal feedback control, where standard optimal control methods require frequent replanning.

1 Introduction

Specifying a task through a reward function and letting an agent autonomously discover a corresponding controller promises to simplify programming of complex robotic behaviors by reducing the required amount of manual engineering. Previous research demonstrated that such approach can successfully generate robot controllers capable of performing dexterous manipulation [1, 2, 3] and locomotion [4, 5]. These controllers were obtained via reinforcement learning or trajectory optimization. While reinforcement learning optimizes a possibly non-linear policy π under the assumption of unknown rewards, dynamics and actuation limits, trajectory optimization plans a sequence of n actions and states using the known model, reward function, initial state and actuator limits. When applied to the physical system, the planned trajectories must be augmented with a hand-tuned tracking controller to compensate modeling errors.

To obtain a globally optimal feedback policy that naturally obeys the actuator limits without randomly sampling actions on the system as in reinforcement learning, we propose to incorporate actuator limits within the cost function and obtain the optimal feedback controller by embedding a deep differential network and the known model in the Hamilton-Jacobi-Bellman (HJB) differential equation. The network weights are learned using a curricular learning scheme that adapts the discounting from short to far-sighted to ensure learning of the optimal policy despite the multiple spurious solutions of the HJB. Assuming the inherent structure of most robotic tasks, i.e., control-affine dynamics due to holonomicity of mechanical systems and perfect approximation of the value function, the learned policy is globally optimal on the state domain Ω, guaranteed to be stable and does not require any replanning or hand-tuning of the feedback gains. Incorporating the actuation limits within the cost

function, transforms the constrained optimization problem to an unconstrained problem. Thereby enabling the learning of feasible feedback policies using gradient descent.

Our technical contributions are the following. First, the derivation of strictly convex cost functions that make common classes of controllers, including torque limited policies, optimal. Second, the derivation of a curricular learning scheme that can learn the optimal value function and optimal feedback policy by minimizing the residual of the HJB. Furthermore, we provide intuitive explanations about the convergence of the proposed curricular learning scheme.

In the following, we introduce the HJB differential equation and simplify the HJB using the inherent structure of robotic problems (Section 2). Afterwards, we derive the principled cost functions that enable torque limited policies to be optimal (Section 3). Using these insights, we propose a curriculum learning scheme to learn the optimal value function and policy in Section 4. Finally, our proposed algorithm is applied to non-linear systems (Section 5) and the differences to existing literature is highlighted (Section 6).

2 The Hamilton-Jacobi-Bellman Differential Equation

The optimal policy π^* solves a task by choosing the optimal actions $\mathbf{u}^*$ given the current state $\mathbf{x}$, such that the total accumulated cost is minimized. Therefore, the optimal policy π^* is described by

$$\pi^*(\mathbf{x}, t) = \arg\min_{\pi} \int_t^T e^{-\rho(\tau - t)} \, c(\mathbf{x}(\tau), \mathbf{u}(\tau)) \, \mathrm{d}\tau \quad \text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \quad \mathbf{x} \in \Omega \subseteq \mathbb{R}^{n_x}, \quad |\mathbf{u}| \leq \mathbf{u}_+ \quad (1)$$

with the action $\mathbf{u} = \pi(\mathbf{x}, t)$, the cost function $c(\mathbf{x}, \mathbf{u})$, the dynamics $f(\mathbf{x}, \mathbf{u})$, the discounting factor ρ, the state domain Ω, the positive actuator limit $\mathbf{u}_+$ and the time horizon T. Let the optimal value function $V^*(\mathbf{x}, t)$ be defined as the minimum accumulated cost-to-go, i.e.,

$$V^*(\mathbf{x}, t) = \min_{\mathbf{u}} \int_t^T e^{-\rho(\tau - t)} \, c(\mathbf{x}(\tau), \mathbf{u}(\tau)) \, \mathrm{d}\tau. \quad (2)$$

For the infinite time horizon, i.e., $T \to \infty$, the optimal value function is independent of time [6]. Using Equation 2, the Hamilton-Jacobi-Bellman differential equation, the continuous counterpart of the discrete Bellmann equation, can be derived by substituting the value function at time $t' = t + \Delta t$, approximating $V^*(x(t'), t')$ with its 1st order Taylor expansion and taking the limit $\Delta t \to 0$ [6]. Therefore, the HJB differential equation is described by

$$\rho V^*(\mathbf{x}, t) = \min_{\mathbf{u}} \; c(\mathbf{x}, \mathbf{u}) \; + f(\mathbf{x}, \mathbf{u})^T \; \frac{\partial V^*}{\partial \mathbf{x}}. \quad (3)$$

In the following, we will refer to $\partial V^* / \partial \mathbf{x}$ as V_x^*. The HJB has multiple solutions and only incorporating the boundary condition

$$f(\bar{\mathbf{x}}, \mathbf{u}^*)^T \boldsymbol{\eta}(\bar{\mathbf{x}}) \leq 0 \quad \text{for } \bar{\mathbf{x}} \in \partial\Omega \quad (4)$$

with the outward pointing normal vector $\boldsymbol{\eta}$ defined on the state domain boundary $\partial\Omega$ makes the solution unique [7]. This boundary condition implies that the optimal action always prevent the system to leave the state domain. Within the LQR-setting, this boundary condition implies the positive-definiteness of the quadratic value function. Using the control affine dynamics of mechanical systems with holonomic constraints, i.e., $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \mathbf{a}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{u}$, Equation 3 and Equation 4 can be further simplified to

$$\rho \, V^* = \mathbf{a}^T(\mathbf{x}) \, V_x^* + \min_{\mathbf{u}} \left[c(\mathbf{x}, \mathbf{u}) + \mathbf{u}^T \, \mathbf{B}^T(\mathbf{x}) \, V_x^* \right], \qquad \left(\mathbf{a}(\bar{\mathbf{x}}) + \mathbf{B}(\bar{\mathbf{x}})\mathbf{u}^* \right)^T \boldsymbol{\eta}(\bar{\mathbf{x}}) \leq 0. \quad (5)$$

3 Incorporating Action Constraints within the Cost Function

Assuming a separable cost, i.e., a separate cost function for state and action, we show that the optimization problem on the right-hand side of Equation 5 can be solved in closed form if the action cost is strictly convex. Moreover, action constraints can be naturally accommodated if the action cost has a barrier shape. Thereby, the constrained optimization in Equation 1 is transformed into an unconstrained problem, with the optimal policy inherently obeying the action limits.

Table 1: Selected action costs. The choice of the action cost $g(\mathbf{u})$ determines the range of actions $\mathbf{u} \in \text{dom}(g)$, as well as the form of the optimal policy $\nabla g^*(\mathbf{w})$ and the type of non-linearity in the HJB equation $g^*(\mathbf{w})$. Section 1 of the table contains policies with standard action domains. Section 2 provides formulae for shifting and scaling actions and scaling costs. Section 3 shows how to use the formulae. Section 4 gives limiting policy shapes.

Policy Name	Action Range	Action Cost $g(\mathbf{u})$	Policy $\nabla g^*(\mathbf{w})$	HJB Nonlinear Term $g^*(\mathbf{w})$
Linear	$\mathbf{u} \in \mathbb{R}^{n_u}$	$\frac{1}{2}\mathbf{u}^T\mathbf{R}\mathbf{u}$	$\mathbf{R}^{-1}\mathbf{w}$	$\frac{1}{2}\mathbf{w}^T\mathbf{R}^{-1}\mathbf{w}$
Logistic	$\mathbf{0} < \mathbf{u} < \mathbf{1}$	$\mathbf{u}^T\log\mathbf{u} + (1-\mathbf{u})^T\log(1-\mathbf{u})$	$\frac{1}{1+e^{-\mathbf{w}}} =: \sigma(\mathbf{w})$	$\mathbf{1}^T\log(1+e^{\mathbf{w}})$
Atan	$-\frac{\pi}{2}\mathbf{1} < \mathbf{u} < \frac{\pi}{2}\mathbf{1}$	$-\log\cos\mathbf{u}$	$\tan^{-1}(\mathbf{w})$	$\mathbf{w}^T\tan^{-1}(\mathbf{w}) - \frac{1}{2}\mathbf{1}^T\log(1+\mathbf{w}^2)$
Action-Scaled	$\mathbf{u} \in \alpha\,\text{dom}(g)$	$\alpha g(\alpha^{-1}\mathbf{u})$	$\alpha\nabla g^*(\mathbf{w})$	$\alpha g^*(\mathbf{w})$
Cost-Scaled	$\mathbf{u} \in \text{dom}(g)$	$\beta g(\mathbf{u})$	$\nabla g^*(\beta^{-1}\mathbf{w})$	$\beta g^*(\beta^{-1}\mathbf{w})$
Action-Shifted	$\mathbf{u} \in \text{dom}(g) - \gamma\mathbf{1}$	$g(\mathbf{u}+\gamma\mathbf{1}) - g(\gamma\mathbf{1})$	$\nabla g^*(\mathbf{w}) - \gamma\mathbf{1}$	$g^*(\mathbf{w}) - \gamma\mathbf{1}^T\mathbf{w}$
Tanh	$-\mathbf{1} < \mathbf{u} < \mathbf{1}$	$g_{\text{logistic}}(\frac{\mathbf{u}+1}{2}) - g_{\text{logistic}}(\frac{1}{2})$	$\tanh\mathbf{w} = 2\sigma(2\mathbf{w}) - \mathbf{1}$	$\mathbf{1}^T\log\cosh\mathbf{w}$
TanhActScaled	$-\alpha\mathbf{1} < \mathbf{u} < \alpha\mathbf{1}$	$\alpha g_{\tanh}(\alpha^{-1}\mathbf{u})$	$\alpha\tanh\mathbf{w}$	$\alpha\mathbf{1}^T\log\cosh\mathbf{w}$
AtanActScaled	$-\alpha\mathbf{1} < \mathbf{u} < \alpha\mathbf{1}$	$-\frac{2\alpha}{\pi}\log\cos(\frac{2\alpha}{\pi}\mathbf{u})$	$\frac{2\alpha}{\pi}\tan^{-1}(\mathbf{w})$	$\frac{2\alpha}{\pi}g^*_{\text{atan}}(\mathbf{w})$
Bang-Bang	$-\mathbf{1} \le \mathbf{u} \le \mathbf{1}$	$\chi_{[-1,1]}(\mathbf{u})$, χ - charact. fun.	$\text{sign}\,\mathbf{w}$	$\|\mathbf{w}\|_1$
Bang-Lin	$-\mathbf{1} \le \mathbf{u} \le \mathbf{1}$	$\frac{1}{2}\mathbf{u}^T\mathbf{u}\,\chi_{[-1,1]}(\mathbf{u})$	$-\mathbf{1}+\sum_{\delta=-1}^{1}\text{relu}(1-\delta\mathbf{w})$	$\mathbf{1}^T L_1(\mathbf{w})$, $L_\delta(a)$ - Huber loss

3.1 Strictly Convex Action Cost

Let the separable cost be defined as $c(\mathbf{x}, \mathbf{u}) = r(\mathbf{x}) + g(\mathbf{u})$, with the task dependent state cost $r(\mathbf{x})$ and the strictly convex action cost $g(\mathbf{u})$. The strong convexity of g generalizes the positive definiteness assumption on the action cost required by LQR. Under these assumptions, the HJB equation becomes

$$\rho\, \mathrm{V}^* = \min_{\mathbf{u}} \left[g(\mathbf{u}) + \mathbf{u}^T\mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}} \right] + r(\mathbf{x}) + \mathbf{a}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}}. \tag{6}$$

The optimal action and the optimal policy can be computed in closed form by employing the convex conjugate function $g^*(\mathbf{w}) = \sup_{\mathbf{u}}\{\mathbf{u}^T\mathbf{w} - g(\mathbf{u})\}$ and exploiting its defining property $\nabla g^* = (\nabla g)^{-1}$,

$$\frac{\partial}{\partial\mathbf{u}}\left[g(\mathbf{u}) + \mathbf{u}^T\mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}} \right] = \nabla g(\mathbf{u}) + \mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}} := 0 \quad \Rightarrow \quad \mathbf{u}^* = \nabla g^*(-\mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}}). \tag{7}$$

The strict convexity of g assures that Equation 7 provides a unique global minimum [8]. Importantly, Equation 7 describes the optimal policy in closed form and therefore no learning of the policy is required. The value function is also a Lyapunov function and hence, the policy is stable [9]. Substituting the optimal action $\mathbf{u}^*$ into the HJB and using the Fenchel-Young identity $g(\nabla g^*(\mathbf{w})) + g^*(\mathbf{w}) = \mathbf{w}^T\nabla g^*(\mathbf{w})$ with $\mathbf{w} = -\mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}}$, we arrive at the final form of the HJB equation

$$\rho\, \mathrm{V}^* = r(\mathbf{x}) + \mathbf{a}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}} - g^*\left(-\mathbf{B}^T(\mathbf{x})\mathrm{V}^*_{\mathbf{x}}\right). \tag{8}$$

Notably, the HJB equation in the form of Equation 8 is a straightforward equality and does not contain a nested optimization problem in contrast to the original formulation in Equation 5. Therefore, one only needs to find the optimal value function and then the optimal feedback controller is directly given.

3.2 Torque Limited Optimal Policies

Exploiting the closed form solution for the optimal policy given in Equation 7 and the convex conjugacy, one can define cost functions such that the standard controllers, including torque limited controllers, become optimal. This approach is favorable compared to the naive quadratic action cost because such cost can potentially take unbounded actions. Clipping the unbounded actions is only optimal for linear systems [10] and increasing the action cost to ensure the action limits leads to over-conservative behavior and underuse of the control range.

The shape of the optimal policy is determined by the monotone function ∇g^*. Therefore, one can define any desired monotone shape and determine the corresponding strictly convex cost by inverting ∇g^* to compute ∇g and integrating ∇g to obtain the strictly convex cost function $g(\mathbf{u})$. For example, the linear policy is optimal with respect to the quadratic action cost, whereas the logistic policy is optimal with respect to the binary cross-entropy cost. The full generality of this concept based on convex conjugacy is shown in Table 1, which shows the corresponding cost functions for Linear, Logistic, Atan, Tanh and Bang-Bang controllers. Furthermore, using the rules from convex analysis [8], the effects of scaling the action limits, shifting the action range, or scaling the action cost can be succinctly described, as shown by Action-Scaled, Action-Shifted, and Cost-Scaled rows in
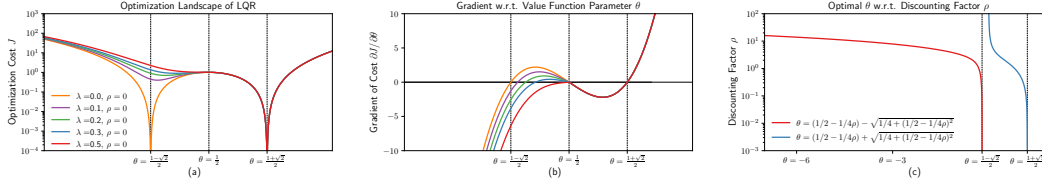
Figure 1: (a) The optimization landscape for the linear system $\dot{x} = x + u$, the cost $c(x, u) = 1/2x^2 + 1/2u^2$ and value function $V(x) = \theta x^2$ with different penalty terms λ. For sufficiently large penalties the negative definite minima disappears. (b) The gradient $\nabla_\theta J$ showing the extrema for different penalty terms. (c) Optimal value function parameter θ for different discounting factors ρ. For $\rho \to \infty$ the negative definite parameter diverges to $-\infty$, while the positive definite solution converges to 0.

Table 1. This enables quick experimentation by mixing and matching costs. For example, the action cost corresponding to the Tanh policy is straightforwardly derived using the well-known relationship between $\tanh(x)$ and the logistic sigmoid $\sigma(x)$ given by $\tanh(x) = 2\sigma(2x) - 1$. Note that a formula for general invertible affine transformations can be derived, not only for scalar scaling and shifting. Classical types of hard nonlinearities [11] can be derived as limiting cases of smooth solutions. For example, taking the Tanh action cost g_{tanh} and scaling it with $\beta \to 0$, i.e., putting a very small cost on actions but nevertheless preserving the action limits, results in the Bang-Bang control shape. Taking a different limit of the Tanh policy in which scaling is performed simultaneously with respect to the action and cost, the resulting shape is what we call Bang-Lin and corresponds to a function which is linear around zero and saturates for larger input values.

4   Learning the Value Function with Differential Networks

To obtain the optimal policy, one must solve the differential equation described in Equation 8 to obtain the optimal value function. Learning this value function using function approximation is non-trivial because the equation contains both the value function as well as the Jacobian and has multiple solutions. The first problem can be addressed, by using a differential network, which previous research used to learn the parameters of the Euler-Lagrange differential equation [12], while the latter prevents the naive optimization of the HJB. Therefore, we first introduce the differential network and the naive optimization loss in Section 4.1 and describe the curricular optimization scheme to learn the unique solution in Section 4.2.

4.1   Deep Differential Network

Deep networks are fully differentiable and one can compute the partial derivative w.r.t. networks at machine precision [13]. Therefore, deep networks are well suited for being embedded within differential equations and learning the solution end-to-end. The deep differential network architecture initially introduced by [12], computes the functional value and the Jacobian w.r.t. to the network inputs within a single forward-pass by adding a additional graph within each layer to directly compute the Jacobian using the chain rule. Therefore, this architecture can be naturally used to model V and $V_{\mathbf{x}}$ and be learned end-to-end by minimizing the error of the HJB using standard deep learning techniques. In addition, this architecture enables the fast computation of the Jacobian s.t. this network can be used for real-time control loops with up to 500Hz [14]. Let $\hat{V}(\mathbf{x}; \psi)$ be the deep network with the network parameters ψ representing the approximated value function and approximated partial derivative. This deep value function can be trained by minimizing the residual of the HJB equality (3) described by

$$\hat{V}^*(\mathbf{x}; \psi) = \min_{\psi} \frac{1}{N} \sum_i^N \left| \rho \hat{V}(\mathbf{x}_i) + g^* \left(-\mathbf{B}^T(\mathbf{x}_i)\hat{V}_{\mathbf{x}_i} \right) - r(\mathbf{x}_i) - \mathbf{a}^T(\mathbf{x}_i)\hat{V}_{\mathbf{x}_i} \right| \qquad (9)$$

where $\mathbf{x}_i$ is uniformly sampled from the state domain Ω.

4.2   Constrained Optimization of the Value Function

The naive optimization of the loss of Equation 9 using gradient descent is not sufficient because the HJB has multiple solutions. Even for the one dimensional linear system with quadratic rewards and quadratic value function, i.e., $\dot{x} = Ax + Bu$, $r(x, u) = Qx^2 + Ru^2$ with $Q, R > 0$ and $V(x) = \theta x^2$, the

HJB equation has two solutions described by

$$V(x) = \frac{1}{2}B^{-2}\left((2A-\rho)R \pm \sqrt{(2A-\rho)^2 R^2 + 4RQB^2}\right)x^2.$$

The problem of multiple solutions can be addressed from two perspectives. First one can enforce the boundary constraint to make the solution unique. Second, one can use the a curricular learning scheme to change the discounting factor during the learning from short to far sighted. In addition, one can exploit the knowledge of terminal states and can locally clamp the value function at the terminal states to the cost function. Similar approaches have been suggested in [15, 16].

4.2.1 Boundary Constraint
The solution of the HJB for the 1d linear system is unique, when the boundary condition of Equation 4 is included, i.e.,

$$\eta(\bar{x}_+)\left(\bar{x}_+ - \frac{1}{2}R^{-1}V_x(\bar{x}_+)\right) = 1 - \frac{1}{2}R^{-1}B^{-2}\left((2A-\rho)R \pm \sqrt{(2A-\rho)^2 R^2 + 4RQB^2}\right)\bar{x}_+^2 \leq 0$$

with $\bar{x}_+ > 0$. Therefore, incorporating this constraint within the learning objective should enforce the desired optimal solution. The boundary constraint can be incorporated as additional penalty term, i.e., $\ell = \ell_1 + \lambda \ell_b$ with

$$\ell_b = \max\left(0, \left(\mathbf{a}(\bar{\mathbf{x}}) + \mathbf{B}(\bar{\mathbf{x}})\mathbf{u}^*\right)^T \boldsymbol{\eta}(\bar{\mathbf{x}})\right)^2 \tag{10}$$

and optimizing this objective yields the desired optimal solution. For the integrator dynamics, the optimization landscape is smoothed and only one minima remains given a sufficiently large penalty term λ (Fig. 1 a). Despite the well posed optimization surface, such constraint is hard to draft even for slightly more complex systems as the double integrator. Defining the state domain, which covers both position and velocity is non-trivial because the boundary constraint implies that the system must be controllable on the given domain boundary. Therefore, one must manually engineer the maximum region of attraction, which is dependent on the cost function $c(\mathbf{x}, \mathbf{u})$ and their relative magnitudes. E.g., the relation of action cost to velocity cost determines the magnitude of deceleration for a moving object and hence, the slope of the boundary constraining positions and velocities. These different slopes cause the domain boundary to be specific to the cost function. Furthermore, one must account for states in which the system is not controllable. If wrong boundary conditions are incorporated, the optimization objectives contradict and one converges pre-maturely. Furthermore, this boundary constraint is only enforced locally and hence, the deep network may be attracted to other solutions inside the domain. Both drawbacks, the necessary engineering and locality, render the boundary constraint not useful for the learning of the HJB.

4.2.2 Changing the discounting from short to far-sighted
Besides adding the boundary constraint, one can continuously decrease the discounting ρ such that the value function changes from short to far-sighted solutions. This can be achieved by first initializing $\rho \gg 1$ and slowly decreasing ρ once the value function is sufficiently learned. Thereby, the value function is initially attracted to a single solution and follows this solution closely through the parameter space when the discounting factor is decreased. This initial solution is unique, because taking the limit $\rho \to +\infty$ for Equation 3 shows that only one unique finite value function, i.e., $V(\mathbf{x}) = 0, \forall \mathbf{x} \in \Omega$, exists. Therefore, the deep network is initially attracted to the desired optimal value functions and follows this solution closely when decreasing ρ. Applying this limit to the 1d linear system example,

$$\lim_{\rho \to \infty} \frac{1}{2}\left((2A-\rho)R \pm \sqrt{(2A-\rho)^2 R^2 + 4RQB^2}\right)x^2 = 0 / -\infty,$$

shows that the undesired solution diverges to $-\infty$, while the desired optimal solution approaches $\theta \to 0$. This divergence is also shown in Figure 1 c. Therefore, the value function is attracted to the desired optimal solution and follows this solution to the undiscounted infinite horizon value function. This learning scheme of changing the parameter ρ during optimization can be interpreted as continuation method [17] and curricular learning [18]. Both approaches gradually increase the task complexity to achieve faster convergence (as curriculum learning) or the avoidance of bad local optima (continuation methods). Especially, continual learning solves an initially simplified non-linear optimization problem and tracks this solutions through parameter space when the task complexity increases. Therefore, gradually decreasing ρ and tracking the short-sighted $V(\mathbf{x}) = 0$ solution through weight space to the potentially discontinuous undiscounted infinite horizon value function is comparable to continuation methods. We only use a heuristic of decreasing ρ when an error or epoch threshold is reached and aim to apply the principled methods of continuation methods for adapting ρ in future work.
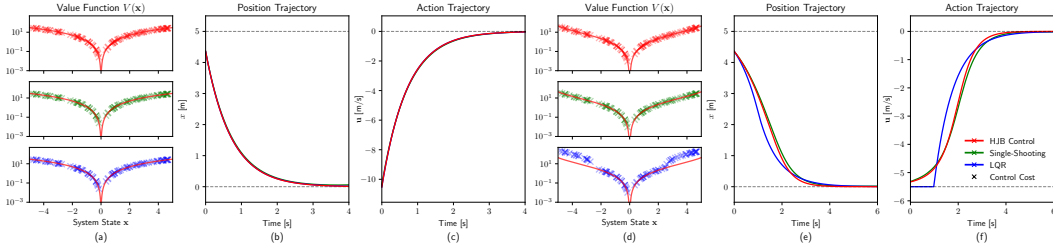
Figure 2: (a) The learned value function for the linear integrator with quadratic cost. The crosses correspond to the closed-loop control cost. The achieved cost for LQR, single shooting and HJB control align with the learned value function. (b-c) State and action trajectory of the integrator with quadratic cost. LQR, single shooting and HJB control achieve optimal performance (c) The learned value function for the linear integrator with log-cosine cost incorporating the action limit. Single shooting and HJB control achieve the optimal cost, which matches the expected cost of the value function. (e-f) State and action trajectory of the integrator with log-cosine cost. Single shooting and HJB control achieve optimal control, while clipped LQR is not optimal.

4.3 Terminal States

One can exploit the knowledge of the terminal states to direct the learning of the value function. For every terminal state $\mathbf{x}_T$, the value function must be equal to the cost function, i.e. $V^*(\mathbf{x}_T) = r(\mathbf{x}_T)$. This known point of the value function can be incorporated by extending the original loss with an additional penalty term. In addition, this penalty term can also be added for any known regularities, e.g., constraining the the multiples of $2\pi\,k$ of a continuous revolute joint to be identical.

5 Experiments

The proposed approach for learning optimal feedback control is evaluated on the stabilization of a one-dimensional linear system, the swing-up of a torque-limited pendulum and the balancing of a flexible Cartpole[1]. For the quadratic action cost, the actions are not limited, while the log-cosine action cost implicitly limits the actions. In the following, we will refer to our proposed approach of learning the optimal value function and applying the closed-form policy as *HJB control*. The performance is compared to LQR and shooting methods[2] augmented with hand-tuned tracking controllers.

5.1 Linear System

The learned value function of the linear system and the corresponding control performance is shown in Figure 2. For the quadratic cost function, HJB control, LQR and single shooting obtain identical state and action trajectories (Fig. 2 b-c). For randomly sampled starting configurations, the learned value function and the accumulated cost of the HJB controller match the cost of LQR and single shooting (Fig. 2 a). For the log-cosine cost, only HJB control and single shooting achieve optimal performance on the complete state domain while LQR only achieves comparable cost for limited starting configurations, when the action limits are not active. This can be seen in Figure 2 d, where the expected control cost of the learned value function and the accumulated cost of the HJB controller match the cost of single shooting. In contrast, LQR achieves comparable cost for $x_0 \leq 2.5$ but significantly larger cost for $x_0 > 2.5$.

5.2 Torque Limited Pendulum

The learned value function of the non-linear torque limited pendulum and the corresponding control performance of the HJB controller for both the quadratic and log-cosine cost is shown in Figure 3. For the quadratic action cost, the value function is continuous and only locally quadratic within the surrounding of the balancing point. The trajectories of HJB control, multiple shooting and LQR from 300 randomly sampled starting configurations $\mathbf{x}_0$ are shown in Figure 3 a-c. The trajectories from HJB control match the trajectories of multiple shooting and hence are optimal for the non-linear system. In contrast, the LQR controller applies unnecessarily large actions because the system linearization at $\theta = 0$ over-estimates the system dynamics. Furthermore, the corresponding cost distribution for the starting configurations is very similar for the HJB controller and multiple shooting, while the LQR cost distributions shows that LQR requires larger cost for some starting configurations (Fig. 3 d). For

---

[1]The exact system descriptions including the system parameters is explained in the appendix.
[2]Single and multiple shooting is implemented using CasADi [19]
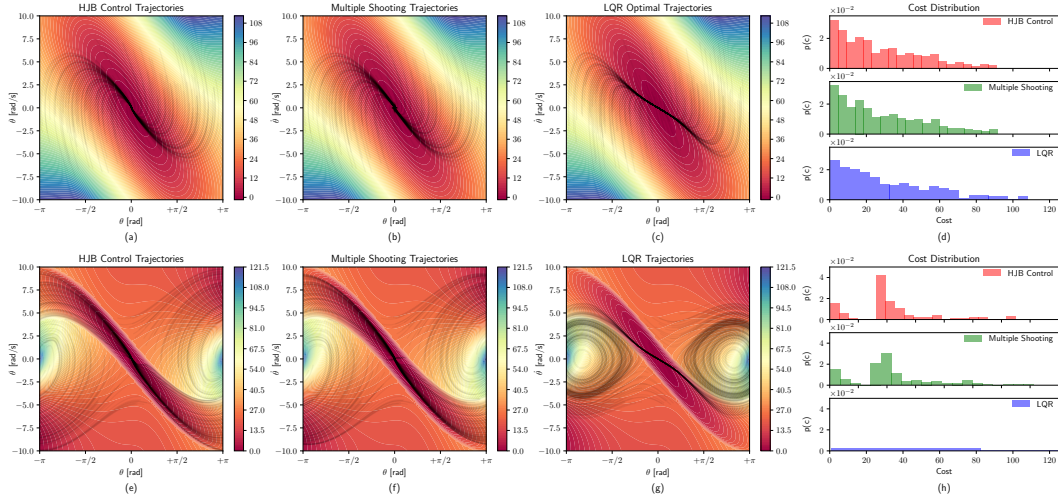
Figure 3: (a-c) Learned value function for the pendulum with quadratic cost and the trajectories for HJB control (a), multiple shooting (b) and LQR (c) from 300 randomly sampled starting configurations. (d) Cost distributions $p(c)$ for the sampled starting configurations. The distributions for HJB control and multiple shooting are similar while the distribution of LQR requires higher cost for some starting configurations. (e-g) Learned discontinuous value function for the pendulum with log-cosine cost and the trajectories for HJB control (e), multiple shooting (f) and LQR (g). HJB control and multiple shooting can swing-up the pendulum from every sampled starting configuration, while LQR can only balance the starting configurations for which the action constraint is not active. (h) The cost distributions $p(c)$ for HJB control and multiple shooting match closely while LQR only achieves low reward for very few starting configurations.

the log-cosine cost, which implicitly limits the feasible actions, the value function is discontinuous and contains two ridges leading to the balancing point, because the balancing point cannot be reached directly from every point within the state domain. The learned value function and the trajectories of HJB control, multiple shooting and LQR are shown in Figure3 e-g. Multiple shooting and HJB control achieve the swing-up from every starting configuration while LQR cannot swing-up the pendulum and only achieves balancing of starting configurations close to the upright position. Furthermore, the cost distributions for HJB control and multiple shooting match closely (Fig. 3 h). This close similarity between cost distribution shows that our proposed approach learned the optimal value function, and the corresponding feedback policy achieves optimal feedback control.

5.3 Flexible Cartpole
The flexible Cartpole is a Cartpole with an additional spring within the linear actuator. Therefore, the cart with the pendulum is not directly actuated and the force must be transmitted through the spring to balance the pendulum. Figure 4 shows the optimal actions computed using HJB control, LQR and multiple shooting as well as the corresponding six dimensional state trajectories consisting of cart position x_c, spring displacement x_s and pendulum angle θ. The action and state trajectories (Fig. 4 a-d) as well as the cost distribution from 300 different starting configurations (Fig. 4 e) overlap closely for HJB control and LQR, which is optimal for this linearized system and quadratic cost. Only the multiple shooting baseline does a slightly larger compensatory movement of the cart due to the hand-tuned feedback gains. These hand-tuned gains also cause the larger cost for the multiple shooting baseline visible in Figure 4 e. This similarity of LQR and HJB control shows that HJB control has learned the optimal value function and demonstrates that HJB control is also applicable to six dimensional problems.

6 Related Work
Non-linear control problems are normally solved by trajectory optimization with explicit inequality constraints on the actions [20]. These approaches yield a single optimal trajectory, that needs to be replanned for every initial configuration and augmented with a non-optimal tracking controller. Locally-optimal tracking controller can be obtained when using iterative linear quadratic programming (iLQR) with action constraints [21] or guided policy search (GPS) [22]. In contrast to these approaches,
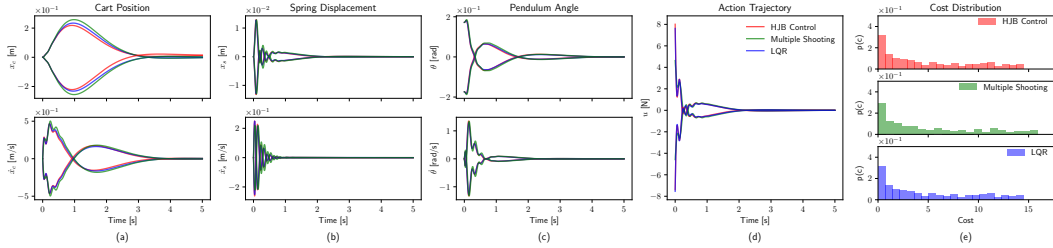
Figure 4: (a-d) State and action trajectories of the flexible Cartpole from two different starting configurations using the control laws derived by HJB control, multiple shooting and LQR. (d) Cost distributions $p(c)$ from 300 uniformly sampled starting configurations. The achieved cost for HJB control and LQR is similar while the achieved cost for multiple shooting is slightly higher due to the hand-tuned feedback gains.

our proposed method transforms the constrained problem to an unconstrained problem by using principled cost function and provides the globally optimal feedback controller on the domain Ω.

The transformation of the inequality constraints to the principled cost function has been explored by a number of authors. Historically the first mention of generalized convex action costs goes back to [23]. Thereafter, the tanh non-linearity derived from the corresponding action cost has been commonly used in the adaptive dynamic programming literature [24, 25, 26]. Furthermore, [6] and [16] present similar derivation to ours, with a generic convex action cost. However, these papers do not arrive at the general convex conjugate theory as these papers only treat the tanh case and do not derive the HJB in the form of Equation 8 using the convex conjugate g^*.

Global optimal feedback controllers were previously learned by using the least squares solution of the HJB differential equation [6, 16, 25, 27]. Early on, [6] used radial-basis-function networks to learn the HJB. Similarly, [25, 27] learned polynomial value function using gradient descent. Most similar to our approach, [16] used neural networks of the Pineda architecture to learn the average cost solution of the HJB. To achieve convergence the authors added domain constraints, adapted the discounting horizon and added stochasticity to the dynamics to smoothen the value function. In contrast to the previous works, our proposed approach learns the value function using a generic deep differential network capable of representing discontinuous value functions and only requires the adaptation of the discounting factor to achieve robust convergence. Furthermore, we provide intuitive insights that show how changing the discounting factor enables the learning of the optimal value function.

7 Conclusion

In this paper, we showed that constrained optimization problems for finding action-limited optimal policies can be transformed into an unconstrained problem by designing principled strictly convex cost-functions, which guarantee optimal policies that naturally obey the actuator limits. Exploiting this transformation as well as the affine control dynamics of mechanical systems with holonomic constraints, we showed that the optimal policy can be described in closed form given the differentiable value function. The differentiable value function can be learned by embedding a deep differential network within the HJB and using a curricular learning approach. This curriculum changes the discounting from short- to far-sighted to achieve robust convergence to the desired value function and avoids the undesired solutions of the HJB, which violate the boundary conditions. The experiments demonstrated that our approach can learn the optimal feedback controller for non-linear and torque limited systems. Furthermore, HJB control achieves similar performance as shooting methods but does not require the replanning for different starting configurations and the hand-tuning of the gains.

In future work, we plan to extend this work to model-based reinforcement learning by combining the HJB optimal feedback control with model learning in the form of Deep Lagrangian Networks [12], which ensures learning control-affine dynamics. Iterating between optimal control and model learning, the optimal policy and model are simultaneously learned. Besides the model learning aspect, we plan to develop an iterative sampling scheme to construct a domain limited to the surroundings of the current policy. This reduced domain should enable the scaling of our proposed approach to high dimensional problems where obtaining the global solution is not feasible.

Acknowledgement

References

[1] I. Mordatch, Z. Popović, and E. Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144. Eurographics Association, 2012.

[2] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

[3] M. Toussaint, K. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems*, 2018.

[4] I. Mordatch, E. Todorov, and Z. Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics*, 31(4):43, 2012.

[5] N. Heess, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, M. Riedmiller, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

[6] K. Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1): 219–245, 2000.

[7] W. H. Fleming and H. M. Soner. *Controlled Markov processes and viscosity solutions*, volume 25. Springer Science & Business Media, 2006.

[8] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[9] D. Liberzon. *Calculus of variations and optimal control theory: a concise introduction*. Princeton University Press, 2011.

[10] J. A. De Doná and G. C. Goodwin. Elucidation of the state-space regions wherein model predictive control and anti-windup strategies achieve identical control policies. In *Proceedings of the 2000 American Control Conference*, volume 3, pages 1924–1928. IEEE, 2000.

[11] S. Ching, Y. Eun, C. Gokcek, P. T. Kabamba, and S. M. Meerkov. *Quasilinear Control: Performance Analysis and Design of Feedback Systems with Nonlinear Sensors and Actuators*. Cambridge University Press, 2010.

[12] M. Lutter, C. Ritter, and J. Peters. Deep lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2019.

[13] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[14] M. Lutter and J. Peters. Deep lagrangian networks for end-to-end learning of energy-based control for under-actuated systems. In *International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[15] M. Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer, 2005.

[16] Y. Tassa and T. Erez. Least squares solutions of the hjb equation with neural network value-function approximators. *IEEE transactions on neural networks*, 18(4):1031–1041, 2007.

[17] E. L. Allgower and K. Georg. *Numerical continuation methods: an introduction*, volume 13. Springer Science & Business Media, 2012.

[18] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[19] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, In Press, 2018.

[20] A. E. Bryson and Y.-C. Ho. Applied optimal control, revised printing. *Hemisphere, New York*, 1975.

[21] Y. Tassa, N. Mansard, and E. Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE, 2014.

[22] S. Levine and V. Koltun. Guided policy search. In *International Conference on Machine Learning*, pages 1–9, 2013.

[23] S. E. Lyshevski. Optimal control of nonlinear continuous-time systems: design of bounded controllers via generalized nonquadratic functionals. In *Proceedings of the 1998 American Control Conference*, volume 1, pages 205–209. IEEE, 1998.

[24] M. Abu-Khalaf and F. L. Lewis. Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network hjb approach. *Automatica*, 41(5):779–791, 2005.

[25] X. Yang, D. Liu, and D. Wang. Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints. *International Journal of Control*, 87 (3):553–566, 2014.

[26] H. Modares, F. L. Lewis, and M.-B. N. Sistani. Online solution of nonquadratic two-player zero-sum games arising in the h∞ control of constrained input systems. *International Journal of Adaptive Control and Signal Processing*, 28(3-5):232–254, 2014.

[27] D. Liu, D. Wang, F.-Y. Wang, H. Li, and X. Yang. Neural-network-based online hjb solution for optimal robust guaranteed cost control of continuous-time uncertain nonlinear systems. *IEEE transactions on cybernetics*, 44(12):2834–2847, 2014.

Appendix

Analytic Derivation of the 1d LQR System

Let the system dynamics, cost function and value function be defined as $\dot{x} = Ax + Bu$, $c(x, u) = Qx^2 + Ru^2$, $V^* = \theta x^2$ and $V^*_x = 2\theta x$. Then Equation 8 is described by

$$\rho\, V^* + g^*\left(-B(x)\,V^*_x\right) - r(x) - a(x)\,V^*_x = 0 \qquad \forall x \in \mathbb{X} \subseteq \mathbb{R}$$

$$\rho\, V^* + \frac{1}{4}R^{-1}\left(B\,V^*_x\right)^2 - Qx^2 - A\,V^*_x\,x = 0$$

$$\rho\theta x^2 + \frac{1}{4}R^{-1}(2B\theta x)^2 - Qx^2 - 2A\theta x^2 = 0$$

$$\left(R^{-1}B^2\theta^2 + (\rho - 2A)\theta - Q\right)x^2 = 0.$$

Solving the quadratic equation yields the optimal parameter of the value function described by

$$\theta = \frac{1}{2}B^{-2}\left((2A - \rho)\,R \pm \sqrt{(2A - \rho)^2\,R^2 + 4RQB^2}\right).$$

System Description

This section describes the exact system specification used for simulation. All simulations were conducted with 500Hz.

1d-Linear System: The dynamics of the linear system are described by $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ and the state cost is quadratic $r(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}\mathbf{x}$. For the experiments, a simple one-dimensional integrator with the parameters $A = B = 1$, $Q = 1/2$ and the domain $-5 \leq x \leq +5$ is used. For the quadratic cost function the actions are unconstrained, while for the log-cosine cost the actions are constrained to $\mathbf{u}_+ = 5.5$.

Torque Limited Pendulum: The torque limited pendulum is a one-degree-of-freedom non-linear system with the joint position θ, velocity $\dot{\theta}$ and torque $\mathbf{u}$. For the experiments the canonical system representation with the state $\mathbf{x} = [\theta, \dot{\theta}]$ with $\theta = 0$ being the upward pointing pendulum is used. The corresponding equations of motions are described by

$$\ddot{\theta} = \frac{3}{ml^2}\left[\mathbf{u} - \frac{mgl}{2}\sin(\theta)\right], \qquad \mathbf{a} = \left[\dot{\theta}, \pm\frac{3g}{2l}\sin(\theta)\right], \qquad \mathbf{B} = \left[0, \frac{3}{ml^2}\right], \qquad (11)$$

with the pendulum mass $m = 1$kg, the length $l = 1$m and the gravitational constant $g = -9.81$m/s^2. The state cost is given by $r(\mathbf{x}) = q_0\pi^2\sin(\theta/2) + q_1\dot{\theta}^2$. For the log-cosine cost, the torque is constrained to $\mathbf{u}_+ = 2.5$N/m.

Flexible Cartpole: The flexible Cartpole has in addition to the normal Cartpole a spring within the linear actuator. This flexible actuator is modelled by two separate carts connected by a spring,

whereas only one cart is actuated. Therefore, the controller must excite/compensate the spring to balance the pendulum. The six dimensional system state is described by $\mathbf{x} = \left[x_c,\ x_s,\ \theta,\ \dot{x}_c,\ \dot{x}_s,\ \dot{\theta}\right]$, with the actuated cart position x_c, the spring displacement x_s and the pendulum angle θ. For the balancing experiments, the system is approximated using a linear system described by $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$. The system matrices are described by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \frac{K_s}{m_a} & 0 & -\frac{B_a}{m_a} & 0 & 0 \\ 0 & -\frac{K_s}{m_a} - \frac{K_s}{m_a} & \frac{m_\theta g}{m_p} & \frac{B_a}{m_a} - \frac{B_p}{m_p} & -\frac{B_p}{m_p} & -\frac{B_p}{m_p l_\theta} \\ 0 & -\frac{K_s}{m_p l_\theta} & \frac{g(m_p + m_\theta)}{m_p l_\theta} & -\frac{B_p}{m_p l_\theta} & -\frac{B_p}{m_p l_\theta} & -\frac{B_\theta(m_p + m_\theta)}{m_p m_\theta l_\theta^2} \end{bmatrix}$$

$$\mathbf{B}^T = \begin{bmatrix} 0 & 0 & 0 & \frac{1}{m_c} & -\frac{1}{m_c} & 0 \end{bmatrix},$$

with the masses $m_a = 0.57$kg, $m_p = 0.375$kg, $m_\theta = 0.127$kg, the spring constant $K_s = 200$N/m, pendulum length $l_\theta = 0.1778$m, the viscous friction coefficients $B_a = B_p = 0.5$Ns/m, $B_\theta = 0.0024$Nms/rad and the gravitational constant $g = 9.81$m/s^2.