arXiv:1706.06491v1 [cs.SY] 20 Jun 2017

Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control

Sanket Kamthe
Department of Computing
Imperial College London
s.kamthe15@imperial.ac.uk

Marc Peter Deisenroth*
Department of Computing
Imperial College London
m.deisenroth@imperial.ac.uk

Abstract

Trial-and-error based reinforcement learning (RL) has seen rapid advancements in recent times, especially with the advent of deep neural networks. However, the majority of autonomous RL algorithms either rely on engineered features or a large number of interactions with the environment. Such a large number of interactions may be impractical in many real-world applications. For example, robots are subject to wear and tear and, hence, millions of interactions may change or damage the system. Moreover, practical systems have limitations in the form of the maximum torque that can be safely applied. To reduce the number of system interactions while naturally handling constraints, we propose a model-based RL framework based on Model Predictive Control (MPC). In particular, we propose to learn a probabilistic transition model using Gaussian Processes (GPs) to incorporate model uncertainties into long-term predictions, thereby, reducing the impact of model errors. We then use MPC to find a control sequence that minimises the expected long-term cost. We provide theoretical guarantees for the first-order optimality in the GP-based transition models with deterministic approximate inference for long-term planning. The proposed framework demonstrates superior data efficiency and learning rates compared to the current state of the art.

1 Introduction

Reinforcement learning (RL) is a principled mathematical framework for experienced-based autonomous learning of policies. The trial-and-error learning process is one of the most distinguishing features of RL [40]. Despite many recent advances in RL [24, 39, 44], a main limitation of current RL algorithms remains its data inefficiency, i.e., the required number of interactions with the environment is impractically high. For example, many RL approaches in problems with low-dimensional state spaces and fairly benign dynamics require thousands of trials to learn. This *data inefficiency* makes learning in real control/robotic systems without task-specific priors impractical and prohibits RL approaches in more challenging scenarios.

A promising way to increase the data efficiency of RL without inserting task-specific prior knowledge is to learn models of the underlying system dynamics. When a good model is available, it can be used as a faithful proxy of the real environment, i.e., good policies can be obtained from the model without additional interactions with the real system. However, learning models of the underlying transition dynamics is hard and inevitably leads to model errors. To account for model errors, it has been proposed to use probabilistic models [38, 8]. By explicitly taking model uncertainty into account, the number of interactions with the real system can be substantially reduced. For example, in [9, 29, 8, 7], the authors use Gaussian processes (GPs) to model the dynamics of the underlying system. The PILCO algorithm [9] propagates uncertainty through time for long-term planning and

*Also with PROWLER.io

learns parameters of a feedback policy by means of gradient-based policy search. It achieves an unprecedented data efficiency for learning control policies for from scratch.

While the PILCO algorithm is very data efficient, it possesses some shortcomings: 1) Learning closed-loop feedback policies requires looking at the full planning horizon to stabilise the system, which results in a significant computational burden. 2) PILCO requires us to specify a parametrised policy a priori. Typically, an RBF policy is used, which possesses hundreds of parameters. 3) PILCO handles control constraints by using a differentiable squashing function that is applied to the RBF policy. This allows us to explicitly take control constraints into account during planning. However, this kind of constraint handling can produce unreliable predictions near constraint boundaries [37, 25].

In this paper, we develop an RL algorithm that is a) data efficient, b) does not require to look at the full planning horizon, c) handles constraints naturally, d) does not require a parametrised policy, e) is theoretically justified. The key idea of our method is to show that we can find optimal trajectories in a constrained setting and improving the data efficiency over PILCO. In particular, we reformulate the optimal control problem with learned GP models as an equivalent deterministic problem. This reformulation allows us to exploit Pontryagin's maximum principle to find optimal control signals, while handling constraints in a principled way. We use model predictive control (MPC) with learned GP models, while propagating uncertainty through time, to plan ahead for only relatively short horizons, which limits the computational burden and allows for infinite-horizon control applications. Our MPC approach does not require a parametrised policy since the control signals themselves are treated as free parameters.

Related Work *Model-based RL:* A recent survey of model based RL in robotics [31] highlights the importance of models for building adaptable robots. Instead of GP dynamics model with a zero prior mean (as used in this paper) a RBF and linear mean functions are proposed [7, 3]. This accelerates learning and facilitates transferring a learned model from simulation to a real robot. Even implicit model learning can be beneficial: The UNREAL learner proposed in [16] learns a predictive model for the environment as an axillary task, which turns out to accelerate learning.

MPC with GP transition models: GP-based predictive control was used for boiler and building control [12, 26], but the model uncertainty was discarded. In [17], the predictive variances were used within a GP-MPC scheme to actively reject periodic disturbances, although not in an RL setting. In [5, 26], the authors considered MPC problems with GP models, where only the GP's posterior mean was used while ignoring the variance for planning. MPC methods with deterministic models are useful only when model errors and system noise can be neglected in the problem [18, 11].

Optimal Control: The application of optimal control theory for the models based on GP dynamics employs some structure in the transition model, i.e., there is an explicit assumption of control affinity [15, 29, 30, 5] and linearisation via locally quadratic approximation [5, 29]. The AICO model [43] uses approximate inference with (known) locally linear models. The probabilistic trajectories, for model-free RL, in [36] are obtained by minimising the KL divergence, while we use moment matching approximation.

Contribution The contributions of this paper are threefold: 1) We propose a new 'deterministic' formulation for probabilistic MPC with learned GP models and uncertainty propagation for long-term planning. 2) This reformulation allows us to apply Pontryagin's Maximum Principle (PMP) for the open-loop planning stage of MPC with GPs. Using the PMP we can handle control constraints in a principled fashion while still maintaining necessary conditions for optimality. 3) The proposed algorithm is not only theoretically justified by optimal control theory, but also achieves a state-of-the-art data efficiency in RL while maintaining the probabilistic formulation.

2 Practical Approach to Data-Efficient Learning of Controllers via MPC

We consider a stochastic dynamical system with states $\boldsymbol{x} \in \mathbb{R}^D$ and admissible controls (actions) $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^U$, where the state follows Markovian dynamics

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{w} \tag{1}$$

with an (unknown) transition function f and i.i.d. system noise $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$, where $\boldsymbol{Q} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_D^2)$. In this paper, we consider an RL setting where we seek control signals

$\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{T-1}^*$ that minimise the expected long-term cost

$$J = \mathbb{E}[\Phi(\boldsymbol{x}_T)] + \sum_{t=0}^{T-1} \mathbb{E}[\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)] \,, \tag{2}$$

where $\Phi(\boldsymbol{x}_T)$ is a terminal cost and $\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)$ the cost associated with applying control $\boldsymbol{u}_t$ in state $\boldsymbol{x}_t$.

We assume that the initial state is Gaussian distributed, i.e., $p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$. We further assume that the terminal state is free.

For reasons of data efficiency, we follow a model-based RL strategy, i.e., we learn a model of the unknown transition function f, which will subsequently be used to find open-loop[2] optimal controls $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{T-1}^*$ that minimise (2). After every application of the control sequence, we update the learned model with the newly acquired experience and re-plan. Section 2.1 summarises the model learning step; Section 2.2 details how to obtain the desired open-loop trajectory.

2.1 Learning a Probabilistic Transition Model

We learn a probabilistic model of the unknown underlying dynamics f to be robust to model errors [38, 9]. In particular, we use a Gaussian process (GP) as a prior $p(f)$ over plausible transition functions f. A GP is a probabilistic non-parametric model for regression. In a GP, any finite number of function values is jointly Gaussian distributed [35]. A GP is fully specified by a mean function $m(\cdot)$ and a covariance function (kernel) $k(\cdot, \cdot)$.

The inputs for the dynamics GP are given by tuples $\widetilde{\boldsymbol{x}}_t := (\boldsymbol{x}_t, \boldsymbol{u}_t)$ and the corresponding targets are $\boldsymbol{x}_{t+1}$. We denote the collections of training inputs and targets by $\widetilde{\boldsymbol{X}}, \boldsymbol{y}$, respectively. Furthermore, we assume a Gaussian (RBF, squared exponential) covariance function

$$k(\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_j) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\widetilde{\boldsymbol{x}}_i - \widetilde{\boldsymbol{x}}_j)^T \boldsymbol{L}^{-1} (\widetilde{\boldsymbol{x}}_i - \widetilde{\boldsymbol{x}}_j)\right) \,, \tag{3}$$

where σ_f^2 is the signal variance and $\boldsymbol{L} = \mathrm{diag}(l_1, \dots, l_{D+U})$ is a diagonal matrix of length-scales $l_1, \dots, l_{D+U}$. The GP is trained via the standard procedure of evidence maximisation [20, 35].

In the context of our setting, we make the standard assumption that the GPs for each target dimension of the transition function $f : \mathbb{R}^D \times \mathcal{U} \to \mathbb{R}^D$ are independent. For a given set of hyper-parameters and a new test input $\widetilde{\boldsymbol{x}}_*$, the GP yields the predictive distribution $p(f(\widetilde{\boldsymbol{x}}_*)|\widetilde{\boldsymbol{X}}, \boldsymbol{y}) = \mathcal{N}(f(\widetilde{\boldsymbol{x}}_*)|m(\widetilde{\boldsymbol{x}}_*), \Sigma(\widetilde{\boldsymbol{x}}_*))$, where

$$m(\widetilde{\boldsymbol{x}}_*) = [m_1(\widetilde{\boldsymbol{x}}_*), \dots, m_D(\widetilde{\boldsymbol{x}}_*)] \,, \quad m_d(\widetilde{\boldsymbol{x}}_*) = k_d(\widetilde{\boldsymbol{x}}_*, \widetilde{\boldsymbol{X}})(\boldsymbol{K}_d + \sigma_d^2 \boldsymbol{I})^{-1} \boldsymbol{y}_d \tag{4}$$

$$\boldsymbol{\Sigma}(\widetilde{\boldsymbol{x}}_*) = \mathrm{diag}\big(\sigma_1^2(\widetilde{\boldsymbol{x}}_*), \dots, \sigma_D^2(\widetilde{\boldsymbol{x}}_*)\big) \,, \quad \sigma_d^2 = \sigma_{f_d}^2 - k_d(\widetilde{\boldsymbol{x}}_*, \widetilde{\boldsymbol{X}})(\boldsymbol{K}_d + \sigma_d^2 \boldsymbol{I})^{-1} k_d(\widetilde{\boldsymbol{X}}, \widetilde{\boldsymbol{x}}_*) \,, \tag{5}$$

for all predictive dimensions $d = 1, \dots, D$.

2.2 Open-Loop Control

To find the desired open-loop control sequence $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{T-1}^*$, we follow a two-step procedure proposed in [9]. (1) Use the learned GP model to predict the long-term evolution $p(\boldsymbol{x}_1), \dots, \boldsymbol{p}(\boldsymbol{x}_T)$ of the state for a given control sequence $\boldsymbol{u}_0, \dots, \boldsymbol{u}_{T-1}$. (2) Compute the corresponding expected long-term cost (2) and find an open-loop control sequence $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{T-1}^*$ that minimises the expected long-term cost. In the following, we will detail these steps.

2.2.1 Long-term Predictions

To obtain the state distributions $p(\boldsymbol{x}_1), \dots, p(\boldsymbol{x}_T)$ for a given control sequence $\boldsymbol{u}_0, \dots, \boldsymbol{u}_{T-1}$, we iteratively predict

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{u}_t) = \iint p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t) p(\boldsymbol{x}_t) p(f) df d\boldsymbol{x}_t \,, \quad t = 0, \dots, T-1 \,, \tag{6}$$

by making a *deterministic* Gaussian approximation to $p(\boldsymbol{x}_{t+1}|\boldsymbol{u}_t)$ using moment matching [11, 33, 9]. This approximation has been shown to work well in practice in RL contexts [9, 8, 7, 2, 3, 29, 30] and can be computed in closed form with the Gaussian kernel (3).

[2] 'Open-loop' refers to the fact that the control signals are independent of the state, i.e., there is no state feedback incorporated.

A key property that we will exploit later is that moment matching allows us to formulate the uncertainty propagation in (6) as a 'deterministic system function'

$$\boldsymbol{z}_{t+1} = f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)\,, \quad \boldsymbol{z}_t := [\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t]\,, \tag{7}$$

where $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$ are the mean and the covariance of $p(\boldsymbol{x}_t)$. For a deterministic control signal $\boldsymbol{u}_t$, we further define the moments of the control-augmented distribution $p(\boldsymbol{x}_t, \boldsymbol{u}_t)$ as

$$\widetilde{\boldsymbol{z}}_t := [\widetilde{\boldsymbol{\mu}}_t, \widetilde{\boldsymbol{\Sigma}}_t]\,, \qquad \widetilde{\boldsymbol{\mu}}_t = [\boldsymbol{\mu}_t^T,\, \boldsymbol{u}_t^T]\,, \quad \widetilde{\boldsymbol{\Sigma}}_t = \text{blkdiag}[\boldsymbol{\Sigma}_t,\, \mathbf{0}]\,, \tag{8}$$

such that (7) can equivalently be written as the deterministic system equation

$$\boldsymbol{z}_{t+1} = f_{MM}(\widetilde{\boldsymbol{z}}_t)\,. \tag{9}$$

2.2.2 Finding the Optimal Open-Loop Control Sequence

To find the optimal open-loop sequence $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{T-1}^*$, we first compute the expected long-term cost J in (2) using the Gaussian approximations $p(\boldsymbol{x}_1), \dots, p(\boldsymbol{x}_T)$ obtained via (6) for a given open-loop control sequence $\boldsymbol{u}_0, \dots, \boldsymbol{u}_{T-1}$. Second, we find a control sequence that minimises the expected long-term cost (2). In the following, we detail these steps.

Computing the Expected Long-Term Cost To compute the expected long-term cost in (2), we sum up the expected immediate costs

$$\mathbb{E}[\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)] = \int \ell(\widetilde{\boldsymbol{x}}_t)\mathcal{N}(\widetilde{\boldsymbol{x}}_t|\widetilde{\boldsymbol{\mu}}_t, \widetilde{\boldsymbol{\Sigma}}_t)d\widetilde{\boldsymbol{z}}_t\,, \quad t = 0, \dots, T-1\,. \tag{10}$$

We choose ℓ, such that this expectation and the partial derivatives $\partial\mathbb{E}[\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)]/\partial\boldsymbol{z}_t$, $\partial\mathbb{E}[\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)]/\partial\boldsymbol{u}_t$ can be computed analytically.[3] Similar to (7), this allows us to define deterministic mappings ℓ_{MM}, Φ_{MM}, such that

$$\mathbb{E}[\ell(\boldsymbol{x}_t, \boldsymbol{u}_t)] =: \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) = \ell_{MM}(\widetilde{\boldsymbol{z}}_t) \tag{11}$$

$$\Phi_{MM}(\boldsymbol{z}_T) := \mathbb{E}[\Phi(\boldsymbol{x}_T)] \tag{12}$$

in (2), respectively, that map the mean and covariance of $\widetilde{\boldsymbol{x}}$ onto the corresponding expected costs.

Remark 1. The open-loop optimisation turns out to be sparse [4]. However, optimisation via the value function or dynamic programming is valid only for unconstrained controls. To address this practical shortcoming, we define Pontryagin's Maximum Principle (PMP) that allows us to formulate the constrained problem while maintaining the sparsity. We detail this sparse structure for the constrained GP dynamics problem in section 3.

2.3 Feedback Control with Model Predictive Control

Thus far, we have presented a way for efficiently determining an open-loop controller. However, an open-loop controller cannot be use to stabilise the system [21]. Therefore, it is essential to obtain a state-feedback controller. Model-predictive control (MPC) is a practical framework for this [21, 14]. While interacting with the system MPC determines an H-step open-loop control trajectory $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{H-1}^*$, starting from the current state $\boldsymbol{x}_t$[4]. Only the first control signal $\boldsymbol{u}_0^*$ is applied to the system. When the system transitions into the successor state $\boldsymbol{x}_{t+1}$, we update the GP model with the newly available information, and MPC re-plans $\boldsymbol{u}_0^*, \dots, \boldsymbol{u}_{H-1}^*$. This procedure turns an open-loop controller into an implicit closed-loop (feedback) controller by repeated re-planning H steps ahead from the current state. Typically, $H \ll T$, and MPC even allows for $T = \infty$.

In this section, we provided an algorithmic framework for nonlinear MPC (NMPC) with learned GP models for the underlying system dynamics, where we explicitly use the GP's uncertainty for long-term predictions (6). In the following section, we will justify this using optimal control theory. Additionally, we will discuss how to account for constrained control signals in a principled way without the necessity to warp/squash control signals [9].

[3] Choices for ℓ include the standard quadratic (polynomial) cost, but also costs expressed as Fourier series expansions or radial basis function networks with Gaussian basis function.

[4] A state distribution $p(\boldsymbol{x}_t)$ would work equivalently in our framework.

3 Theoretical Justification and Results

Bellman's optimality principle[1] yields a recursive formulation for calculating the total expected cost (2) and gives a sufficient optimality condition. Pontryagin's Maximum Principle (PMP) [32] provides the corresponding necessary optimality condition. The PMP allows us to compute gradients $\partial J/\partial \boldsymbol{u}_t$ of the expected long-term cost w.r.t. the variables that only depend on variables with neighbouring time index, i.e., $\partial J/\partial \boldsymbol{u}_t$ depends only variables with index t and $t+1$. Furthermore, it allows us to explicitly deal with constraints on the control signals. In the following, we detail how to solve the optimal control problem OCP with Pontryagin's maximum principle for learned GP dynamics with a deterministic uncertainty propagation. We additionally provide a computationally efficient way to compute derivatives based on the maximum principle.

To facilitate our discussion we first define some notation. Practical control signals are often constrained, we formally define a class of *admissible controls* $\mathcal{U}$ that are piecewise continuous functions defined on a compact space $U \subset \mathbb{R}^U$. This definition is fairly general, and, for example, commonly used zero-order-hold or first-order-hold signals satisfy this requirement. Applying admissible controls to the deterministic system dynamics f_{MM} defined in (9) yield a set $\mathcal{Z}$ of *admissible controlled trajectories*. We define the tuple $(\mathcal{Z}, f_{MM}, \mathcal{U})$ as our control system.

We now define the control-Hamiltonian [6, 37, 41] for this control system as

$$\mathcal{H}(\boldsymbol{\lambda}_{t+1}, \boldsymbol{z}_t, \boldsymbol{u}_t) = \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) + \boldsymbol{\lambda}_{t+1}^T f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t). \tag{13}$$

This formulation of the control-Hamiltonian is the centre piece of the Pontryagin's approach to the OCP. The vector $\boldsymbol{\lambda}_{t+1}$ can be viewed as a Lagrange multiplier for dynamics constraints associated with the OCP [6, 37].

To successfully apply PMP we need the system dynamics to have a unique solution for a given control sequence. Traditionally, this is interpreted as the system is 'deterministic'. This interpretation has been cited as a limitation of Maximum principle [41]. In this paper, however, we exploit the fact that the moment-matching approximation (6) is a deterministic operator, similar to the projection used in EP [28, 22]. This yields the 'deterministic' system equations (7), (9) that map moments of the state distribution at time t to moments of the state distribution at time $t+1$.

To apply the PMP we need to extend some of the important characteristics of ODEs to our system. In particular, we need to show the existence and uniqueness of a (local) solution to our difference equation (9). For existence of a solution we need to satisfy the difference equation point-wise over the entire horizon and for uniqueness we need the system to have only one singularity. For our discrete-time system equation (via the moment-matching approximation) in (7) we have the following

Lemma 1. The moment matching mapping f_{MM} is Lipschitz continuous for controls defined over a compact set $\mathcal{U}$.

The proof is based on bounding the gradient of f_{MM} and detailed in the supplementary material. Existence and uniqueness of the trajectories for the moment matching difference equation are given by the following lemma:

Lemma 2. A solution of $\boldsymbol{z}_{t+1} = f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)$ exists and is unique.

Proof Sketch Difference equations always yield an answer for a given input. Therefore, a solution trivially exists. Uniqueness directly follows from the Picard-Lindelöf theorem, which we can apply due to Lemma 1. This theorem requires the discrete-time system function to be deterministic (see Appendix B of [37]). Due to our re-formulation of the system dynamics (7), this follows directly, such that the $\boldsymbol{z}_{1:T}$ for a given control sequence $\boldsymbol{u}_{0:T-1}$ is unique.

With these results and the definition of the control-Hamiltonian 13 we can now state the PMP for the control system $(\boldsymbol{Z}, f_{MM}, \mathcal{U})$ as follows:

Theorem 1. ***Pontryagin's Maximum Principle for GP Dynamics*** *Let $(\boldsymbol{z}_t^*, \boldsymbol{u}_t^*)$, $0 \leq t \leq H-1$ be an admissible controlled trajectory defined over the horizon H. If $(\boldsymbol{z}_{0:H}^*, \boldsymbol{u}_{0:H-1}^*)$ is optimal, then there exists an ad-joint vector $\boldsymbol{\lambda}_t \in \mathbb{R}^D \setminus \{\mathbf{0}\}$ satisfying the following conditions:*

1. Ad-joint equation: The ad-joint vector $\boldsymbol{\lambda}_t$ is a solution to the discrete difference equation

$$\boldsymbol{\lambda}_t^T = \frac{\partial}{\partial \boldsymbol{z}_t} \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) + \boldsymbol{\lambda}_{t+1}^T \frac{\partial f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{z}_t}\,. \tag{14}$$

2. Transversality condition: At the endpoint ad-joint vector $\boldsymbol{\lambda}_H$ satisfies

$$\boldsymbol{\lambda}_H = \frac{\partial}{\partial \boldsymbol{z}_H} \Phi_{MM}(\boldsymbol{z}_H)\,. \tag{15}$$

3. Minimum Condition: For $t = 0, \ldots, H-1$, we have

$$\mathcal{H}(\boldsymbol{\lambda}_{t+1}, \boldsymbol{z}_t^*, \boldsymbol{u}_t^*) = \min_{\boldsymbol{\nu}} \mathcal{H}(\boldsymbol{\lambda}_{t+1}, \boldsymbol{z}_t^*, \boldsymbol{\nu})\,, \quad \boldsymbol{\nu} \in \mathcal{U}\,. \tag{16}$$

Remark 2. The minimum condition (16) can be used to find an optimal control. The Hamiltonian is minimised point-wise over the admissible control set $\mathcal{U}$: For every $t = 0, \ldots, H-1$we find optimal control $\boldsymbol{u}_t^* \in \arg\min_{\boldsymbol{\nu}} \mathcal{H}(\boldsymbol{\lambda}_{t+1}, \boldsymbol{z}_t^*, \boldsymbol{\nu})$. The minimisation problem possesses additional variables $\boldsymbol{\lambda}_{t+1}$. These variables can be interpreted as Lagrange multipliers for the optimisation. They capture impact of control $\boldsymbol{u}_t$ over the whole trajectory and, hence, these variables make the optimisation problem sparse [10]. For the GP dynamics we compute the multipliers $\boldsymbol{\lambda}_t$ in closed form, thereby, significantly reducing the computational burden to minimise the cost J in equation (2)We detail this calculation in section 3.1.

Remark 3. The Hamiltonian $\mathcal{H}$ in (16) is constant for unconstrained control in time-invariant dynamics and equals 0 everywhere when final time H is not fixed [37].

Remark 4. For linear dynamics the proposed method is a generalisation of iLQG [42]: The moment matching transition f_{MM} implicitly linearises the transition dynamics at each time step, whereas in iLQG an explicit local linear approximation is made. For a linear f_{MM}, and quadratic cost we can write the LQG case as shown in Theorem 1 in [43]. If we iterate with successive corrections to the linear approximations we obtain iLQG. Recently, iLQG based on these principles was proposed in [19].

3.1 Efficient Gradient Computation

With the definition of the Hamiltonian $\mathcal{H}$ in (13) we can efficiently calculate the gradient of the expected total cost J efficiently. For a time horizon H we can write the accumulated cost as the Bellman recursion [1]

$$J_H(\boldsymbol{z}_H) := \Phi_{MM}(\boldsymbol{z}_H), \quad J_t(\boldsymbol{z}_t) := \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t) + J_{t+1}(f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)) \tag{17}$$

for $t = H-1, \ldots, 0$. Since the (open-loop) control $\boldsymbol{u}_t$ only impacts the future costs via $\boldsymbol{z}_{t+1} = f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)$ the derivative of the total cost with $\boldsymbol{u}_t$ is given by

$$\frac{\partial J_t}{\partial \boldsymbol{u}_t} = \frac{\partial \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{u}_t} + \frac{\partial J_{t+1}}{\partial \boldsymbol{z}_{t+1}} \frac{\partial f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{u}_t}\,. \tag{18}$$

Comparing this expression with the definition of the Hamiltonian (13), we see that if we make the substitution $\boldsymbol{\lambda}_{t+1}^T = \frac{\partial J_{t+1}}{\partial \boldsymbol{z}_{t+1}}$ we obtain

$$\frac{\partial J_t}{\partial \boldsymbol{u}_t} = \frac{\partial \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{u}_t} + \boldsymbol{\lambda}_{t+1}^T \frac{\partial f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{u}_t} = \frac{\mathcal{H}}{\partial \boldsymbol{u}_t}. \tag{19}$$

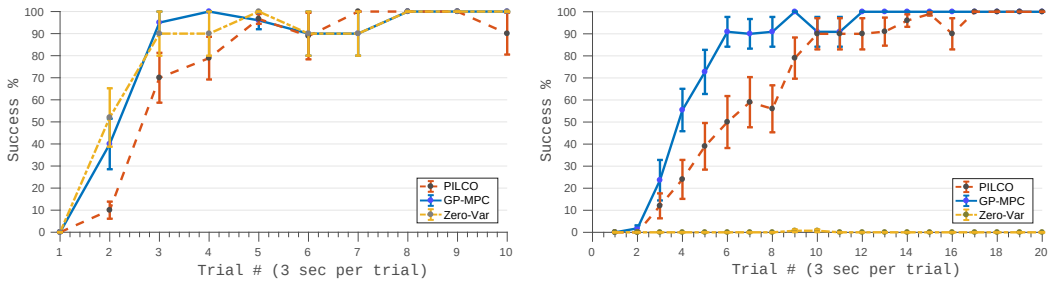
This implies that the gradient of the expected long-term cost w.r.t. $\boldsymbol{u}_t$ can be efficiently computed using the Hamiltonian [41]. Next we show that the substitution $\boldsymbol{\lambda}_{t+1}^T = \frac{\partial J_{t+1}}{\partial \boldsymbol{z}_{t+1}}$ is valid for the entire horizon H. For the terminal cost $\Phi_{MM}(\boldsymbol{z}_H)$ this is valid by the transversality condition (15). For other time steps we differentiate (17) w.r.t. $\boldsymbol{z}_t$

$$\frac{\partial J_t}{\partial \boldsymbol{z}_t} = \frac{\partial \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{z}_t} + \frac{\partial J_{t+1}}{\partial \boldsymbol{z}_{t+1}} \frac{\partial \boldsymbol{z}_{t+1}}{\partial \boldsymbol{z}_t} = \frac{\partial \ell_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{z}_t} + \boldsymbol{\lambda}_{t+1}^T \frac{\partial f_{MM}(\boldsymbol{z}_t, \boldsymbol{u}_t)}{\partial \boldsymbol{z}_t}\,, \tag{20}$$

which is identical to the ad-joint equation (14). Hence, in our setting the PMP implies that gradient descent on the Hamiltonian $\mathcal{H}$ is equivalent to gradient descent on the total cost (2) [37, 10].

Algorithmically, in an RL setting, we find the optimal control sequence $\boldsymbol{u}_0^*, \ldots, \boldsymbol{u}_{H-1}^*$ as follows:

1. For a given initial (random) control sequence $\boldsymbol{u}_{0:H-1}$ we follow the steps described in section 2.2.1 to determine the corresponding trajectory $\boldsymbol{z}_{1:H}$. Additionally, we compute Lagrange multipliers $\boldsymbol{\lambda}_{t+1}^T = \frac{\partial J_{t+1}}{\partial \boldsymbol{z}_{t+1}}$ during the forward propagation. Note that traditionally ad-joint equations are propagated backward to find the multipliers [6, 37].



(a) Under-actuated cart-pole swing-up.

(b) Fully-actuated double-pendulum swing-up.

Figure 1: Performance of RL algorithms. Error bars represent the standard error. (a) Cart-pole; (b) Double pendulum. GP-MPC (blue) consistently outperforms PILCO (red) and the zero-variance MPC approach (yellow) in terms of data efficiency. While the zero-variance MPC approach works well on the cart-pole task, it fails in the double-pendulum task. We attribute this to the inability to explore the state space sufficiently well.

2. Given $\boldsymbol{\lambda}_t$ and a cost function ℓ_{MM} we can determine the Hamiltonians $\mathcal{H}_{1:H}$. Then we find a new control sequence $\boldsymbol{u}^*_{0:H-1}$ via any gradient descent method using (19).
3. Return to 1. or exit when converged.

We use Sequential Quadratic Programming (SQP) with BFGS for Hessian updates [27]. The Lagrangian of SQP is a partially separable function [13]. In PMP this separation is explicit via the Hamiltonians, i.e., we $\mathcal{H}_t$ is a function of variables with index t or $t+1$. This leads to a block-diagonal Hessian of SQP Lagrangian [13]. The structure can be exploited to approximate Hessian via block-updates within a BFGS method [13, 4]

4 Experimental Results

We evaluate the learning performance of our proposed MPC-based RL algorithm on two different RL problems with non-linear dynamics and control constraints. The first RL problem is the under-actuated cart-pole swing-up benchmark. The second RL problem is the double-pendulum swing-up, where both links can be actuated [8]. In both experiments, we use the exact saturating cost $\ell = 1 - \exp(-d^2/\kappa^2)$ used in [8], where d/κ is the Euclidean distance of the tip of the (outer) pendulum from the target position. This cost function provides a scarce reward with a peak only near the desired location, which makes MPC challenging.

We compare our GP-MPC approach with the PILCO algorithm [9, 8], which defines the current state of the art in terms of data efficiency when it comes to solving these problems. Furthermore, we assess the suitability of a zero-variance GP-MPC algorithm (in the flavour of [26, 5]) for RL, where the GP's predictive variances are discarded.

All RL algorithms start off with a single random trajectory, which is used for learning the dynamics model. As in [9, 8] the GP is used to predict state differences $\boldsymbol{x}_{t+1} - \boldsymbol{x}_t$. The learned GP dynamics model is then used to determine a controller (section 2.2.1), which is then applied to the system, starting from $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$. Model learning, controller learning and application constitute a 'trial'. After each trial, the model is updated with the newly acquired experience and learning continues. We average over 20 independent experiments, where every algorithm is initialised with the same first (random) trajectory. The performance differences of the RL algorithms are therefore due to different approaches to controller learning and the induced exploration.

Under-actuated Cart-Pole Swing-Up The cart pole system is an under-actuated system with a freely swinging pendulum of $50\,\mathrm{cm}$ mounted on a cart. The swing-up and balancing task has non-linearities that a linear model may not be sufficient to solve [34]. The cart-pole system state space consists of the position of the cart x, cart velocity $\dot{x}$, the angle θ of the pendulum and the angular velocity $\dot{\theta}$. A horizontal constrained force $u \in [-10, 10]\,\mathrm{N}$ can be applied to the cart. Starting in a

position where the pendulum hangs downwards, the objective is to automatically learn a controller that swings the pendulum up and balances it in the inverted position in the middle of the track.

Fig. 1(a) shows that our MPC-based controller (blue) successfully[5] completes the task a few trials before the state-of-the-art PILCO method (red). The zero-variance approach (yellow) performs well, somewhere between the other two models. From the repeated trials we see that GP-MPC learns faster and more reliably, i.e., it is more robust to variations in the start state, than PILCO. In particular, GP-MPC can solve the cart-pole task with high probability after 3 trials (9 seconds), where the first trial was random. PILCO and the zero-variance approach need two additional trials.

Fully actuated Double-Pendulum The double pendulum system is a two-link robot arm (links lengths: $1\,\mathrm{m}$) with two actuators at each joint. The state space consists of 2 angles and 2 angular velocities $[\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]$ [8]. The torques u_1 and u_2 are limited to $[-2, 2]$ Nm. Starting from a position where both links are in a downward position, the objective is to learn a control strategy that swings the double-pendulum up and balances it in the inverted position.

Fig. 1(b) highlights that our proposed GP-MPC approach (yellow) requires on average only six trials ($18\,\mathrm{s}$) of experience to achieve a 90% success rate[6], including the first random trial. PILCO requires four additional trials, whereas the zero-variance MPC approach completely fails in this RL setting. The reason for this is that the deterministic predictions with a poor model in this complicated state space do not allow for sufficient exploration. We also observe that GP-MPC is more robust to the variations amongst trials.

5 Discussion and Conclusion

Key to the success and learning speed of our method are the ability to (1) immediately react to observed states by adjusting the long-term plan and (2) update the GP model on the fly as soon as a new state transition is observed. Both properties turn out to be crucial in the very early stages of learning when nearly no information is available. If we ignored the on-the-fly updates of the GP dynamics model, our approach would still successfully learn, although the learning efficiency would be slightly decreased (in our experience by about two trials).

The MPC based model proposed here has fewer parameters than methods relying on parametrised policies. In our approach, the control signals are directly optimised, which typically reduces the dimensionality of the problem significantly. For example, the PILCO algorithm [9] needs to optimise hundreds/thousands of policy parameters per control dimension, whereas the number of parameters we need to optimise corresponds to the length of the MPC horizon times the dimensionality of the control signal. Hence, unlike PILCO we can update all control signals in the sequence simultaneously leading to significant computational advantage over PILCO while using the same long-term prediction model.

We learn the dynamics of the system via difference-learning allowing us to extend the optimal control theory to GP dynamics. The difference equation can be used to obtain extremal trajectories, i.e., we can guarantee that a first-order optimality criteria is met.

Our approach exhibits similarities to iLQR/iLQG [42], which explicitly linearises the dynamics model locally. In our framework, we do the same implicitly via moment matching. The local linearity assumption may be valid for short horizons only. For long control horizons moment-matching is significantly better [28, 23]. However, if the true model is linear, the moment matching and linearisation are identical, and our approach would be equivalent to LQG [43].

In this paper, we focussed on moment matching as a way to re-formulate the probabilistic system dynamics. However, any deterministic approximate inference method (e.g., linearisation, unscented transformation) can be used instead.

We have proposed an algorithm for data-efficient RL that is based on probabilistic MPC with learned transition models using Gaussian processes. By exploiting Pontryagin's maximum principle our algorithm can naturally deal with control constraints. Key to this theoretical underpinning of a practical algorithm was the re-formulation of the optimal control problem with uncertainty

[5] We define 'success' if the pendulum is closer than $8\,\mathrm{cm}$ to the target position.

[6] The tip of outer pendulum is closer than $22\,\mathrm{cm}$ to the target.

propagation via moment matching into an deterministic optimal control problem. We provided empirical evidence that our framework is not only theoretically sound, but also extremely data efficient, even compared to the state of the art.

References

[1] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1957.

[2] B. Bischoff, D. Nguyen-Tuong, T. Koller, H. Markert, and A. Knoll. Learning Throttle Valve Control Using Policy Search. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, 2013.

[3] B. Bischoff, D. Nguyen-Tuong, H. van Hoof, A. McHutchon, C. E. Rasmussen, A. Knoll, J. Peters, and M. Deisenroth. Policy Search for Learning Robot Control using Sparse Data. In *2014 IEEE International Conference on Robotics and Automation*, pages 3882–3887. IEEE, 5 2014.

[4] H. G. Bock and K. J. Plitt. A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems. In *Proceedings 9th IFAC World Congress Budapest*, pages 243–247. Pergamon Press, 1984.

[5] J. Boedecker, J. T. Springenberg, J. Wulfing, and M. Riedmiller. Approximate real-time optimal control based on sparse Gaussian process models. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pages 1–8. IEEE, 12 2014.

[6] F. H. Clarke. *Optimization and Non-Smooth Analysis*. Society for Industrial and Applied Mathematics SIAM, 1990.

[7] M. Cutler and J. P. How. Efficient reinforcement learning for robots using informative simulated priors. In *2015 IEEE International Conference on Robotics and Automation*, pages 2605–2612. IEEE, 5 2015.

[8] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–23, 2 2015.

[9] M. P. Deisenroth and C. E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *Proceedings of the International Conference on Machine Learning*, pages 465–472, New York, NY, USA, 6 2011. ACM.

[10] M. Diehl. *Lecture Notes on Optimal Control and Estimation*. 2014.

[11] A. Girard, C. E. Rasmussen, J. Q. Candela, and R. Murray-Smith. Gaussian Process Priors with Uncertain Inputs-Application to Multiple-Step Ahead Time Series Forecasting. *Advances in Neural Information Processing Systems*, page 545–552, 2003.

[12] A. Grancharova, J. Kocijan, and T. A. Johansen. Explicit Stochastic Predictive Control of Combustion Plants based on Gaussian Process Models. *Automatica*, 44(6):1621–1631, 6 2008.

[13] A. Griewank and P. L. Toint. Partitioned Variable Metric Updates for Large Structured Optimization Problems. *Numerische Mathematik*, 39(1):119–137, 1982.

[14] L. Grüne and J. Pannek. Stability and Suboptimality Using Stabilizing Constraints. In *Nonlinear Model Predictive Control Theory and Algorithms*, pages 87–112. Springer, 2011.

[15] P. Hennig. Optimal Reinforcement Learning for Gaussian Systems. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 325–333. Curran Associates, Inc., 2011.

[16] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement Learning with Unsupervised Auxiliary Tasks. In *5th International Conference on Learning Representations*, 11 2016.

[17] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig. Gaussian Process-Based Predictive Control for Periodic Error Correction. *IEEE Transactions on Control Systems Technology*, 24(1):110–121, 2016.

[18] J. Kocijan. *Modelling and Control of Dynamic Systems Using Gaussian Process Models*. Advances in Industrial Control. Springer International Publishing, Cham, 2016.

[19] G. Lee, S. S. Srinivasa, and M. T. Mason. GP-ILQG: Data-driven Robust Optimal Control for Uncertain Nonlinear Dynamical Systems. 5 2017.

[20] D. J. C. MacKay. Introduction to Gaussian Processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168, pages 133–165. Springer, Berlin, Germany, 1998.

[21] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[22] T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1 2001.

[23] T. P. Minka. Expectation Propagation for Approximate Bayesian Inference. In J. S. Breese and D. Koller, editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 362–369, Seattle, WA, USA, 8 2001. Morgan Kaufman Publishers.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2 2015.

[25] D. S. D. S. Naidu and R. C. Naidu, Subbaram/Dorf. *Optimal Control Systems*. CRC Press, 2003.

[26] T. X. Nghiem and C. N. Jones. Data-driven Demand Response Modeling and Control of Buildings with Gaussian Processes. In *Proceedings of the American Control Conference*, 2017.

[27] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

[28] M. Opper and O. Winther. Tractable approximations for probabilistic models: The adaptive TAP mean field approach. *Physical Review Letters*, 86(17):5, 2001.

[29] Y. Pan and E. Theodorou. Probabilistic Differential Dynamic Programming. *Advances in Neural Information Processing Systems*, 2014.

[30] Y. Pan, E. Theodorou, and M. Kontitsis. Sample Efficient Path Integral Control under Uncertainty, 2015.

[31] A. S. Polydoros and L. Nalpantidis. Survey of Model-Based Reinforcement Learning: Applications on Robotics. *Journal of Intelligent and Robotic Systems*, 86(2):153–173, 1 2017.

[32] L. S. Pontryagin, E. F. Mishchenko, V. G. Boltyanskii, and R. V. Gamkrelidze. *The Mathematical Theory of Optimal Processes*. Wiley, 1962.

[33] J. Quiñonero-Candela, C. C. E. Rasmussen, J. Quinonero-Candela, and C. C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *The Journal of Machine Learning Research*, 6(2):1939–1960, 2005.

[34] T. Raiko and M. Tornio. Variational Bayesian Learning of Nonlinear Hidden State-Space Models for Model Predictive Control. *Neurocomputing*, 72(16–18):3702–3712, 2009.

[35] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 2006.

[36] K. Rawlik, M. Toussaint, and S. Vijayakumar. On Stochastic Optimal Control and Reinforcement Learning by Approximate Inference. In *Robotics: Science and Systems*, 2012.

[37] H. Schättler and U. Ledzewicz. *Geometric Optimal Control: Theory, Methods and Examples*, volume 53. 2012.

[38] J. G. Schneider. Exploiting Model Uncertainty Estimates for Safe Dynamic Control Learning. In *Advances in Neural Information Processing Systems*. Morgan Kaufman Publishers, 1997.

[39] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 529(7587), 2016.

[40] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, USA, 1998.

[41] E. Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences of the United States of America*, 106(28):11478–83, 2009.

[42] E. Todorov and Weiwei Li. A Generalized Iterative LQG Method for Locally-Optimal Feedback Control of Constrained Nonlinear Stochastic Systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 300–306. IEEE.

[43] M. Toussaint. Robot Trajectory Optimization using Approximate Inference. In *Proceedings of the 26th International Conference on Machine Learning,*, Montreal, QC, Canada, 6 2009.

[44] A. Yahya, A. Li, M. Kalakrishnan, Y. Chebotar, and S. Levine. Collective Robot Reinforcement Learning with Distributed Asynchronous Guided Policy Search. *arXiv preprint arXiv:1610.00673*, 2016.

6 Appendix

Lemma 3. The moment matching mapping f_{MM} is Lipschitz continuous for controls defined over a compact set $\mathcal{U}$.

Proof: Lipschitz continuity requires that the gradient $\partial f_{MM}/\partial \boldsymbol{u}_t$ is bounded. The gradient is

$$\frac{\partial f_{MM}}{\partial \boldsymbol{u}_t} = \frac{\partial \boldsymbol{z}_{t+1}}{\partial \boldsymbol{u}_t} = \left[\frac{\partial \boldsymbol{\mu}_{t+1}}{\partial \boldsymbol{u}_t}, \frac{\partial \boldsymbol{\Sigma}_{t+1}}{\partial \boldsymbol{u}_t}\right]. \tag{21}$$

The derivatives $\left[\frac{\partial \boldsymbol{\mu}_{t+1}}{\partial \boldsymbol{u}_t}, \frac{\partial \boldsymbol{\Sigma}_{t+1}}{\partial \boldsymbol{u}_t}\right]$ can be computed analytically [8].

We first show that the derivative $\partial \boldsymbol{\mu}_{t+1}/\partial \boldsymbol{u}_t$ is bounded. Defining $\boldsymbol{\beta}_d := (\boldsymbol{K}_d + \sigma_{f_d}^2 \boldsymbol{I})^{-1} \boldsymbol{y}_d$, from [8], we obtain for all state dimensions $d = 1, \ldots, D$

$$\mu_{t+1}^d = \sum\nolimits_{i=1}^{N} \beta_{d_i} q_{d_i}\,, \quad q_{d_i} = \sigma_{f_d}^2 |\boldsymbol{I} + \boldsymbol{L}_d^{-1} \widetilde{\boldsymbol{\Sigma}}_t|^{-\frac{1}{2}} \exp\big(-\tfrac{1}{2}(\widetilde{\boldsymbol{x}}_i - \widetilde{\boldsymbol{\mu}}_t)^T (\boldsymbol{L}_d + \widetilde{\boldsymbol{\Sigma}}_t)^{-1} (\widetilde{\boldsymbol{x}}_i - \widetilde{\boldsymbol{\mu}}_t)\big)\,, \tag{22}$$

where N is the size of the training set of the dynamics GP and $\widetilde{\boldsymbol{x}}_i$ the ith training input. The corresponding gradient w.r.t. $\boldsymbol{u}_t$ is given by the last F elements of

$$\frac{\partial \mu_{t+1}^d}{\partial \widetilde{\boldsymbol{\mu}}_t} = \sum\nolimits_{i=1}^{N} \beta_{d_i} \frac{\partial q_{d_i}}{\partial \widetilde{\boldsymbol{\mu}}_t} = \sum\nolimits_{i=1}^{N} \beta_{d_i} q_{d_i} (\widetilde{\boldsymbol{x}}_i - \widetilde{\boldsymbol{\mu}}_t)^T (\widetilde{\boldsymbol{\Sigma}}_t + \boldsymbol{L}_d)^{-1} \in \mathbb{R}^{1 \times (D+F)} \tag{23}$$

Let us examine the individual terms in the sum on the rhs in (23): For a given trained GP $\|\boldsymbol{\beta}_d\| < \infty$ is constant. The definition of q_{d_i} in (22) contains an exponentiated negative quadratic term, which is bounded between $[0, 1]$. Since $\boldsymbol{I} +^{-1} L_d \widetilde{\boldsymbol{\Sigma}}_t$ is positive definite, the inverse determinant is defined and bounded. Finally, $\sigma_{f_d}^2 < \infty$, which makes $q_{d_i} < \infty$. The remaining term in (23) is a vector-matrix product. The matrix is regular and its inverse exists and is bounded (and constant as a function of $\boldsymbol{u}_t$. Since $\boldsymbol{u}_t \in \mathcal{U}$ where $\mathcal{U}$ is compact, we can also conclude that the vector difference in (23) is finite, which overall proves that f_{MM} is (locally) Lipschitz continuous and Lemma 3.