2018 IEEE Conference on Decision and Control (CDC)
Miami Beach, FL, USA, Dec. 17-19, 2018

Bayesian nonparametrics and feedback-linearisation of discretised control-affine systems

Jan-Peter Calliess[1], Antonis Papachristodoulou[1] and Stephen J. Roberts[1]

***Abstract*— We propose *random field system identification and inversion control (RF-SIIC)* as a method for simultaneous probabilistic identification and control of time-discretised control-affine systems. Identification is achieved by conditioning random field priors on observations of configurations and noisy estimates of configuration derivatives. In contrast to previous work that has utilised random fields for identification, we leverage the structural knowledge afforded by Lagrangian mechanics and learn both the drift and control input matrix functions of a control-affine system. We employ feedback-linearisation to reduce, in expectation, the uncertain nonlinear control problem to one that is easy to regulate. Our method combines the flexibility of nonparametric Bayesian learning with epistemological guarantees on the expected closed-loop trajectory. We illustrate the viability of our approach in the context of a discretised, fully-actuated mechanical system. Our simulations suggest that our approach can adapt rapidly to a priori uncertain dynamics sufficiently well to succeed in feedback-linearising and controlling the plant as desired.**

I. Introduction

Control may be regarded as decision making in a dynamic environment. Decisions have to be based on beliefs over the consequences of actions encoded by a model. Dealing with uncertain or changing dynamics is the realm of adaptive control. In classical adaptive control, parametric approaches are used (e.g. [26]) and uncertainties are typically modelled by Brownian motion (yielding stochastic adaptive control [13], [8]) or via set-based considerations (an approach followed by robust adaptive control [19]). In contrast, we adopt an epistemological take on probabilistic control and bring to bear Bayesian nonparametric learning methods, whose introspective qualities [9] can address exploration-exploitation trade-offs in a principled manner [1].

In contrast to classical adaptive control where inference has to be restricted to finite-dimensional parameter space, the nonparametric approach affords the learning algorithms with greater flexibility to identify and control systems with very few model assumptions. This is possible because these methods grant the flexibility to perform Bayesian inference over rich, infinite-dimensional function spaces that could encode the dynamics. This property has led to a surge of interest in Bayesian nonparametrics; particularly benefiting their algorithmic advancement and application to a plethora of learning problems. Due to their favourable analytic properties, *Gaussian processes* (GPs) [2], [20] have been the main choice of method in recent years. Among other domains, GPs have been applied to learning discrete-time dynamic systems in the context of model-predictive control [11], [12], [14], [21], learning the error of inverse models [15], [17], dual control [1] as well as reinforcement learning and dynamic programming [6], [7], [22], [10]. For articles surveying the field of learning for model learning and control, the reader is referred to [16].

The extent of flexibility inherent in these models can, however, lead to black-box use, disregarding important structural knowledge in the underlying dynamics [11], [12], [22], [10], [14]. This can result in unnecessarily high-dimensional learning problems, slow convergence rates and often necessitates large training corpora, often collected offline. In the extreme, the latter requirement can cause slow prediction and conditioning times. Moreover, black-box GP models have been utilised in combination with computationally intensive planning methods such as dynamic programming [6], [7], [22] rendering many online learning-based control tasks, such as tracking, difficult.

In contrast to this body of work, we incorporate structural *a priori* knowledge of the dynamics, afforded by Lagrangian mechanics (without sacrificing the flexibility of nonparametrics). This requires, in some instances, a (partial) departure from Gaussianity but improves the detail with which the system is identified and can reduce the dimensionality of the identification problem. Our method uses uncertainties inherent in the models to achieve active data selection and decision making. Our approach also employs feedback-linearisation [23] in an outer-loop control law to reduce the complexity of the control problem. The nominal problem is thus reduced to controlling a (discrete-time) double-integrator via an inner-loop control law which can be set to any desired reference acceleration. If we combine the outer-loop controller with a pseudo-controller that has desirable guarantees (e.g. stability) for the double-integrator, these guarantees can extend to the expected closed-loop dynamics inferred by the trained posterior dynamics. The resulting approach enables rapid decision making and can be deployed in online learning and control.

Our approach can be seen as a generalisation of GP-MRAC [4]. In that paper, the authors utilise a Gaussian process on joint state-control space to learn the error of an inversion controller in model-reference adaptive control. The paper contains a convergence theorem based on several restrictive assumptions, including the assumption that the solution trajectory of the GP-driven model could be stated as a Markov process with orthogonal increments represented as an Ito-SDE in time. In addition, their method requires knowledge of the invertibility of the adaptive element. For control-affine systems this means that the control input vector

[1] Department of Engineering Science, University of Oxford, UK.

field has to be known and the uncertainty is only in the drift. In contrast, we consider discrete-time systems where both the drift and control input matrix can be unknown *a priori*. Our method is capable of identifying the drift and control input vector fields constituting the underlying control-affine system individually, yielding a more fine-grained identification result. Moreover, our method is not limited to Gaussian processes. If the control-input vector fields are identified with log-normal processes, our controller will automatically be cautious in scarcely explored regions.

The remainder of this paper is structured as follows: In Sec. II, we first describe the dynamic systems we consider. We then move on to describe our system-identification and inversion-control approach employing Bayesian nonparametrics. Here, we assume that only the drift vector field of the dynamics is uncertain *a priori* while the control input acts upon the state in some known manner. In this setting, we also prove closed-loop stability of the expected trajectory, where the expectation is computed relative to the Bayesian posterior beliefs.

In Sec. III we lift the assumption of knowing the control input nonlinearity and propose an approach where all constituent parts of a control affine dynamical system can be identified based on repeated interactions with the system.

Sec. IV contains simulations that illustrate our approach in the context of an undamped, torque-actuated double pendulum. The simulations suggest that our method can rapidly simultaneously learn and control such a system online.

Sec. V discusses limitations of the present approach as well as open avenues for future investigations.

II. Discrete-time learning and inversion control with an uncertain drift field

In this section, we consider time-discretised versions of second-order systems. We limit our attention to the case where the control input b is known and can be inverted. With notation as before, assume $n = m, d = m + n$.

Let $q \in \mathbb{R}^m$ be a configuration and let $x := [q, \dot{q}] \in \mathcal{X} = \mathbb{R}^d, d = 2m$ be the full state as well as $u \in \mathcal{U}$ denote the control decision that has to reside in control space $\mathcal{U} \subseteq \mathbb{R}^m$. To keep the exposition concrete, we focus on discretisations of second-order systems. So, consider the continuous dynamics

$$\ddot{q} = a(x) + b(x, u) \tag{1}$$

where $b : \mathbb{R}^d \times \mathcal{U} \to \mathbb{R}^m$ is a known, deterministic function that is control invertible, i.e. where we know there exists a function $b^- : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^m$ such that $b\big(x, b^-(x, u')\big) = u', \forall x \in \mathcal{X}, u' \in \mathbb{R}^m$. Examples are fully-actuated, control-affine systems with invertible control input matrices as considered above. That is, where $b(x, u) = B(x)u$ and $B(x) \in \mathbb{R}^{m\times m}$ is invertible. Let $I_m \in \mathbb{R}^{m\times m}$ be the identity matrix, $O_m = 0I_m$ a matrix of zeros and $o_m \in \mathbb{R}^m$ denote the m-dimensional vector of zeros.

We can rewrite our dynamics as the first-order system:

$$\dot{x} = \begin{pmatrix} O_m & I_m \\ O_m & O_m \end{pmatrix} x + f(x) + g(x, u) \tag{2}$$

where $f(x) = \big(o_m, a(x)\big)^\top$ and $g(x) = \big(o_m, b(x, u)\big)^\top$.

An Euler-approximated, time-discrete version may be written as:

$$x_{k+1} = E\, x_k + \Delta\, f(x_k) + \Delta\, g(x_k, u_k) \tag{3}$$

where k denotes the time-index, $\Delta \in \mathbb{R}_+$ is a time-increment and $E = \begin{pmatrix} I_m & \Delta\, I_m \\ O_m & I_m \end{pmatrix}$. This is the system we desire to stabilise. That is we will design a control law $u : \mathcal{X} \to \mathcal{U}$ that drives the state towards goal state 0. This is without loss of generality, since the case of tracking a reference is a trivial extension of stabilisation at zero and therefore, the task of stabilising 0 is the canonical case most often considered in control (cf. e.g. [3]).

Since we will assume f to be uncertain, we can only do so relative to our beliefs over f. Encoding our epistemic beliefs in terms of probabilities, any convergence guarantee on reaching the goal state will therefore have to be of a probabilistic nature. Before deriving a controller with such guarantees, we will next outline how to update one's prior beliefs in the light of data.

A. Bayesian nonparametric drift learning

We assume we know the dynamics are given by Eq. 3, but are uncertain regarding drift vector field $f = (o_m, a)^\top$. The first step in Bayesian nonparametric learning is to model the uncertainty by assuming the drift is drawn from a prior process, $f \sim \Pi^f = (o_m, \Pi^a)^\top$. The notation $\Pi^f = (o_m, \Pi^a)$ indicates that:

$\forall x \in \mathcal{X}$, measurable sets $S = S_1 \times \ldots \times S_{2m} \subset \mathbb{R}^{2m}$: $\Pr_{\Pi^f}[f(x) \in S] = 0$ if $\exists i \leq m : S_i \neq \{0\}$, while also $\Pr_{\Pi^f}[f(x) \in S] = Pr_{\Pi^a}[f_{m+1}(x) \times \ldots \times f_{2m}(x) \in S_{m+1} \times \ldots \times S_{2m}]$, if $\forall i \leq m : S_i = \{0\}$.

Observing a sequence of (state, control, successor-state) triples (x_i, u_i, x_{i+1}) allows one to compute a sequence $\mathcal{D} = \{(x_i, f_i)\}$ where

$$f_i := f(x_i) = \frac{1}{\Delta}(x_{i+1} - E\, x_i) + g(x_i, u_i). \tag{4}$$

Bayesian *learning* consists of *computing the posterior belief process*, $\Pi^f|\mathcal{D} = (o_m, \Pi^a|\mathcal{D})^\top$, on the basis of a collected set $\mathcal{D}$ of training examples extracted from state observations.

B. Inversion control law

Assume we are at time step $k \in \mathbb{N}$, having collected data $\mathcal{D}_k$. Based on our posterior belief, $\Pi^f|\mathcal{D}_k = (o_m, \Pi^a|\mathcal{D}_k)^\top$, we define an inversion-control law as follows:

$$u(k, x_k; u'_k) = b^-(x_k, -\mathfrak{m}_k + u'_k), \tag{5}$$

where

$$\mathfrak{m}_k = \langle a(x_k)|\mathcal{D}_k, x_k\rangle \tag{6}$$

is the expected value of the drift computed with respect to the posterior $\Pi^a|\mathcal{D}_k$, and, u'_k is referred to as the *pseudo-control*. Note, for Gaussian processes, the posterior expectation $\mathfrak{m}_k$ can be computed in closed-form (see e.g. [20]).

The closed-loop dynamics degenerate to

$$x_{k+1} = E\, x_k + \Delta\, F_k + \Delta\, (O_m, I_m)^\top u'_k \tag{7}$$

where $(F_k)_{k\in\mathbb{N}}$, with $F_k = f(x_k) - (o_m, \mathfrak{m}_k)^\top$ is a random field.

Linear-feedback pseudo-control. Let $K \in \mathbb{R}^{m\times d}$ be a feedback gain matrix with positive definite sub-matrices $K_1, K_2 \in \mathbb{R}^{m\times m}$, $K = [K_1, K_2]$. We assume the objective is to drive the state to goal state $\xi = 0$. If we set the *pseudo control* to a linear feedback, $u'_k := -Kx_k$, the control law becomes

$$u(k, x; K) = b^-(x, -\mathfrak{m}_k - Kx) \tag{8}$$

yielding the closed-loop dynamics

$$x_{k+1} = Mx_k + \Delta\, F_k \tag{9}$$

where

$$M = \begin{pmatrix} I_m & \Delta\, I_m \\ -\Delta K_1 & I_m - \Delta K_2 \end{pmatrix}. \tag{10}$$

Here, we normally devise K such that $\|\!|M|\!\|_2 \geq 1$ but $\rho(M) < 1$, in which case M is a *stable* or *Hurwitz* matrix.

C. Convergence guarantee for the expected trajectory

With this setup we can guarantee convergence of the expected state trajectory to the goal state $\xi = 0$.

Theorem. *Assume the closed-loop dynamics are given by Eq. 9 with known matrix M and time increment Δ. Moreover, assume the learner is capable of keeping the training data $\mathcal{D}_k$ up to date, containing the entire history of states up to time step k. That is, $\mathcal{D}_k = \{x_0, ..., x_k\}$. Then, $\langle x_k\rangle = M^k\langle x_0\rangle$. Stability of M ($\rho(M) < 1$) implies convergence of the expected trajectory to the goal $\big(\langle x_k\rangle\big)_{k\in\mathbb{N}_0}$. That is, $\lim_{k\to\infty} \|\langle x_k\rangle\| = 0$.*

Proof: Let $k \in \mathbb{N}$ be an arbitrary time step. Since $x_{k+1} = Mx_k + \Delta\, F_k$ we have $\langle x_{k+1}\rangle = M\langle x_k\rangle + \Delta\langle F_k\rangle$.

Showing $\langle F_k\rangle = 0$ would imply $\langle x_{k+1}\rangle = M\langle x_k\rangle$ and thus, $\langle x_{k+1}\rangle = M^k\langle x_0\rangle$. M being stable would then imply that the recurrence converges to zero as desired.

So, it remains to be shown that we indeed have $\langle F_k\rangle = 0$: By definition of F_k, it suffices to show that $\langle a_k - \mathfrak{m}_k\rangle = 0$. Let $\mathcal{F}_k = \{F_0, \ldots, F_{k-1}\}$ denote the history of random increments. By the law of iterated expectations, we have $\langle a_k - \mathfrak{m}_k\rangle = \langle\langle a_k - \mathfrak{m}_k|\mathcal{F}_k\rangle_{a_k}\rangle_{\mathcal{F}_k}$ (here the subscripts next to the expectation brackets indicate which variables the expectations are taken over). We will show that the inner expectation $\langle a_k - \mathfrak{m}_k|\mathcal{F}_k\rangle_{a_k} = \langle a_k|\mathcal{F}_k\rangle_{a_k} - \langle \mathfrak{m}_k|\mathcal{F}_k\rangle_{a_k}$ is zero. By knowing $\mathcal{F}_k$, the state history $x_0, ..., x_k$ is deterministically determined via the recurrence of Eq. 9 (and vice versa).

Hence,

$$\begin{aligned}
\langle\cdot|\mathcal{F}_k\rangle &= \langle\cdot|\{x_0, ..., x_k, F_0, ..., F_{k-1}\}\rangle && (11)\\
&= \langle\cdot|\{x_0, ..., x_k, a_0, ..., a_{k-1}\}\rangle && (12)\\
&= \langle\cdot|\mathcal{D}_k\rangle && (13)
\end{aligned}$$

where the last step follows by our assumption of $\mathcal{D}_k = \{x_0, ..., x_k\}$, allowing the reconstruction of the $a_0, ..., a_{k-1}$. Thus, $\langle a_k|\mathcal{F}_k\rangle_{a_k} - \langle \mathfrak{m}_k|\mathcal{F}_k\rangle_{a_k} = \langle a_k|\mathcal{D}_k\rangle_{a_k} - \mathfrak{m}_k$ $\overset{Eq.6}{=} \mathfrak{m}_k - \mathfrak{m}_k = 0$. ∎

The theorem tells us that, when we can keep our data always up-to-date, the control actions as per Eq. 8 guarantee that our subjective (i.e. Bayesian) expectation over the controlled closed-loop trajectory succeeds in converging to the goal state $\xi = 0$ as desired. Investigating stronger notions such as mean-square stability will have to be deferred to future work.

At a cursory glance, our result that the expected trajectory is stable may not seem surprising since we always subtract the mean. However, it should be emphasised that for establishing this result, the assumption that the data is always kept up to date proved key (cf. Eq. 13). Of course, for fast sampling rates this is unrealistic and the mean $\mathfrak{m}_k$ we subtract will be conditional on a subset of the actual history. That is, $\mathcal{D}_k \subsetneq \mathcal{F}_k$. In this case, one might attempt to establish the desired result by marginalising over the unobserved elements of the increment history. In the standard theory of Ito stochastic differential equations, the orthogonality of the increments a_i would make this approach successful in establishing the desired result. Unfortunately though, in our situation, the uncertainty arises from the draw $a \sim \Pi^a$ where Π^a may introduce strong correlations. Developing a better understanding of the impact of these correlations on the expected closed-loop trajectory is something we consider an open problem and is deferred to future work.

III. Identification and control under completely uncertain control-affine dynamics

Above we have considered the case where the drift vector field $f(\cdot)$ was uncertain, but the control input vector field was completely known. This allowed us to compute training examples about the the drift value $f(x)$ based on knowing the control u and state observations (cf. Eq. 4). Next, we will consider the more general case where also b (or equivalently g, has to be learned. For simplicity, we assume fully-actuated control-affine dynamics where $b(x, u) = B(x)u$ and $B(x) \in \mathbb{R}^{m\times m}$ is invertible for all inputs $x \in \mathcal{X}$.

Consequently, our discrete dynamics of Eq. 3 can be written in the form:

$$d_k = f(x_k) + G(x_k)u_k \tag{14}$$

where we have defined $G(x) := \begin{pmatrix} O_m \\ B(x) \end{pmatrix} \in \mathbb{R}^{d\times m}$ and

$$d_k := \frac{1}{\Delta}(x_{k+1} - E\, x_k) \tag{15}$$

To further simplify the exposition, we assume $B(x) = (b_1(x)|\ldots|b_m(x))$ for some vector-valued functions $b_1 : \mathbb{R}^d \to \mathbb{R}^m, \ldots, b_m : \mathbb{R}^d \to \mathbb{R}^m$ bounded away from zero.

This is a realistic assumption for many dynamical systems, including the rigid-body mechanical systems considered below.

To learn about a and the b_j simultaneously, we will set up a cascade of learners, one for each of the vector fields. In order to separate the contribution of $B(x)u$ and $a(x)$ to an observed state transition, we can attempt to eliminate the influence of the β_j in order to learn about a. Fortunately, in a control-affine system this can be done by setting the control input signal to zero.[1] Conversely, learning about each $b_j(x)$ can be done in states x where $a(x)$ is relatively certain. This suggests an interleaved online learning approach which we will describe next.

Epistemic uncertainty and learning. Both dynamics functions a and b can be uncertain *a priori*. That is, *a priori* our uncertainty is modelled by the assumption that $a \sim \Pi^a, b_1 \sim \Pi^{b_1}, \ldots, b_m \sim \Pi^{b_m}$ where $\Pi^a, \Pi^{b_1}, \ldots, \Pi^{b_m}$ are random fields. The processes reflect our epistemic uncertainty about the true underlying (deterministic) dynamics functions a and b. That is, all probabilities have to be interpreted in a Bayesian manner.

If data becomes available over the course of the state evolution, we can update our beliefs over the dynamics in a Bayesian fashion. That is, at time $k \in I \subseteq N$ we assume $a \sim \Pi^a|\mathcal{D}_k, b_1 \sim \Pi^{b_1}|\mathcal{D}_k, \ldots, b_m \sim \Pi^{b_m}|\mathcal{D}_k$ where $\mathcal{D}_k$ is the data recorded up to time step k.

Data collection. We assume our controller can be called at an ordered set of time steps $I_u \subset I$. At each time step $k \in I_u$, the controller is able to observe the state x_k (possible up to some stochastic noise) and to set the control input $u_k = u(k, x_k)$. The controller may choose to evoke learning at an ordered subset $I_\lambda \subset I_u$ of time steps. To this end, at each time step $\tau \in I_\lambda$, the controller evokes a procedure explicated in Sec. III-.1 if it decides to incorporate an additional data point $(\tau, x_\tau, u_\tau, x_{\tau+1})$ into data set $\mathcal{D}_{\tau+1}$. The decision on whether to update the data will be based on the belief over the data point's anticipated informativeness as approximated by the variance (of the pertaining random vector the data point would instantiate).[2]

For simplicity, we assume that learning can occur every Δ_λ time steps and the controller is called every $\Delta_u \leq \Delta_\lambda$ time steps. For simplicity, as before, we assume $\Delta_u = 1$. A continuous control takes place in the limit of infinitesimal time step duration Δ.

1) Learning procedure: As before, the data sets $\mathcal{D}_k$ are found incrementally in an online learning fashion. Since it is hard to use the data to infer a and b simultaneously, we will have to actively decide which one we desire to learn about (and set the control accordingly – which we will henceforth refer to as a *separating control*). To this end, we distinguish between the following learning components:

- *Learning the uncertain drift component* $a(\cdot)$: Assume we are at time step $k \in I_\lambda$ and that we decide to learn about a. This decision is made, whenever our uncertainty about $a_k := a(x_k)$, encoded by $|||\text{Var}[a_k]|||$, is above a certain, predefinable threshold θ^a_{var}. To learn about $a(x_k)$ we would like to observe its value. Remember $a(x_k)$ are the last n components of $f(x_k)$. Rearranging Eq. 14 we see

$$f(x_k) = d_k - G(x_k)u_k \tag{16}$$

where d_k is computed from x_k and its successor state as per Eq. 15. If $G(x_k)$ is known, we simply wait another time step to also observe x_{k+1} and generate the desired training example (x_k, a_k) as we did above. If $G(x_k)$ is uncertain however, it is possible to disable its influence by choosing a *probing control action* $u_k := 0$. This choice allows us to compute the training example by storing the last n components of $f(x_k) = d_k$ as a sample of the value a_k. That is, we compute

$$a(x_k) = P\, d_k \tag{17}$$

where P is a matrix projecting a vector onto its last n components.

For the most part, we assume the states to be directly observable. However, to accommodate noisy observations, we might assume that instead of observing $f(x_k)$ directly, above equation only allows us to compute a noisy version $\tilde{f}_k = f(x_k) + \nu_k$ where ν_k is a zero-mean random vector with variance-covariance matrix $\text{Var}[\nu_k]$. We will discuss the connection to noisy state observations in an extended version of this paper. For the time-being, it will suffice to note that for normally distributed noise, incorporation of the noisy samples in the posterior inference can be done with ease within the Gaussian process learning framework [20].

- *Learning* $b_j(x)$*:* At time step $k \in I_\lambda$, we choose to learn about function b_j whenever our uncertainty about a_k is sufficiently small (i.e. $|||\text{Var}[a_k]||| \leq \theta^a_{\text{var}}$) and our uncertainty about b_j is sufficiently large ($|||\text{Var}[b_j(x_k)]||| > \theta^b_{\text{var}}$) and maximal, i.e. $j \in \text{argmax}_{i=1,\ldots,m} |||\text{Var}[b_i(x_k)]|||$. Let $e_j \in \mathbb{R}^m$ be the jth unit vector. To learn about $b_j(x_k)$ at state x_k, we apply a probing control action $u := u_j e_j$ where $u_j \in \mathbb{R}\backslash\{0\}$. Inspecting Eq. 16 it is clear that $b_j(x_k)$ coincides with the last n components of $\frac{1}{u_j}\big(d_k - f(x_k)\big)$. That is, to generate the desired training example $(x_k, b_j(x_k))$ we observe the states x_k and x_{k+1} and compute

$$b_j(x_k) = \frac{1}{u_j} P\big(d_k - \tilde{f}(x_k)\big). \tag{18}$$

Under the observational noise assumption we might again have to deal with noisy training examples computed as per $\tilde{b}_j(x_k) = b_j(x_k) + \eta_{k,j}$ where again $(\eta_{k,j})_{k\in\mathbb{N}}$ is an i.i.d. noise process with zero mean and variance-covariance matrix $\text{Var}[\eta_{k,j}]$ defined to model state-observation errors.

Consequently, at time $k+1$, $\tilde{b}_j(x_k)$ is a random vector

[1] An alternative approach is suggested in the future work section.

[2] Variance is known to approximate entropic measures of uncertainty (cf. [1]) and often easier to compute than entropy.

with posterior mean

$$\langle \tilde{b}_j(x_k)|\mathcal{D}_{k+1}\rangle = \frac{-1}{u_j} P\langle f(x)|\mathcal{D}_{k+1}\rangle = \frac{-1}{u_j}\langle a(x)|\mathcal{D}_{k+1}\rangle \tag{19}$$

and variance-covariance matrix

$$\text{Var}[\tilde{b}_j(x)] = \text{Var}[\eta_{k,j}] + \text{Var}[b_j(x_k)] \tag{20}$$

$$\leq \text{Var}[\eta_{k,j}] + \frac{1}{u_j^2}\text{Var}[\nu_k] + \frac{1}{u_j^2} V^a_{k+1} \tag{21}$$

where V^a_{k+1} denotes the posterior variance-covariance matrix about $a(\cdot)$ given data $\mathcal{D}_{k+1}$. By construction $\left|\left|\left|V^a_{k+1}\right|\right|\right| \leq \theta^a_{\text{var}}$.

A. Special case- diagonal control matrix $B(x)$

In the general setup above, from any given state-transition, our probing actions allowed us to maximally learn about one of the columns b_j of matrix B. Fortunately, in many relevant mechanical systems, including the coupled pendula considered below, $B(x)$ is diagonal. That is, where there exist $\beta_1(x), ..., \beta_m(x) \in \mathbb{R}\backslash\{0\}$ such that $B(x) = \text{diag}(\beta_1, \ldots, \beta_m)$.

This allows us to treat each output dimension in isolation. Furthermore, one probing action u_k, whose components all are non-zero, suffices to learn simulatenous about all diagonal entries as per: $\beta_j(x_k) = \frac{1}{u_j} P\big(d_k - \tilde{f}(x_k)\big)$.

B. Control law

Unless the control actions are chosen to be probing actions designed to aid system identification (as described above), we will want to base our control on our probabilistic belief model over the dynamics. Given such an uncertain model, it remains to define a control policy $u : \mathbb{N} \times \mathcal{X} \to \mathcal{U}$ with desirable properties. In this work, we attempt to feedback-linearise on the basis of the posterior model gained during learning. This translates to the linearising control law:

$$u(k, x_k; u') := \langle B^\dagger(x_k)|\mathcal{D}_k, x_k\rangle[-\langle a(x_k)|\mathcal{D}_k, x_k\rangle + u'] \tag{22}$$

where $B^\dagger(x_k)$ is the Moore-Penrose pseudo-inverse. In fully-actuated systems, the true B is invertible and hence, $B^{-1}(x) = B^\dagger(x)$ for all inputs x.

In case of perfect identification, i.e. when the posterior expectations coincide with the ground truth dynamics (and if the pseudo-inverse coincides with the inverse), this controller yields the linearised, simple closed-loop dynamics

$$d_k = (o_m, u'_k)^\top \tag{23}$$

as we can see when substituting our control into Eq. 14. Here, $o_m\mathbb{R}^m$ is an m-dimensional zero vector.

The free parameter u' is the pseudo- or inner-loop control and can be set at will to control the state evolution in some desired manner. As discussed in Sec. II, if it is our objective to have exponentially quickly vanishing state magnitude in the limit of $k \to \infty$ it would suffices to set the pseudo-controller to the PD-law $u' = -[K_1 K_2]x_k$ where K_1, K_2 are positive definite $m \times m$ matrices.

More generally, if the objective is to track the state trajectory $(\xi_k)_{k\in\mathbb{N}}$ the pseudo-control can be set to

$$u'(k, x_k) := [K_1, K_2](\xi_k - x_k) \tag{24}$$

to guarantee vanishing error.

IV. Simulations

In this section, we illustrate our method by applying it to online identification of a simple simulated control-affine system. We explored our controller's performance applied to the discrtised dynamics of a simulated frictionless, torque-actuated double-pendulum as derived in [25]. In continuous-time the configuration q consisted of the two joint angles and the state $x \in \mathbb{R}^4$ consisted of two joint angle positions and velocities. The uncontrolled system is known to exhibit chaotic behaviour and is unstable for all states except zero. Furthermore, the double pendulum has been used as model for a simple two-link robotic manipulator [24].

The system could be controlled by applying a torque to each joint. Given an initial state $x_0 = [0; 0; -1; -1]$ (downward position, with negative initial velocity), the task was to drive the double-pendulum upwards and stabilise the state at $x_f = [\pi; \pi, 0, 0]$ (motionless upward position). Hence, in our notation, $m = n = 2$ and $d = 4$.

Our simulations are based on a first-order Euler approximation of the dynamics of this system, choosing a time discretisation interval $\Delta = 0.01$ [sec]. Here, matrix $B(x) = \text{diag}(\beta_1(x), \ldots, \beta_m(x))$ has diagonal form as considered in Sec. III-A and full feedback-linearisation is possible when B is known. However, we assume both drift field a and control input fields $\beta_1, ..., \beta_m$ are uncertain a priori.

Before learning, we modelled our prior beliefs over the drift and control input vector fields by assuming $a \sim \mathcal{GP}(0, K_a)$ and each $\beta_j \sim \log \mathcal{GP}(0, K_{\beta_j})$ $(j = 1, ..., 2)$ had been drawn from a normal and log-normal process, respectively. The latter assumption encodes a priori knowledge that control input function β_j can only assume positive values (but, to demonstrate the idea of cascading processes, we had discarded the information that each β_j was a constant). During learning, the latter process was based on a standard normal process conditioned on log-observations of $\tilde{\beta}_j$. To compute the control law, we need to convert the posterior mean over $\log b$ into the expected value over the β_j. The required relationship is known to be as follows:

$$\langle \beta_j(x)|\mathcal{D}_k, x\rangle = \exp\Big(\langle \log b_j(x)|\mathcal{D}_k\rangle + \frac{1}{2}\text{var}[\log b(x)|\mathcal{D}_k]\Big). \tag{25}$$

If required, the posterior variance can be obtained as

$$\text{var}[\beta_j(x)|\mathcal{D}_k, x] = \Big(\exp(2\langle \log \beta_j(x)|\mathcal{D}_k\rangle) + \text{var}[\log \beta_j(x)|\mathcal{D}_k]\Big) \exp\Big(\text{var}[\log \beta_j(x)|\mathcal{D}_k, x] - 1\Big).$$

Note, the posterior mean over each β_j increases with the variance of our normal process in log-space, and, the control law as per Eq. 22 is inversely proportional to the magnitude

of this mean. Hence, the resulting controller is *cautious*, in the sense that control output magnitude is damped in regions of high uncertainty (variance). Depending on the situation, this can either be a curse or a blessing. If the system is stable under zero excitation the law has the advantage of gradually exploring the state space before moving to new unexplored parts. On the other hand, if zero excitation leads to instability, this behaviour may of course be problematic or at least not help (for instance think of a UAV falling from the sky under zero control action). In this case, it might be recommended to couple the controller with a stabilising feedback controller in a hybrid control setup.

Our learning-based controller, which we will refer to as *RF-SIIC* (which stands for *random field system identification and inversion control*) was initialised with a the prior as described above. Kernels K_a, K_{β_1} and K_{β_2} were chosen to be from the class of rational quadratic kernels with automated relevance detection (RQ-ARD). The observational noise variance was set to 0.01. The log-normal process over $b(\cdot)$ was implemented by placing a normal over each $\log \beta_j(\cdot)$ with zero mean and RQ-ARD kernel with fixed observational noise level 0.02. Note, the latter was set higher to reflect the uncertainty due to Π^a. In the future, we will consider incorporating hetereoscedastic observational noise based on $\mathrm{Var}[a(x)]$ and the sampling rate. Also, one could incorporate knowledge about periodicity in the kernel.

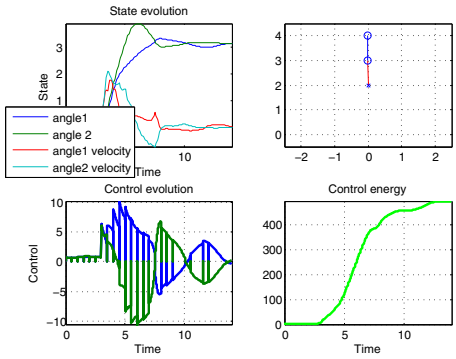
To showcase the learning behaviour, we conducted the experiment in three stages.

(I) As always, learning was done by conditioning on the observed training examples. However, in this first run, every third learning step, we allowed for full hyper-parameter training by optimising the marginal log-likelihood (see [20]). This hyper-parameter optimisation can make a significant difference if the ground truth dynamics are unlikely under the presupposed prior. Results are depicted in Fig. 1(a) and 1(b). Our RF-SIIC method managed to stabilise the system at the goal state after 14 seconds (corresponding to 1400 time steps) of control and online learning. Note, the drops in control signal (see Fig. 1(a)) are the probing actions performed to extract training examples of the drift $a(\cdot)$.

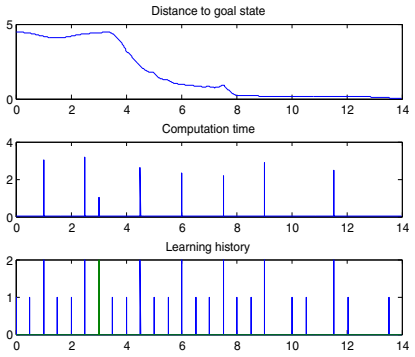
(II) The previous experiment was restarted, but with the learner pre-trained from Exp. 1 and without hyper-parameter optimisation. That is, learning was based on conditioning only. The results are depicted in Fig. 2(a) and 2(b). Observe, the controller benefits from the learning experience from the previous round evoking fewer learning steps yielding faster convergence to the goal. Furthermore, the processing time for controller calls is drastically reduced due to the absence of hyper-parameter optimisation. Furthermore, the improved posterior model and faster convergence results in a marked reduction of expended control energy.

(III). Once again, the experiment was restarted. This time, however, the learner was switched off so that the controller had to rely on the posterior models trained during the previous two rounds. The learner controller successfully drove the system to the goal (see Fig. 3(a) and 3(b)).

(IV). The ultimate aim of our controller is to track the



(a) Control evolution and state with untrained prior.

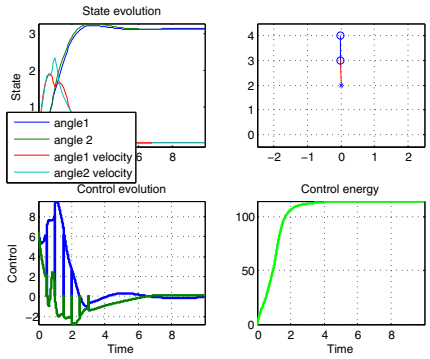


(b) Performance history as a function of time (sec).

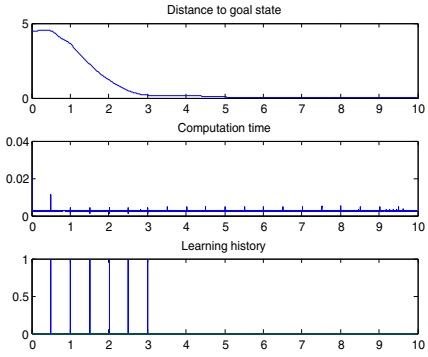
Fig. 1. Exp. I. Fig. 1(a): Control and state history. Bottom left plot: Blue curve: $u_1(x,t)$. Green curve: $u_2(x,t)$. Fig. 1(b): Evolution of distance to goal (top), computation time t = Δk [sec.] of the controller and record of when learning took place (bottom). For the latter, values have the following meaning: 0: no learning took place, 1: learning by conditioning only, 2: full learning, including hyper-parameter optimisation.

reference trajectory $(r_k)_{k\in\mathbb{N}}$ given by (the discretised version of) the perfectly linearised double-integrator dynamics. An alternative approach previously considered in robotics (cf. [18]) would be to employ a Gaussian process to learn an inverse dynamics model $\psi : (x_k, x_{k+1}) \mapsto u$ and choose the inverse model control policy $u(x_k) = \psi(x_k, r_{k+1})$. We refer to this method as *Gaussian process inverse model learning control (GP-IMLC)*.

In a final experiment, we compare our RF-SIIC approach to the GP-IMLC method. To this end, we randomly choose a sub-sample of 50 trajectory points (x_k, x_{k+1}, u_k) recorded in Exp. III and use it to train the random fields of *RF-SIIC* a well as the Gaussian process of the *GP-IMLC* controller. The reference trajectory mimics a double-integrator driving the double-pendulum from the start state $x_0 = [0; 0; 0; 0]$ to the goal state $x_f = [\pi; \pi, 0; 0]$. The results are depicted in Fig. 4. Note, the *RF-SIIC* is much more capable of taking advantage of the small subsample to accurately track the reference than GP-IMLC is. Perhaps this might be unsurprising as GP-IMLC faces a higher-dimensional learning problem and due the fact that the inverse mapping might not even be unique.

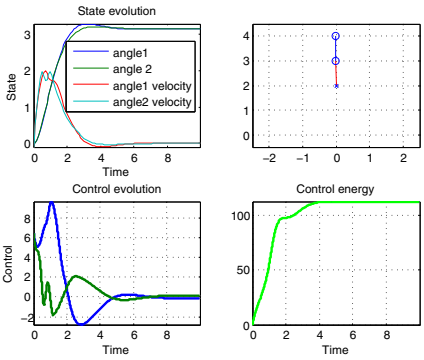


(a) Control and state evolution.

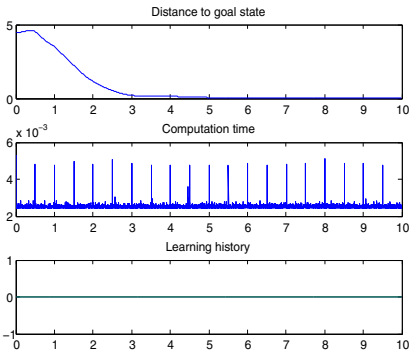


(b) Performance history as a function of time (s).

Fig. 2. Exp. II. Repetition of Exp. I with pre-trained controller and without hyper-parameter optimisation. Bottom left plot: Blue curve: $u_1(x,t)$. Green curve: $u_2(x,t)$.



(a) Control evolution of RF-SIIC.



(b) Performance history as a function of time (s).

Fig. 3. Exp. III. Repetition of Exp. I with pre-trained controller and without any further learning.

V. Discussion and Future Work

We have proposed a learning-based control approach, RF-SIIC, that combines learning with random fields and adaptive feedback linearisation to learn a control policy in a Bayesian fashion. In contrast to most related methods, our approach takes into account the structure of the dynamics equation and can learn drift and control input vector fields separately. This is interesting from a system identification point of view as it allows for a more detailed understanding of the physics of the underlying system. Furthermore, if the drift changes (e.g. due to change in the plant's environment) the identified control input function remains valid and the system does not have to be relearned from scratch. In addition the structural knowledge allowed the learners to learn forward models whose dimensionality equalled the dimensionality of state space. This is in contrast to many competing methods that require learning on the joint state-action space [5], [16].

For the control of discrete-time systems, we were able to leverage the structural knowledge to provide a guarantee of convergence of the expected closed-loop trajectory to the desired goal state. Here, it is to be emphasized that since all probabilities are degrees of subjective beliefs, the convergence guarantee also is a guarantee about epistemic beliefs over the (deterministic) dynamic system behaviour.

Our simulations have illustrated our controller's behaviour

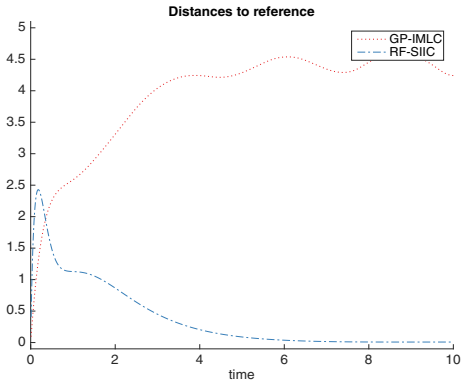


Fig. 4. Exp. IV. Comparison to the *Gaussian process inverse model learning control (GP-IMLC)* approach [18] based on 50 randomly chosen training examples from the history of Exp. III. The SIIC approach largely succeeds in tracking the desired reference trajectory while the GP-IMLC method fails to track the trajectory.

in the context of simple simulated rigid body systems and served as first demonstration of the viability of the approach. They show that our approach can be successful in simultaneous online learning and control and that it is fast enough to be applied at high sample rates. Furthermore, utilising the variance as a learning criterion the online learning process was able to keep the training corpora small.

The efficiency of our learning and control methods comes at a price. In particular, if both the drift and the input mappings B are uncertain, the need to distinguish between the two motivated us to set the control input to zero for brief periods of time in order to learn about the drift. This can be very disadvantageous in settings where such "zero-spikes" can destabilise the system and future work will investigate ways to circumvent this. Furthermore, to learn about input mapping B, we need to be reasonably certain about the drift at the state where b is to be learned. This can imply that data about b will often be much sparser than the data available for the model of a.

A. Future work.

So far, bounded control is not considered. While this could be modelled by squashing the control output through a bounded function (e.g. in lieu of [7]), the present absence of a planning method precludes the controller from solving tasks such as swing-ups under bounded control. The latter would involve forecasting and planning. Future work could address this and seek to combine our RF-SIIC approach in combination with MPC. One idea would be to replace the pseudo-controller by a predictive controller that is capable of avoiding obstacles in state space. We might then explore how to add virtual obstacles to state space, preventing the controller from choosing actions that steer the state into a region of state space that is likely to cause the actuators to saturate. A control sequence that connects a start state with a desired goal state in free-space should then solve the control problem of the closed-loop dynamic system under constrained control.

Utilising random fields to learn the dynamics has the benefit of providing uncertainty quantifications that can be utilised for deciding when to learn and when not. In the present work, we have used the variance as such a criterion. However, as we have explored in the context of the pendulum experiments, the variance is a subjective quantity that may be misleading if reality does not match the beliefs of the learner. Therefore, we will investigate alternative methods. As a simple first step for instance, we could imagine evoking learning, if and only if the observed state transitions do not match with the predictions made on the basis of the current model. Apart from possibly being a more effective criterion this would impose "never changing a winning team" behaviour, keep the data sets sparse and limit the number of aforementioned zero control spikes. Indeed, as discussed thesis version of this paper [3], it would be interesting to investigate a procedure that removes the necessity of injecting $u = 0$ as probing actions into the control signal and allows us to generate observations about a and b simultaneously on the basis of two distinct visitations of the same state.

REFERENCES

[1] T. Alpcan. Dual control with active learning using Gaussian process regression. *Arxiv preprint arXiv:1105.2211*, pages 1–29, 2011.

[2] H. Bauer. *Wahrscheinlichkeitstheorie*. deGruyter, 2001.

[3] Jan-Peter Calliess. *Conservative decision-making and inference in uncertain dynamical systems*. PhD thesis, University of Oxford, 2014.

[4] Girish Chowdhary, H.A. Kingravi, J.P. How, and P.A. Vela. Bayesian nonparametric adaptive control using Gaussian processes. Technical report, MIT, 2013.

[5] M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2013.

[6] MP Deisenroth, J. Peters, and C. E. Rasmussen. Approximate dynamic programming with Gaussian processes. *ACC*, June 2008.

[7] M.P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 2009.

[8] T.E. Duncan and B.Pasik-Duncan. Adaptive control of a scalar linear stochastic system with a fractional brownian motion. In *FAC World Congress*, 2008.

[9] H.Grimmett, R.Paul, R. Triebel, and I.Posner. Knowing when we don't know: Introspective classification for mission-critical decision making. In *ICRA*, 2013.

[10] J. Ko, D. Klein, D. Fox, and D. Haehnel. Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp. In *ICRA*, 2007.

[11] J. Kocijan and R. Murray-Smith. Nonlinear Predictive Control with a Gaussian. *Lecture Notes in Computer Science 3355, Springer*, pages 185–200, 2005.

[12] J. Kocijan, R. Murray-Smith, C.E. Rasmussen, and B. Likar. Predictive control with Gaussian process models. In *The IEEE Region 8 EUROCON 2003. Computer as a Tool.*, volume 1, pages 352–356. Ieee, 2003.

[13] P. R. Kumar. A survey of some results in stochastic adaptive control. *Siam J. Control and Optimization*, 23, 1985.

[14] Roderick Murray-smith, Carl Edward Rasmussen, and Agathe Girard. Gaussian Process Model Based Predictive Control. In *IEEE Eurocon 2003: The International Conference on Computer as a Tool*, 2003.

[15] D. Nguyen-Tuong and J. Peters. Using model knowledge for learning inverse dynamics. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[16] D Nguyen-Tuong and J. Peters. Model learning for robot control: a survey. *Cognitive processing*, 2011.

[17] D. Nguyen-Tuong, J. Peters, M. Seeger, and B. Schölkopf. Learning inverse dynamics: a comparison. In *Europ. Symp. on Artif. Neural Netw.*, 2008.

[18] Duy Nguyen-Tuong, Jan Peters, Matthias Seeger, and Bernhard Schölkopf. Learning inverse dynamics: a comparison. In *European Symposium on Artificial Neural Networks*, number EPFL-CONF-175477, 2008.

[19] Ioannou P. and J. Sun. *Robust Adaptive Control*. Prentice Hall, 1995.

[20] C.E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[21] Alex Rogers, Sasan Maleki, Siddhartha Ghosh, and N.R. Jennings. Adaptive Home Heating Control Through Gaussian Process Prediction and Mathematical Programming. In *2nd Int. Workshop on Agent Technology for Energy Systems (ATES 2011)*, 2011.

[22] A. Rottmann and W. Burgard. Adaptive Autonomous Control using Online Value Iteration with Gaussian Processes. In *ICRA*, 2009.

[23] M. W. Spong. Partial feedback linearization of underactuated mechanical systems. In *Proc. IEEE Int. Conf. on Intel. Robots and Sys. (IROS)*, 1994.

[24] M.W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Dynamics and Control*. Wiley and Sons, 2006.

[25] Russ Tedrake. Underactuated robotics: Learning, planning, and control for efficient and agile machines. Course Notes for MIT 6.832, 2009.

[26] K.Y. Volyanskyy, M.M. Haddad, and A.J. Calise. A new neuroadaptive control architecture for nonlinear uncertain dynamical systems: Beyond sigma- and e-modifications. In *CDC*, 2008.