Disentangling Sources of Uncertainty for Active Exploration

David Lines

Supervisor: Prof. C.E. Rasmussen
Dr M. Van Der Wilk

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy

Magdalene College August 2019

Declaration

I, David Lines of Magdalene College, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

This report makes use of the following software to aid in the production of the results:

1. SSGP code: Available at: http://www.tsc.uc3m.es/~miguel/simpletutorialssgp.php
    - Code modified to include sampling from the posterior distribution.
    - Used in the production of results in Section 3.2.
2. PILCO MATLAB code: Available at: http://mlg.eng.cam.ac.uk/pilco/
    - Code modified to include the Monte-Carlo uncertainty decomposition method presented in Section 2.4 and the SSGP code listed above.
    - Used in the production of results in Section 3.2.

Word count: 11,270

David Lines
August 2019

Abstract

PILCO is a learning algorithm that evaluates and minimises the expected cost function by cascading uncertain inputs through the GP dynamics model, which only allows it to make inferences based on the total uncertainty of the cost. Using variance as a metric for uncertainty, I apply the law of total variance to decompose total cost uncertainty into aleatoric and epistemic components. I develop a gold-standard Monte Carlo scheme to separately estimate each quantity by propagating trajectories through a finite-parameter trigonometric basis function approximation to PILCO's dynamics model. I show that when PILCO is learning efficiently, it is selecting policies associated with a high ratio of epistemic cost uncertainty to total cost uncertainty.

Acknowledgements

I would like to acknowledge my supervisors Professor C.E. Rasmussen and Dr M. Van Der Wilk for providing excellent advice and guidance throughout the course of the project.

Table of contents

List of figures

List of Algorithms

Chapter 1

Introduction

Probabilistic Inference for Learning Control or PILCO (Deisenroth and Rasmussen, 2011) is a model-based indirect policy search method for continuous state and action dynamical systems. PILCO evaluates a particular *policy* by assuming that the distribution over inputs is Gaussian and propagating that distribution through its probabilistic representation of the transition dynamics under a particular policy. In doing so, it gains insight into the variety of states that could be visited under that policy and is able to make inferences about how effective the policy is at achieving low cost.

However, this approach only allows PILCO to make decisions based on the total uncertainty as quantified by the models predictive distribution, when in fact there are two sources of uncertainty present in the system: *aleatoric* and *epistemic*. Aleatoric uncertainty is uncertainty due to unknowns that differ each time an agent encounters the environment (such as measurement noise) and is irreducible. Epistemic uncertainty arises from information that the agent could know in principle but currently does not, and can be reduced by observing more data. In model-based reinforcement learning, epistemic uncertainty is due to uncertainty about the model parameters. This means that in some situations PILCO could be exploring regions in the state space that are irrelevant for the task of learning control because it is making decisions based on the total uncertainty without knowledge of its constituents. In this case it could be targeting areas of high aleatoric uncertainty which could prohibit learning.

In this dissertation I investigate how PILCO uses uncertainty in its decision making process. In particular, I disentangle and quantify the different sources of uncertainty in the model of the transition function under a given policy. PILCO is a direct policy search method and is formulated so that the *cost function* is consulted directly; therefore, I examine the influence of the transition function uncertainty on cost uncertainty. I use variance as a metric for uncertainty and employ the law of total variance to decompose the total cost uncertainty into its constituents. I introduce a gold-standard Monte-Carlo scheme to separately estimate the aleatoric and epistemic uncertainties in the cost by propagating trajectories through an approximation to PILCO's dynamics model. Finally, I show that when PILCO is learning efficiently it is selecting policies associated with a high ratio of epistemic cost uncertainty to total cost uncertainty.

The aim of this dissertation is to lay the foundations for an active-exploration scheme to complement PILCO. I make the following contributions:

- Present a variance decomposition that disentangles the sources of uncertainty in the model's predictive distribution that influence PILCO's cost under a particular policy π.
- Create a gold-standard Monte Carlo scheme that separates the two sources of uncertainty and quantifies them.
- Show that when PILCO is learning efficiently, it is selecting policies that correspond to a high ratio of epistemic cost uncertainty to total cost uncertainty.
- Provide well documented code to reproduce all the results in this dissertation (see Appendix B).

1.1 Background

Reinforcement learning is a general sequential decision making framework that is best described as learning a mapping from situations to actions in an attempt to maximise a numerical reward signal. The agent, or learner, is not told what to do and so must embark on a mission of trial-and-error to discover actions that produce the most reward. In many cases, the action taken not only influences the instantaneous reward but also the next situation or state, and therefore, all future rewards as a consequence. To find a set of optimal actions given a sequence of situations, the agent must then take into account the effect of an action on all future rewards. These two ideas; trial-and-error search and delayed reward, are the two most distinguishing features of reinforcement learning (Sutton and Barto, 2018).

Another highly influential idea is the trade-off between *exploration* and *exploitation*. For an agent to maximise the long-term reward it must favour actions which it has previously tried and found to be successful in yielding high reward. However, in order to discover those actions, it must first have tried actions that it had not previously selected. The agent must *exploit* its current knowledge of the system to gain a high reward but also *explore* new strategies to improve its *policy*. A dilemma arises in that exclusively executing either approach will lead to a failed task. The agent must interchangeably try both approaches and progressively tend towards actions that prove to be better at attaining high reward.

There are two main approaches to reinforcement learning; *model-based* and *model-free* methods. Model-free methods are explicit trial-and-error learners and directly use the experience they gain through interactions with an environment to make decisions. In contrast, model-based techniques use experience indirectly by building a model of the state transition dynamics and reward structure of the environment, and evaluate actions by searching this model (Gläscher et al., 2010). Model-free methods are therefore said to rely on *learning* whilst model-based methods primarily rely on *planning* (Sutton and Barto, 2018).

Recently, model-free methods have achieved impressive performance in a range of complex tasks such as playing Atari games, receiving only pixels and game scores as inputs (Mnih et al., 2015). These methods, however, typically require millions of interactions with the environment before they achieve reasonable performance levels. The large number of required interactions can prohibit the use of these algorithms in some domains; such as mechanical systems with components that quickly wear out (Deisenroth and Rasmussen, 2011) or safety-critical systems where carrying out many field trials is prohibitively expensive. With computational power increasing exponentially, the primary bottleneck in the deployment of real-world reinforcement learning applications is fast becoming the number of interactions with the environment (Wan et al., 2018).

Model-based methods use field trials to create a belief about the underlying environment. There are several advantages to this approach, amongst others; the learning process can be more sample-efficient (Deisenroth et al., 2013a), prior knowledge and experience can be integrated more easily (particularly when Bayesian models are used) (Lopes et al., 2012), and more recently the incorporation of *counterfactual* reasoning (making inferences about actions that were not actually taken) which can be complex to implement without an explicit model-based representation of the environment (Buesing et al., 2018). Model-based methods are, however, not without their challenges, building accurate representations of complex real-world environments is difficult and failing to do so can lead to highly suboptimal algorithm behaviour.

Until recently, model-based techniques had not been widely applied to real-world systems. One of the main reasons for this is they can suffer from model bias. This happens when, given a data set of observed state transitions, the learned model incorrectly assumes that it fully describes the environment, when in fact there are many plausible functions that

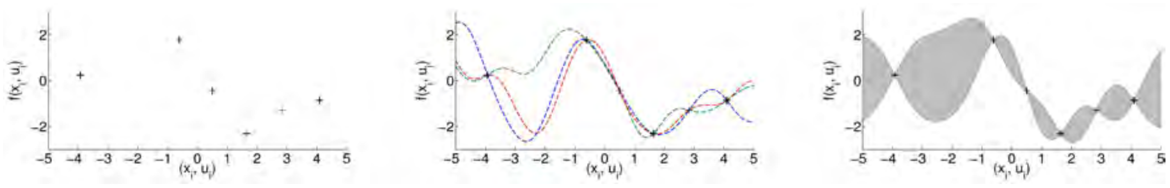


Fig. 1.1 Small data set of observed state transitions (left). Several plausible functions that could have generated the data (centre). Posterior distribution showing model uncertainty (right). Reproduced from (Deisenroth and Rasmussen, 2011).

could have generated the data (Atkeson and Santamaria (1997); Schneider (1997)). This is demonstrated in Fig 1.1 where a small data set of transitions are observed (left panel) and several transition functions exist that could have generated the data (centre panel). Selecting any single transition function can have severe consequences because predictions are then arbitrary at positions away from the data, but are claimed with full confidence (Deisenroth and Rasmussen, 2011).

PILCO (Deisenroth and Rasmussen, 2011) is a model-based indirect policy search method for continuous state and action dynamical systems. PILCO learns a probabilistic Bayesian representation of the dynamical systems it seeks to govern. Bayesian models explicitly quantify their uncertainty, and PILCO uses this information in a principled way to reduce model bias by explicitly incorporating model uncertainty into long-term planning. Fig 1.1 (right panel) shows how PILCO represents the observed data by placing a posterior distribution over the transition function.

PILCO reports unprecedented data-efficiency for a variety of control tasks, such as the cart-pole and cart-double-pole problems. Surprisingly, it does so without any intentional exploration i.e. it is *greedy*. Any exploration that does occur comes from one of three sources: first, the presence of stochasticity in the system; second, the use of a saturating cost function resulting in the controller favouring uncertain states (Deisenroth et al., 2013a); third, future controllers being informed of the performance of past controllers.

Attempting to further increase PILCO's data-efficiency would require either more informative prior knowledge of the task or extracting more relevant information from the available data. The addition of a exploration scheme would constitute the latter. Many approaches to learning control realise exploration through introducing randomness into action selection. Some approaches comprise a *random exploration phase*, where the controller generates actions randomly, followed by an *exploitation phase* (Thrun and Möller, 1992). However, with this strategy once the exploration phase has finished, the agent is unable to adapt to environmental changes during the purely exploitative phase. Others use ε-*greedy* policies, which exploit the action with the maximum expected reward most of the time, but with some probability ε a random action is selected (Sutton and Barto, 2018). While ε-*greedy* approaches encourage learning, for real-world systems, selecting random actions can repeatedly steer the system towards undesirable or dangerous states. In addition, random action selection can be inefficient and often causes the agent to repeatedly return to already well-explored regions of the state-space because exploration is *undirected*.

Information-directed or *active* exploration schemes aim to overcome the inefficiencies and risks associated with undirected exploration by driving exploration towards promising states (Guo and Brunskill, 2019). Since PILCO is primarily designed for learning control of mechanical dynamical systems, it is natural to consider this class of exploration algorithm when attempting to further improve its efficiency. Furthermore, since PILCO already quantifies model uncertainty and uses this uncertainty in long-term planning, it is also logical to attempt to incorporate this information in any considered exploration strategies.

Currently, PILCO evaluates a particular policy by cascading uncertain inputs through the probabilistic model of the transition dynamics. In doing so, it gains insight into the variety of states that could be visited under that policy and is able to make an informed decision about how effective the policy is at achieving low cost. However, this approach only allows PILCO to make decisions based on the total uncertainty as quantified by the model, when in fact there are two sources of uncertainty present in the system. One source of uncertainty is *aleatoric*; that is, representative of unknowns that differ each time the agent encounters the environment. Examples include measurement error or chaotic motion in dynamical systems that lead to stochastic transitions. The second source of uncertainty is *epistemic*; that is, arising from information that the agent could know in principle but currently does not. In model-based methods, this can be thought of as lack of knowledge about the system's transition dynamics. Hence, epistemic uncertainty can be reduced by observing more data while aleatoric uncertainty is irreducible. This means that in certain situations PILCO could be making decisions based on uncertainty of which the primary constituent is aleatory. In this case, PILCO could be repeatedly selecting policies corresponding to trajectories associated with high aleatoric uncertainty which could be prohibitive to learning.

This dissertation attempts to disentangle and quantify the different sources of uncertainty (aleatoric and epistemic) present in the model of the transition function. Since PILCO is a direct policy search method, the *cost function* is consulted directly, therefore, the influence of the uncertainty in the transition function on the uncertainty in the cost is examined. This is done by using variance as a metric to quantify uncertainty and employing the law of total variance to decompose the total model uncertainty into its constituents. The uncertainties are then estimated, establishing a "gold-standard" Monte-Carlo scheme that propagates trajectories through PILCO's dynamics model. The intention of the research is to lay the foundations for an active-exploration scheme to complement PILCO.

1.2 Related Work

The use of uncertainty for exploration in reinforcement learning is well-studied, either to improve sample efficiency (Schneider, 1997) (Jung and Stone, 2010) (Deisenroth and Rasmussen, 2011) or avoid worst-case scenarios and mitigate model bias (Bagnell et al., 2001) (Nilim and El Ghaoui, 2005) (Kahn et al., 2017) (Deisenroth et al., 2013a). While the majority of prior work focuses on estimation methods for either aleatoric or epistemic uncertainty, there has recently been increasing interest in *decomposing* total uncertainty into these two components, which can be used independently in the decision making process.

Prior work that estimates aleatoric uncertainty, or aleatoric risk, has focused on both the randomness in the environment and the agent's actions which lead to stochastic returns. Tamar et al. (2016) extend temporal difference methods to estimate the variance of the reward distribution by jointly estimating the second moment of the reward-to-go and the value function with a linear function approximator. The authors then derive a relationship between the two estimates to quantify risk. Prashanth and Ghavamzadeh (2013) define variance measures for policies to get risk-sensitive criteria for optimisation which are used alongside actor-critic gradient estimators. Other attempts at creating risk-sensitive agents include the development of distributional reinforcement learning methods which approximate the entire distribution of returns. Morimura et al. (2010) extend the Bellman equations to include the cumulative return distribution and propose an algorithm that leads to a risk-sensitive reinforcement learning paradigm. Bellemare et al. (2017) argue the fundamental importance of the value distribution in reinforcement learning and present theoretical results in both the policy evaluation and control settings.

Several methods for estimating the epistemic uncertainty have been proposed. Azizzadenesheli et al. (2018) and Lipton et al. (2018) both present exploration schemes that take into account the epistemic uncertainty by performing Bayesian inference over the parameters that define the value function. Pearce et al. (2018) estimate parameter uncertainty by regularising the parameters of an ensemble of neural networks about their initialisation values, instead of zero, and use these estimates to govern the exploration-exploitation process, resulting in steadier, more stable learning. Bellemare et al. (2016) take inspiration from the intrinsic motivation literature and quantify the uncertainty of an agent's knowledge through the use of density models. The authors present an algorithm for deriving a pseudo-count of state visits from an arbitrary density model which generalises count-based exploration algorithms to the non-tabular case. Finally, Gal and Ghahramani (2016) use Bayesian drop-out as an uncertainty estimate and provide a quantitative assessment of model uncertainty in the setting of reinforcement learning.

In the literature, a number of papers tackle the problem of producing estimates for both the aleatoric and epistemic uncertainties. Clements et al. (2019) build on the work of Gal and Ghahramani (2016) and Pearce et al. (2018) and show that the disagreement between only two neural networks is sufficient to produce a low-variance estimate of the epistemic uncertainty on the return distribution and estimate aleatoric risk through a distributional framework. Moerland et al. (2017) uses Bayesian drop-out to estimate epistemic uncertainty and aleatoric uncertainty by propagating the return uncertainty through the Bellman equation as a Gaussian distribution.

There are also a number of papers that consider both kinds of uncertainties in model-based reinforcement learning. Chua et al. (2018) propose a probabilistic ensembles with trajectory sampling (PETS) algorithm that combines uncertainty-aware deep network dynamics models with sampling-based uncertainty propagation which has the ability to isolate both sources of uncertainty. Finally, Depeweg et al. (2017) present two uncertainty decompositions for Bayesian neural network models with latent variables that isolate both sources of uncertainty. The first is in the form a variance decomposition and is similar to what is presented in this work and the second in terms of mutual information. The sources of uncertainty are then related to risk-sensitive reinforcement learning.

Chapter 2

Methodology

This section describes how the sources of uncertainty in PILCO's decision making process are disentangled and quantified. First, section 2.1 provides a brief introduction to the PILCO algorithm. In Section 2.2 a finite-parameter trigonometric Bayesian regression model is introduced to approximate PILCO's probabilistic Gaussian Process (GP) dynamics model. Section 2.2.2 relates the trigonometric model to a full GP by showing that it can be interpreted as sparse spectrum GP that can approximate any full GP. The sources of uncertainty in transition function and their effect on the cost are discussed in section 2.3 and a variance decomposition to separate and quantify the uncertainty into its aleatoric and epistemic constituents is presented. Finally, section 2.4 presents a "gold-standard" Monte-Carlo scheme to estimate the aleatoric and epistemic uncertainties.

2.1 PILCO

PILCO (Deisenroth and Rasmussen, 2011) is a model-based indirect policy search method for continuous state $\mathbf{x} \in \mathbb{R}^D$ and action $\mathbf{u} \in \mathbb{R}^F$ dynamical systems described by

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) + \eta, \quad \eta \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_\eta) \tag{2.1}$$

where the transition dynamics f of the system are unknown. The objective is to find a deterministic *controller/policy* $\pi : \mathbf{x} \mapsto \pi(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{u}$ such that the expected cost

$$J^\pi(\theta) = \sum_{t=0}^{T} \mathbb{E}_{\mathbf{x}_t}\left[\mathbb{C}(\mathbf{x}_t)\right], \quad \mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{\Sigma}_0) \tag{2.2}$$

is minimised after following π for T steps (Deisenroth et al., 2013a). The policy is parametrised by θ, which are the weights and features in the case of nonlinear RBF networks, or the weights matrix and bias terms in the case of linear-affine transformations. The *cost* $\mathbb{C}(\mathbf{x}_t)$, or negative reward, of being in state $\mathbf{x}$ at time t is the generalised binary saturating cost (Deisenroth et al., 2013a)

$$\mathbb{C}(\mathbf{x}_t) = 1 - \exp\left(-\frac{1}{2\sigma_c^2}||\mathbf{x}_t - \mathbf{x}_{\text{target}}||^2\right) \in [0,1] \tag{2.3}$$

which exclusively penalises the Euclidean distance $||\mathbf{x}_t - \mathbf{x}_{\text{target}}||$ from the current state $\mathbf{x}_t$ to the target state $\mathbf{x}_{\text{target}}$. $\mathbb{C}(\mathbf{x}_t)$ is locally quadratic and saturates at 1 for large differences between the current state and target state. The saturation distance is controlled by the width parameter σ_c.

2.1.1 GP Model Learning

The system dynamics are modelled as a probabilistic Gaussian Process (GP). Tuples of the state and action vectors $(\mathbf{x}_t, \mathbf{u}_t) \in \mathbb{R}^{D+F}$ serve as training inputs and differences $\Delta_t = \mathbf{x}_{t+1} - \mathbf{x}_t + \varepsilon \in \mathbb{R}^D, \quad \varepsilon \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\varepsilon)$ as training targets. Conditionally independent GP's are trained for each target dimension (Deisenroth, 2010). The dynamics model provides a *one-step* prediction (Deisenroth and Rasmussen, 2011)

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}(\mathbf{x}_t|\mu_t, \boldsymbol{\Sigma}_t) \tag{2.4}$$

$$\mu_{t+1} = \mathbf{x}_t + \mathbb{E}_f[\Delta_t] \tag{2.5}$$

$$\boldsymbol{\Sigma}_{t+1} = \text{var}_f[\Delta_t] \tag{2.6}$$

where the mean μ of the state $\mathbf{x}_{t+1}$ is obtained through addition of the current state $\mathbf{x}_t$ with the one-step difference prediction $\mathbb{E}_f[\Delta_t]$. A GP is defined completely by a mean function $m(\cdot)$ and a positive semidefinite covariance function $K(\cdot,\cdot)$. PILCO follows the common practice of setting the mean prior function to zero and uses the *stationary* anisotropic squared exponential covariance function

$$k(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) = \sigma_0^2 \exp\left(-\frac{1}{2}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)^\top \boldsymbol{\Lambda}^{-1}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)\right) \tag{2.7}$$

where $\tilde{\mathbf{x}} = [\mathbf{x} \quad \mathbf{u}]^T$ is the state-action vector; $\boldsymbol{\Lambda} = \text{diag}\left([\ell_1^2, \ldots, \ell_{D+F}^2]\right)$ depends on the characteristic lengthscales ℓ; and σ_0 is the variance of the latent function (Deisenroth et al., 2013a). The lengthscales determine the speed that the covariance decays with the distance between inputs (Quiñonero-Candela et al., 2010).

2.1.2 Policy Evaluation

PILCO evaluates a given policy by predicting the state evolution over the course of an episode. This is accomplished by propagating uncertain inputs through the dynamics model, using the one-step predictions (Eqs. 2.4 - 2.6), to obtain $p(\mathbf{x}_1|\pi),\ldots,p(\mathbf{x}_T|\pi)$ from a given start state distribution $p(\mathbf{x}_0)$. Predicting $\mathbf{x}_{t+1}$ from $p(\mathbf{x}_t)$ requires evaluation of the predictive distribution over state differences

$$p(\Delta_t) = \iint p(f(\tilde{\mathbf{x}}_t)|\tilde{\mathbf{x}}_t)\, p(\tilde{\mathbf{x}}_t)\, \mathrm{d}f \mathrm{d}\tilde{\mathbf{x}}_t, \tag{2.8}$$

by integrating out the random function of the GP distribution and the random variable $\tilde{\mathbf{x}}_t$. Here $p(\tilde{\mathbf{x}}_t) = p(\mathbf{x}_t, \mathbf{u}_t)$ is the joint state-action distribution. Since the predictive distribution in Eq. 2.8 is analytically intractable for uncertain inputs, it is approximated as a Gaussian distribution using *moment matching* (see Fig. 2.1). This approximation ensures that the state distribution is given by $p(\tilde{\mathbf{x}}_t) = \mathcal{N}\left(\tilde{\mathbf{x}}_t|\tilde{\mu}_t, \tilde{\mathbf{\Sigma}}_t\right)$ for all t. This, alongside the choice of cost function (Eq. 2.3) enables the expected return (Eq. 2.2) to be evaluated analytically according to

$$\mathbb{E}_{\mathbf{x}_t}\left[\mathbb{C}(\mathbf{x}_t)\right] = \int \mathbb{C}(\mathbf{x}_t)\,\mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_t, \mathbf{\Sigma}_t)\,\mathrm{d}\mathbf{x}_t \tag{2.9}$$

for $t = 1,\ldots,T$.

2.1.3 Policy Improvement

PILCO is an indirect policy search method and does not require an explicit value function model. Instead it learns a *parametrised policy* that can select actions without consulting a value function. Due to the moment matching approximation, the state distribution is considered to be Gaussian $p(\tilde{\mathbf{x}}_t) = \mathcal{N}\left(\tilde{\mathbf{x}}_t|\tilde{\mu}_t, \tilde{\mathbf{\Sigma}}_t\right)$ for all t, where $\tilde{\mathbf{x}}_t = (\mathbf{x}_t, \mathbf{u}_t)$. The control signal $\mathbf{u}_{t-1} = \pi(\mathbf{x}_{t-1}, \theta)$ is a function of the state, and hence the next state is also functionally dependent on the mean μ_u and covariance $\mathbf{\Sigma}_u$ of the control signal. This relationship allows the gradients of the expected return J^π, with respect to the policy parameters $\boldsymbol{\theta}$, to be computed analytically. The policy parameters are then learned to maximise performance (minimise cost) for a given task, so their updates approximate *conjugate gradient minimisation* in J

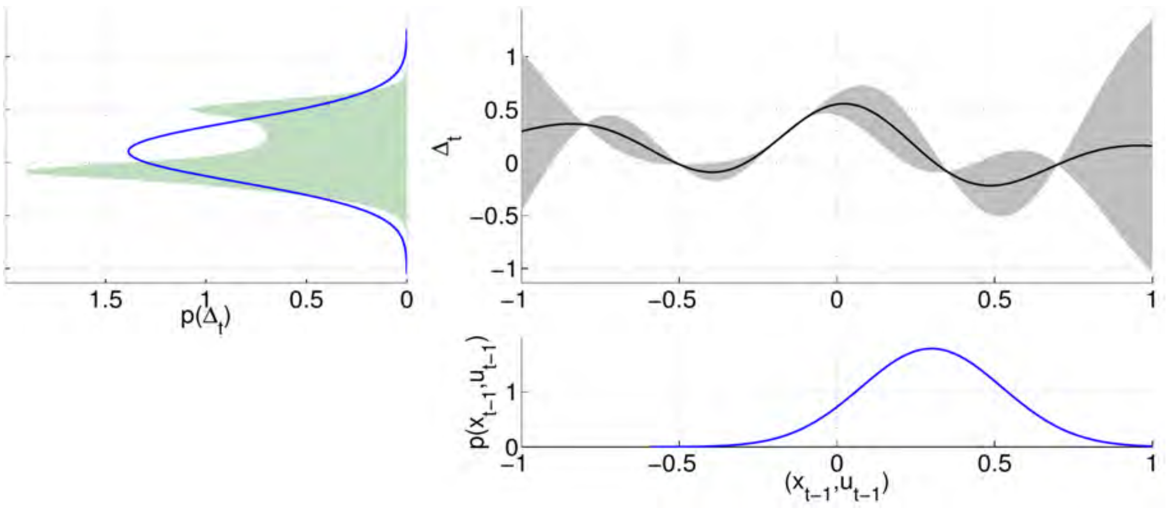


Fig. 2.1 Propagating an uncertain input through the GP dynamics model (upper right panel). The input distribution $p(x_{t-1}, u_{t-1})$, assumed to be Gaussian (lower right panel), is propagated through the dynamics model yielding the shaded distribution $p(\Delta_t)$. The shaded distribution is then approximated by a Gaussian with the same mean and variance (upper left panel). Reproduced from (Deisenroth and Rasmussen, 2011).

(Sutton and Barto, 2018):

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \nabla J^{\pi}(\boldsymbol{\theta}_t), \tag{2.10}$$

where α is the learning rate. For a more thorough explanation of PILCO see (Deisenroth and Rasmussen, 2011)(Deisenroth, 2010)(Deisenroth et al., 2013a). Algorithm 1 provides an overview of PILCO.

Algorithm 1 Probabilistic Inference for Learning Control (PILCO)

```
1:  init: Sample controller parameters $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2:  repeat
3:      Learn probabilistic GP dynamics model using all environment data      ▷ Sec 2.1.1
4:      repeat
5:          Approximate inference for policy evaluation: $J^{\pi}(\theta)$     ▷ Sec 2.1.2
6:          Gradient-based policy improvement: $\mathrm{d}J^{\pi}(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$     ▷ Sec 2.1.3
7:          Update parameters $\boldsymbol{\theta}$                            ▷ CG or L-BFGS
8:      until convergence; return $\boldsymbol{\theta}^*$
9:      Set $\pi^* \leftarrow \pi(\theta^*)$
10:     Apply $\pi^*$ to environment and record data
11: until task learned              ▷ modified from (Deisenroth and Rasmussen, 2011)
```

2.2 Approximating PILCO's GP

To approximate the uncertainty in PILCO's cost function using Monte-Carlo methods, it is necessary to make one-step predictions using a function sampled from the posterior distribution of PILCO's dynamics model. The use of the squared exponential covariance function in PILCO's GP corresponds to a Bayesian linear regression model with an infinite number of basis functions (Williams and Rasmussen, 2006). This means drawing a representative function requires an infinite number of weights since each basis function is accompanied by its own weight. To overcome this issue, a finite-parameter stationary trigonometric Bayesian regression model (Quiñonero-Candela et al., 2010) is used to approximate PILCO's GP. In addition to solving the sampling issue, this model provides other advantages. First, the periodicity of the trigonometric basis functions means that one does not need to explicitly specify the location for each basis function. Second, a direct GP implementation has practical limitations with regards to computational and memory requirements, which scale as $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$, respectively. The trigonometric model has computational requirements of $\mathcal{O}(nm^2)$ and memory requirements of $\mathcal{O}(nm)$ where $m << n$ (Quiñonero-Candela et al., 2010), which are typical values for sparse GP approximations (Quiñonero-Candela and Rasmussen, 2005). The largest PILCO environment used for this research (cart double pendulum) generates approximately $4k$ data points. The model is first introduced here in a traditional treatment of Bayesian linear regression. Section 2.2.1 relates the model to PILCO's one-step predictions. Finally, section 2.2.2 shows that the model can be viewed as a sparse stationary GP that can approximate any full GP.

The model consists of a linear combination of trigonometric functions (Quiñonero-Candela et al., 2010)

$$f(\tilde{\mathbf{x}}) = \sum_{r=1}^{m} a_r \cos\left(2\pi \mathbf{s}_r^\top \tilde{\mathbf{x}}\right) + b_r \sin\left(2\pi \mathbf{s}_r^\top \tilde{\mathbf{x}}\right) \tag{2.11}$$

where $\tilde{\mathbf{x}} = \left[\mathbf{x}^\top \; \mathbf{u}^\top\right]^\top \in \mathbb{R}^{D+F}$ is the state-action vector, $\mathbf{s}_r$ is a $(D+F)$-dimensional vector of spectral frequencies shared by each pair of basis functions and a_r, b_r are amplitude parameters which are independent for each basis function (see Sec 2.2.2 for selecting spectral frequencies). The amplitudes have independent Gaussian priors with linearly scaled variances

$$a_r \sim \mathcal{N}\left(0, \frac{\sigma_0^2}{m}\right), \quad b_r \sim \mathcal{N}\left(0, \frac{\sigma_0^2}{m}\right) \tag{2.12}$$

where m are the number basis functions. The frequencies function as deterministic parameters and the amplitudes are treated in a Bayesian fashion. For this, the model is packaged as the dot product between the set of amplitudes and the basis functions

$$f(\tilde{\mathbf{x}}, \mathbf{w}) = \mathbf{w}^\top \varphi(\tilde{\mathbf{x}}) \tag{2.13}$$

where $\mathbf{w} = [a_1, b_1, \dots, a_m, b_m]^\top$ are the model weights and

$$\varphi(\tilde{\mathbf{x}}) = \left[\cos\left(2\pi\mathbf{s}_1^\top\tilde{\mathbf{x}}\right) \quad \sin\left(2\pi\mathbf{s}_1^\top\tilde{\mathbf{x}}\right) \quad \dots \quad \cos\left(2\pi\mathbf{s}_m^\top\tilde{\mathbf{x}}\right) \quad \sin\left(2\pi\mathbf{s}_m^\top\tilde{\mathbf{x}}\right)\right]^\top. \tag{2.14}$$

Similar to PILCO, state differences Δ serve as the training targets which are assumed to have been generated by the function $f(\tilde{\mathbf{x}}, \mathbf{w})$ and independently corrupted by additive Gaussian noise of constant variance σ_n^2

$$\Delta = f(\tilde{\mathbf{x}}, \mathbf{w}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}\left(0, \sigma_n^2\right). \tag{2.15}$$

This can be written as

$$p(\Delta|\tilde{\mathbf{x}}, \mathbf{w}, \sigma_n^2) = \mathcal{N}\left(\Delta|f(\tilde{\mathbf{x}}, \mathbf{w}), \sigma_n^2\right). \tag{2.16}$$

For a dataset of model inputs $\tilde{\mathbf{X}} = \{\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_N\}$ with corresponding target values $\boldsymbol{\Delta} = \{\Delta_1, \dots, \Delta_N\}$ and assuming that these data points have been drawn independently from $\mathcal{N}(\Delta|f(\mathbf{x}, \mathbf{w}), \sigma_n^2)$, the likelihood function is

$$p(\boldsymbol{\Delta}|\tilde{\mathbf{X}}, \mathbf{w}, \sigma_n^2) = \prod_{i=1}^{N} \mathcal{N}\left(\Delta_i|\mathbf{w}^\top \varphi(\tilde{\mathbf{x}}_i), \sigma_n^2\right). \tag{2.17}$$

The posterior distribution over the model weights is proportional to the product of the likelihood function and the prior. Since the Gaussian prior is conjugate, the posterior distribution is also Gaussian (see Appendix A.1.1 for derivation)

$$q(\mathbf{w}|\boldsymbol{\Delta}, \tilde{\mathbf{X}}, \sigma_0^2, \sigma_n^2) = \mathcal{N}\left(\mathbf{w}|\mu_{\mathbf{w}}, \mathbf{A}^{-1}\right) \tag{2.18}$$

where the posterior mean $\mu_{\mathbf{w}}$ and precision matrix A are

$$\mu_{\mathbf{w}} = \frac{1}{\sigma_n^2}\mathbf{A}^{-1}\Phi\mathbf{y} \tag{2.19}$$

$$\mathbf{A} = \frac{1}{\sigma_n^2}\Phi\Phi^\top + \frac{m}{\sigma_0^2}\mathbf{I}_{2m}. \tag{2.20}$$

Here, $\Phi = [\varphi(\tilde{\mathbf{x}}_1), \ldots, \varphi(\tilde{\mathbf{x}}_N)]$ is the $2m$ by N *design matrix.*

To make a prediction Δ_* for a new input value $\tilde{\mathbf{x}}_*$ requires evaluation of the predictive distribution, defined by

$$p(\Delta_*|\boldsymbol{\Delta}, \tilde{\mathbf{X}}, \tilde{\mathbf{x}}_*, \sigma_0^2, \sigma_n^2) = \int p(\Delta_*|\tilde{\mathbf{x}}_*, \mathbf{w}, \sigma_n^2) q(\mathbf{w}|\boldsymbol{\Delta}, \tilde{\mathbf{X}}, \sigma_0^2, \sigma_n^2) \mathrm{d}\mathbf{w} \tag{2.21}$$

where the conditional distribution of the target variable $p(\Delta_*|\tilde{\mathbf{x}}_*, \mathbf{w}, \sigma_n^2)$ is defined by Eq. 2.16 and the posterior distribution over the weights is given by Eq. 2.18. The predictive distribution for a single input is (see Appendix A.1.2 for derivation)

$$p(\Delta_*|\boldsymbol{\Delta}, \tilde{\mathbf{X}}, \tilde{\mathbf{x}}_*, \sigma_0^2, \sigma_n^2) = \mathcal{N}\left(\Delta_*|\mu_{\Delta_*}, \sigma_{\Delta_*}^2\right) \tag{2.22}$$

where

$$\mu_{\Delta_*} = \frac{1}{\sigma_n^2} \varphi(\tilde{\mathbf{x}}_*)^\top \mathbf{A}^{-1} \Phi \boldsymbol{\Delta} \tag{2.23}$$

$$\sigma_{\Delta_*}^2 = \sigma_n^2 + \varphi(\tilde{\mathbf{x}}_*)^\top \mathbf{A}^{-1} \varphi(\tilde{\mathbf{x}}_*). \tag{2.24}$$

A note on model uncertainty: The predictive variance (Eq. 2.24) consists of two terms. The first represents the noise present on the data while the second quantifies the uncertainty associated with the model weights $\mathbf{w}$ (Bishop, 2006). The latter is particularly relevant to this research and will be expanded on in Section 2.3 and in the discussion (Section 3.3). As more data is observed, the model becomes increasingly confident and consequently the posterior distribution narrows due to contraction of the second term. Qazaz et al. (1997) show that $\sigma_{\Delta_*(N+1)}^2 \leqslant \sigma_{\Delta_*(N)}^2$ while in the limit $N \to \infty$ the second term of Eq. 2.24 reduces to zero. In this case the model has full confidence in the weights and the predictive variance's sole contributor is the data noise governed by the σ_n^2 term.

2.2.1 One-Step Predictions

The one-step predictions generated by PILCO's GP dynamics model are produced by the posterior predictive distribution, however, for the uncertainty estimates in Sec 2.3, functions drawn from the posterior distribution over the model weights are used to make one-step predictions. For this, first a set of $2m$ weights is drawn from the posterior distribution $\mathbf{w} \sim q(\mathbf{w})$ (Eq. 2.18) where the conditioning variables are dropped for brevity. The one-step predictions are then computed with

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta_t = \mathbf{x}_t + \mathbf{w}^\top \varphi(\tilde{\mathbf{x}}_t) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \Sigma_\varepsilon). \tag{2.25}$$

2.2.2 A Sparse Approximation to the Full GP

This section follows the treatment in Quiñonero-Candela et al. (2010) and relates the trigonometric basis function model of the previous section to a full GP by creating a sparse representation of the stationary covariance function. GP regression is a probabilistic, non-parametric Bayesian approach that is completely specified by a mean function and covariance function (Williams and Rasmussen, 2006)

$$\begin{aligned} m(\tilde{\mathbf{x}}) &= \mathbb{E}[f(\tilde{\mathbf{x}})] \\ k\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j\right) &= \mathbb{E}\left[\left(f(\tilde{\mathbf{x}}_i) - m(\tilde{\mathbf{x}}_i)\right)\left(f\left(\tilde{\mathbf{x}}_j\right) - m\left(\tilde{\mathbf{x}}_j\right)\right)\right]. \end{aligned} \tag{2.26}$$

Similar to the PILCO GP, a zero mean function $m(\tilde{\mathbf{x}}) = 0$ and stationary squared exponential covariance function

$$k\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j\right) = k\left(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\right) = k(\tau) \tag{2.27}$$

are considered (see Eq 2.7). A stationary covariance function is a function $\boldsymbol{\tau} = \tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j$ that depends only on the difference between inputs. Now considering the trigonometric basis function model in Eq 2.11. Under the prior, the distribution over functions is Gaussian with zero mean and stationary covariance function (Quiñonero-Candela et al., 2010)

$$k\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j\right) = \frac{\sigma_0^2}{m} \varphi\left(\tilde{\mathbf{x}}_i\right)^\top \varphi\left(\tilde{\mathbf{x}}_j\right) = \frac{\sigma_0^2}{m} \sum_{r=1}^{m} \cos\left(2\pi \mathbf{s}_r^\top \left(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\right)\right). \tag{2.28}$$

Theorem 2.2.1 (Bochner's theorem) *A complex-valued function k on $\mathbb{R}^D$ is the covariance function of a weakly stationary mean square continuous complex-valued random process on $\mathbb{R}^D$ if and only if it can be represented as*

$$k(\boldsymbol{\tau}) = \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s} \cdot \boldsymbol{\tau}} d\mu_{fm}(\mathbf{s}) \tag{2.29}$$

where μ_{fm} is a positive finite measure. ▷ *Quoted from (Stein, 1999)*

If μ_{fm} has a probability density $S(\mathbf{s})$ then S is the *spectral density* associated with k (Williams and Rasmussen, 2006) and is proportional to a probability measure $S(\mathbf{s}) \propto p_S(\mathbf{s})$. The proportionality constant can be obtained by evaluating the covariance function in Eq 2.27 at $\boldsymbol{\tau} = \mathbf{0}$ to give

$$S(\mathbf{s}) = k(\mathbf{0})p_S(\mathbf{s}) = \sigma_0^2 p_S(\mathbf{s}). \tag{2.30}$$

Should $S(\mathbf{s})$ exist, according to the Wiener-Khintchine theorem (Chatfield, 1989), the covariance function and the spectral density form a Fourier pair

$$k(\boldsymbol{\tau}) = \int S(\mathbf{s}) e^{2\pi i \mathbf{s}\cdot\boldsymbol{\tau}} d\mathbf{s}, \quad S(\mathbf{s}) = \int k(\boldsymbol{\tau}) e^{-2\pi i \mathbf{s}\cdot\boldsymbol{\tau}} d\boldsymbol{\tau}. \tag{2.31}$$

Since $S(\mathbf{s})$ is proportional to a multivariate probability density in $\mathbf{s}$ the covariance function can be rewritten as an expectation with respect to the probability measure $p_S(\mathbf{s})$

$$\begin{aligned} k(\boldsymbol{\tau}) &= \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}^\top (\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)} S(\mathbf{s}) \mathrm{d}\mathbf{s} \\ &= \sigma_0^2 \int_{\mathbb{R}^D} e^{2\pi i \mathbf{s}^\top \tilde{\mathbf{x}}_i} \left(e^{2\pi i \mathbf{s}^\top \tilde{\mathbf{x}}_j}\right)^* p_S(\mathbf{s}) d\mathbf{s} \\ &= \sigma_0^2 \mathbb{E}_{p_S(\mathbf{s})} \left[e^{2\pi i \mathbf{s}^\top \tilde{\mathbf{x}}_i} \left(e^{2\pi i \mathbf{s}^\top \tilde{\mathbf{x}}_j}\right)^* \right] \end{aligned} \tag{2.32}$$

where the superscript asterisk denotes complex conjugation. The result in Eq. 2.32 can be estimated by simple Monte-Carlo. Since the spectral density is symmetric about zero, sampling frequency pairs $\{\mathbf{s}_r, -\mathbf{s}_r\}$ preserves the exact expansion where the imaginary terms cancel

$$\begin{aligned} k\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j\right) &\simeq \frac{\sigma_0^2}{2m} \sum_{r=1}^{m} \left[e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_i} \left(e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_j}\right)^* + \left(e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_i}\right)^* e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_j} \right] \\ &= \frac{\sigma_0^2}{m} \operatorname{Re} \left[\sum_{r=1}^{m} e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_i} \left(e^{2\pi i \mathbf{s}_r^\top \tilde{\mathbf{x}}_j}\right)^* \right] \\ &= \frac{\sigma_0^2}{m} \sum_{r=1}^{m} \cos\left(2\pi \mathbf{s}_r^\top \left(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\right)\right). \end{aligned} \tag{2.33}$$

Here, *Re* is the real part of the complex number and the finite set of Monte-Carlo frequencies $\mathbf{s}_r$, called *spectral points*, are sampled from $p_S(\mathbf{s})$. This result shows that Eq. 2.28 is recovered by sparsifying the stationary covariance function of a full GP meaning that the trigonometric basis function model of the previous section is indeed a sparse spectrum Gaussian process.

Taking the Fourier transform of the squared exponential covariance function used in PILCO's GP dynamics model gives a probability density of multivariate Gaussian form

$$p_S(\mathbf{s}) = \frac{1}{k(\mathbf{0})} \int_{\mathbb{R}^D} e^{-2\pi i \mathbf{s}^\top \tau} k(\tau) d\tau = \sqrt{|2\pi\Lambda|} \exp\left(-2\pi^2 \mathbf{s}^\top \Lambda \mathbf{s}\right) \tag{2.34}$$

from which the spectral points are drawn for the Monte-Carlo estimate of $k(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$.

2.2.3 Hyperparameter Optimisation

In a fully Bayesian treatment of the trigonometric basis function model one would introduce priors over σ_0^2 and σ_n^2 and marginalise with respect to both the hyperparameters and the weights $\mathbf{w}$ to make predictions. However, while marginalisation over either the hyperparameters or the weights is plausible, complete marginalisation over both is analytically intractable (Bishop, 2006). Instead, here the hyperparameters are learned through optimising the log marginal likelihood

$$\log p(\boldsymbol{\Delta}|\boldsymbol{\theta}) = -\frac{1}{2\sigma_n^2}\left[\boldsymbol{\Delta}^\top\boldsymbol{\Delta} - \frac{1}{\sigma_n^2}\boldsymbol{\Delta}^\top\boldsymbol{\Phi}^\top\mathbf{A}^{-1}\boldsymbol{\Phi}\boldsymbol{\Delta}\right] - \frac{1}{2}\log|\mathbf{A}| + m\log\frac{m}{\sigma_0^2} - \frac{n}{2}\log 2\pi\sigma_n^2 \tag{2.35}$$

with respect to the hyperparameters σ_0^2, σ_n^2, and the lengthscales $\{\ell_1, \ldots, \ell_D\}$ governing each input dimension. The hyperparameters are initialised to the variance of $\{y_i\}$ and $\sigma_0^2/4$, and half the ranges of the input dimensions for the lengthscales (Quiñonero-Candela et al., 2010). One hundred sets of $m \times D$ spectral points are drawn from Eq. 2.34 and the log marginal likelihood is evaluated for each set. The set corresponding to the highest log marginal likelihood value are kept. No further optimisation is performed on the spectral points.

2.3 Disentangling Sources of Uncertainty

PILCO evaluates and minimises the expected return J^π by cascading uncertain inputs through the GP dynamics model to create long term predictions of the state evolution $p(\mathbf{x}_1|\pi), \ldots, p(\mathbf{x}_T|\pi)$. This gives insight into the variety of states that could be visited under a given policy, enabling the algorithm to quantify the quality of the controller and select its successor. However, this approach only considers the total uncertainty quantified by the model's predictive distribution, when in fact there are multiple sources of uncertainty present in the model. Here, the sources of uncertainty in the model are identified and disentangled into their aleatoric and epistemic constituents.

The one-step predictions used to cascade uncertain inputs are generated by the GP predictive distribution. In Section 2.2 this was approximated by the trigonometric model predictive distribution, which is reproduced here with shortened notation,

$$p(\Delta_*|\tilde{\mathbf{x}}_*) = \int p(\Delta_*|\tilde{\mathbf{x}}_*, \mathbf{w})q(\mathbf{w})\mathrm{d}\mathbf{w} \tag{2.36}$$

where $p(\Delta_*|\tilde{\mathbf{x}}_*, \mathbf{w})$ is the model likelihood and $q(\mathbf{w})$ is the posterior distribution over the model weights. The sources of uncertainty or randomness on Δ_* are the model weights $\mathbf{w} \sim q(\mathbf{w})$, the prior $a_r, b_r \sim \mathcal{N}\left(0, \sigma_0^2/m\right)$ and the additive noise $\varepsilon \sim \mathcal{N}\left(0, \sigma_n^2\right)$ (Depeweg et al., 2017). Indeed, these are the same sources of uncertainty present in PILCO's GP dynamics model and are analogous to the posterior and prior distributions over functions and the additive noise. This means that there are two types of uncertainties entangled in the predictions of Δ_*; *aleatoric* and *epistemic* (Der Kiureghian and Ditlevsen, 2009)(Kendall and Gal, 2017). The aleatory arises from the randomness present in a_r, b_r and ε and is irreducible. In contrast, the epistemic uncertainty represents uncertainty associated with the model weights $\mathbf{w}$ which reduces as more data is collected, resulting in the contraction of the posterior $q(\mathbf{w})$.

PILCO is a direct policy search method and is formulated so that the cost function is consulted directly when minimising J^π in Eq. 2.2, therefore, only uncertainty in the transition function that directly influences uncertainty on the cost $\mathbb{C}(\mathbf{x})$ is of concern. Another way to view this is that the model's representation of the state-action space can be exceedingly large for complex environments and exploring regions in state space that are irrelevant for the task of learning control is a waste of both time and memory resources . Ultimately, one is only interested in uncertainty that is associated with trajectories through the space that yield low cost solutions. In the following treatment, the cost is specified as $\mathbb{C}^\pi(\mathbf{x}_t)$ to denote the cost of state $\mathbf{x}$ and time t under policy π.

There are a number of metrics that can be used to represent uncertainty in the transition function. In this work, variance of the cost is used as the primary agent. First, the variance of the cost is expressed of the aleatoric uncertainty where all sources of aleatory are grouped into the term ε

$$\mathbb{V}_\varepsilon\left(\mathbb{C}^\pi\left(\mathbf{x}_t\right)\right) = \mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)^2\right] - \mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]^2. \tag{2.37}$$

In order to make use of the law of total expectation in the next step, the cost is conditioned on the target state $\mathbf{x}_{target}$. Here, the explicit conditioning has been omitted for notational convenience. Applying the law of total expectation with respect to the posterior distribution $q(\mathbf{w})$ over the model weights, which represents epistemic uncertainty, gives

$$\mathbb{V}\left(\mathbb{C}^\pi\left(\mathbf{x}_t\right)\right) = \mathbb{E}_{q(\mathbf{w})}\left[\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)^2\right]\right] - \mathbb{E}_{q(\mathbf{w})}\left[\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]\right]^2 \tag{2.38}$$

where the introduction of the expectation with respect to the posterior distribution has changed the left hand side of the equation so that it now represents total uncertainty. Noting that $\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)^2\right]$ can be replaced by the sum of the variance and the first moment gives

$$\mathbb{V}\left(\mathbb{C}^\pi\left(\mathbf{x}_t\right)\right)=\mathbb{E}_{q(\mathbf{w})}\left[\mathbb{V}_\varepsilon\left(\mathbb{C}^\pi(\mathbf{x}_t)\right)+\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]^2\right]-\mathbb{E}_{q(\mathbf{w})}\left[\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]\right]^2. \tag{2.39}$$

The expectation of the sum is also the sum of the expectation so separating the first term and regrouping the last terms together gives

$$\mathbb{V}\left(\mathbb{C}^\pi\left(\mathbf{x}_t\right)\right)=\mathbb{E}_{q(\mathbf{w})}\left[\mathbb{V}_\varepsilon\left(\mathbb{C}^\pi(\mathbf{x}_t)\right)\right]+\left(\mathbb{E}_{q(\mathbf{w})}\left[\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]^2\right]-\mathbb{E}_{q(\mathbf{w})}\left[\mathbb{E}_\varepsilon\left[\mathbb{C}^\pi(\mathbf{x}_t)\right]\right]^2\right). \tag{2.40}$$

Noting that the parenthesised term is the variance with respect to the posterior distribution in terms of the first and second moments gives the final uncertainty decomposition

$$\underbrace{\mathbb{V}\left(\mathbb{C}^\pi(\mathbf{x}_t)\right)}_{\textcircled{i}}=\underbrace{\mathbb{E}_{q(\mathbf{w})}[\mathbb{V}_\varepsilon(\mathbb{C}^\pi(\mathbf{x}_t))]}_{\textcircled{ii}}+\underbrace{\mathbb{V}_{q(\mathbf{w})}(\mathbb{E}_\varepsilon[\mathbb{C}^\pi(\mathbf{x}_t)])}_{\textcircled{iii}} \tag{2.41}$$

where $\textcircled{i}$ is the total uncertainty, $\textcircled{ii}$ is aleatoric uncertainty and $\textcircled{iii}$ is the epistemic uncertainty. All uncertainties are for state $\mathbf{x}$ at time t under policy π.

2.4 Gold-Standard Monte-Carlo Uncertainty Estimates

A fundamental problem with PILCO's policy evaluation method is that it doesn't acknowledge that in practice there is only one true underlying dynamics and it is just not aware of the correct one. Consequently, it repeatedly entangles all possible dynamics at each time point when predicting the state evolution. Estimating Eq. 2.41 by Monte-Carlo sampling solves this issue because the dynamics be can sampled, the state propagated forwards and then the dynamics sampled again. This allows the averaging required for the computation of Eq. 2.41 to be computed in the right order.

Thinking about Eq. 2.41 in terms of Monte-Carlo sampling can also provide good intuition into how this decomposition disentangles aleatoric and epistemic uncertainties in the cost under policy π when considering the evolution of trajectories through PILCO's state-action representation. Starting with the term representing aleatory $\textcircled{ii}$. Consider a single set of weights sampled from the posterior distribution $\mathbf{w} \sim q(\mathbf{w})$ and using them according to Eq. 2.25 to perform a one-step prediction of N trajectories through the transition function under policy π. Once the set of weights is drawn and held fixed, the one-step prediction is deterministic save for the additive Gaussian noise $\boldsymbol{\varepsilon}$. The variance of the N one-step predictions is now taken giving the empirical estimate of the aleatoric uncertainty for a single transition function drawn from $q(\mathbf{w})$. This process is then repeated M times and the empirical average taken is over the M functions drawn from the posterior. So in effect, one is computing the average variance of transitions under the influence of noise alone. Provided M and N are sufficiently large, what has been described above is an unbiased estimate of the total aleatoric uncertainty in the cost for a single state transition under policy π. The aleatoric uncertainty over the course of T steps through the transition function is then the sum of aleatoric uncertainties from each step.

Similarly, for the epistemic uncertainty term $\textcircled{\textit{iii}}$. Each set of weights $\mathbf{w}$ drawn from the posterior distribution and held fixed provides a deterministic one-step prediction (Eq. 2.25) for N trajectories save for the additive Gaussian noise $\boldsymbol{\varepsilon}$. For N trajectories and provided that N is sufficiently large, the expectation for a single one-step prediction is the mean of the function sampled from the posterior distribution because the mean of the Gaussian noise is zero. This expectation has removed the aleatoric uncertainty. The variance is then taken over M posterior transition functions and provided M and N are sufficiently large then this is an unbiased estimate of the epistemic uncertainty present in the cost for a single transition under policy π. The epistemic uncertainty over the course of T steps through the transition function is then the sum of epistemic uncertainties from each step. The total uncertainty present in the cost under policy π is then the sum of the aleatoric and epistemic uncertainties.

Formalising these steps gives a "gold-standard" Monte-Carlo scheme for estimating the aleatoric and epistemic uncertainties present in the cost under policy π according to Eq. 2.41. The Monte-Carlo estimates can be generated by initialising $M \times N$ trajectories, each with start state $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$, which are propagated through the approximate GP dynamics model T times under policy π. For this, M sets of weights are drawn from the posterior distribution $q(\mathbf{w})$ and for each set N roll-outs of T steps are performed with fixed $\mathbf{w}$ where Gaussian additive noise is sampled for each transition. A deterministic control action $\mathbf{u}$ is computed for the state at each time t as required by the dynamics model and the one-step prediction for each transition is computed with Eq. 2.25. At each time step the cost is calculated (Eq. 2.3) and stored to give an array of cost values $\mathbb{C}^{\pi}(\mathbf{x}_t)$ of shape (M,N) for each time t. Equation 2.41 can then be estimated with

$$\mathbb{V}(\mathbb{C}^\pi(\mathbf{x})) = \sum_{t=0}^{T-1} \left\{ \mathbb{V}_N\left(\frac{\mathbb{C}^\pi(\mathbf{x}_t)\mathbf{1}}{N}\right) + \frac{\mathbf{1}^\top \mathbb{V}_M\left(\mathbb{C}^\pi(\mathbf{x}_t)\right)}{M} \right\} \tag{2.42}$$

where $\mathbf{1}$ is a column vector of ones and $\mathbb{V}_N$ and $\mathbb{V}_M$ are the empirical variance of the (M,N) cost values $\mathbb{C}^\pi(\mathbf{x}_t)$ over the rows and columns, respectively.

The detailed steps of the procedure are shown in Algorithm 2. In the Algorithm 2 it may seem as if the use of multiple for loops could lead to slower performance than perhaps drawing N start states and collectively propagating them through the dynamics model. However, in step 9 a for loop is still required to draw N control actions from PILCO, although there may in fact be a way to compute multiple control actions simultaneously. Indeed, the main reason for using 3 for loops is that the only loop that is dependent on calculations from the previous iteration is the T loop. Hence, the other two loops can be parallelised for optimal performance.

Algorithm 2 Gold Standard Monte-Carlo Uncertainty Estimate

```
1:  init: initialise Costs array: $\mathbb{C} \leftarrow nan(M,N,T)$
2:  Learn approximate GP dynamics model using all environmental data    ▷ Sec 2.2
3:  for $m = 0$ to $M-1$ do:
4:      Sample a set of weights: $\mathbf{w} \sim q(\mathbf{w})$    ▷ Eq. 2.18
5:      for $n = 0$ to $N-1$ do:
6:          Sample start state: $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}_0|\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0)$
7:          for $t = 0$ to $T-1$ do:
8:              Store the cost of the current state: $C[m,n,t] \leftarrow \mathbb{C}(\mathbf{x}_t)$    ▷ Eq. 2.3
9:              Get state-action representation: $\tilde{\mathbf{x}}_t = (\mathbf{x}_t,\mathbf{u}),\quad \mathbf{u} = \pi(\mathbf{x}_t,\boldsymbol{\theta})$
10:             Do posterior one-step: $\mathbf{x}_{t+1} = \mathbf{x}_t + \Delta_t + \varepsilon,\quad \varepsilon \sim \mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_n)$    ▷ Eq. 2.25
11: Compute aleatoric uncertainty: $\sum_{t=0}^{T-1}\left\{\mathbb{V}_N\left(\frac{\mathbb{C}^\pi(\mathbf{x}_t)\mathbf{1}}{N}\right)\right\}$
12: Compute epistemic uncertainty: $\sum_{t=0}^{T-1}\left\{\frac{\mathbf{1}^\top\mathbb{V}_M(\mathbb{C}^\pi(\mathbf{x}_t))}{M}\right\}$    ▷ Eq. 2.42
```

Chapter 3

Results

This section presents the results of experiments conducted using the Monte Carlo uncertainty estimation methodology described above. Section 3.1 describes three PILCO environments on which the uncertainty estimates are tested. The results themselves are presented in Section 3.2 followed by a discussion in Section 3.3. Finally, the conclusions are summarised in Section 3.4.

3.1 Environments

Cart-Pole Swing Up

The simplest of the three environments used for experimentation, the *cart-pole swing up problem*, is shown in Fig. 3.1. It consists of a cart of mass m_1, as well as a pendulum of mass m_2 and length l that is free to swing about the pivot by which it is connected to the cart. The angle of the pendulum θ_2 is measured anti-clockwise from the downward position. Continuous control actions u applied to the cart cause it to move horizontally in the x-direction. The objective of the task is to apply horizontal force to the cart in such a way that the pendulum is swung to the upright vertical position and balanced there while positioning the cart in the centre of the system at $x = 0$.

The system has 4 continuous state variables: the position of the cart x, the velocity of the cart $\dot{x}$, the pendulum angle θ_2, and the angular velocity of the pendulum $\dot{\theta}_2$ of the pendulum. There is a target associated with each state variable.

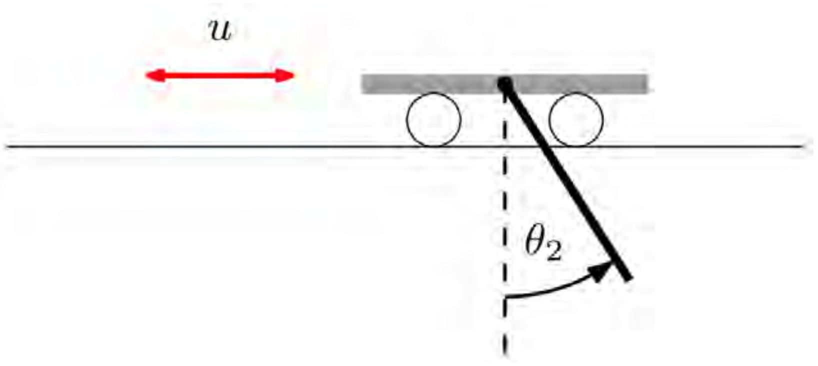


Fig. 3.1 Cart-pole swing up. Source: Deisenroth et al. (2013b).

Pendubot

The *Pendubot*, shown in Fig. 3.2, is the second most complex environment used for experimentation. It consists of an *inner* and an *outer* pendulum, with masses m_1 and m_2 and lengths l_1 and l_2, respectively. The two pendulums are linked together, with the inner pendulum also being attached to the ground. The inner and outer pendulum angles are given by θ_2 and θ_3, respectively, measured anticlockwise from the upright vertical position. The link between the ground and the inner pendulum can be acted upon by the agent by applying a torque u to the joint; however, the link between the pendulums cannot be acted upon. The objective of the system is to apply torque to the central joint in such a way as to swing both pendulums to the upright vertical position and balance them there.

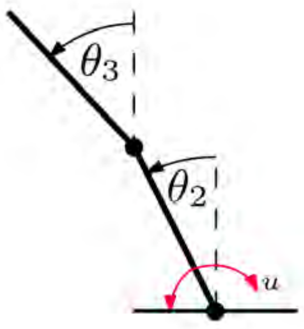


Fig. 3.2 Source: Deisenroth et al. (2013b).

The system has 4 continuous state variables: the angles of the pendulums θ_2 and θ_3 and the angular velocities of the pendulums $\dot{\theta}_2$ and $\dot{\theta}_3$. There is a target associated with each state variable.

Cart-Double-Pendulum

The *cart-double-pendulum* problem, shown in Fig. 3.3, is the most complex environment used for experimentation. It consists of a cart of mass m_1, as well as an inner and an outer pendulum of masses m_2 and m_3 and lengths l_2 and l_3, respectively. The two pendulums are linked together, with the inner pendulum also being attached to the cart. The central and outer pendulum angles are given by θ_2 and θ_3, respectively, and are measured anticlockwise from the upright vertical position. Continuous control actions u applied to the cart cause it to move horizontally in the x-direction. The objective of the task is to apply horizontal force to the cart in such a way that the double pendulum system is swung to the upright vertical position and balanced there while positioning the cart in the centre of the system at $x = 0$. The unconstrained double pendulum system exhibits rich dynamical behaviour and is a chaotic system.

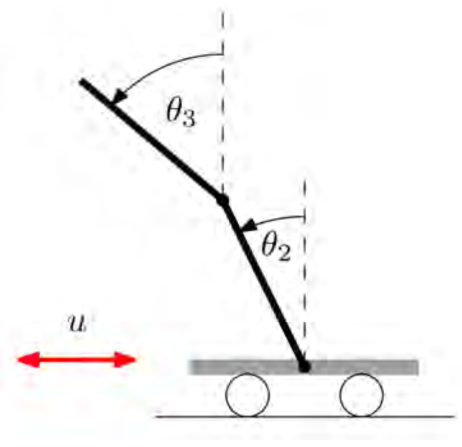


Fig. 3.3 Source Deisenroth et al. (2013b).

The system has 6 continuous state variables: the position of the cart x, the velocity of the cart $\dot{x}$, the angles of the pendulums θ_2 and θ_3 the angular velocities of the pendulums $\dot{\theta}_2$ and $\dot{\theta}_3$. There is a target associated with each state variable.

3.2 Experiments

3.2.1 The Effect of Cost Function Uncertainty on Learning

In this section, I present the Monte Carlo uncertainty analyses for the 3 PILCO environments introduced in section 3.1. The procedure to produce the results is as follows. After each PILCO episode, I train the trigonometric basis function model (see section 2.2) on all the observed data up to the end of that episode. The inputs are the state-action tuples and the targets are the state differences. Conditionally independent models are trained for each target dimension. In all cases, I use 500 basis functions. The Monte Carlo trajectory roll-outs are performed according to algorithm 2 under policy π for $(M = 100, N = 100, T = T_E)$, where T_E and the transition noise $\boldsymbol{\varepsilon}_E$ are environment specific. I then decompose the total uncertainty into uncertainties representing total aleatoric and total epistemic uncertainty in the cost $\mathbb{C}(\mathbf{x})$ under policy π for an entire episode, following the procedure detailed in section 2.3.

Cart-pole

Figures 3.4 and 3.5 show a random sample of the Monte Carlo trajectory roll-outs for the state variables of the cart-pole environment: cart position x and cart velocity $\dot{x}$ (fig. 3.4), and pendulum angular velocity $\dot{\theta}$ and pendulum angle θ (fig. 3.5). In both figures, the left vertical axis represents the state variable and corresponds to the trends for the Monte Carlo trajectories, the actual trajectory recorded during that episode, the target of the state variable, and PILCO's prediction of the state evolution before the episode. The right axis shows the percentage of Monte Carlo trajectories that fall within the 2 standard deviation error bars of PILCO's prediction for that time step. The trajectories shown are from episode 15, when PILCO has achieved the environment objective, and so represent trajectories under a mature policy π. For the cart-pole environment, a high agreement (greater than 80%) is noted between the Monte Carlo trajectories and PILCO's prediction of the state evolution.

Fig. 3.6 shows the average cost per episode and cost uncertainty decomposition under policy π over the course of learning. Panel 3.6a shows the full uncertainty decomposition and panel 3.6b shows the ratio of epistemic to total uncertainty. The average cost per episode, shown on the left vertical axes, is indicative of success in learning. High cost indicates that the agent is not achieving the objective, low cost indicates the objective is being achieved and a decreasing average cost indicates that the agent is learning. The right axis shows total cost uncertainty and its decomposition into aleatoric and epistemic cost uncertainty.

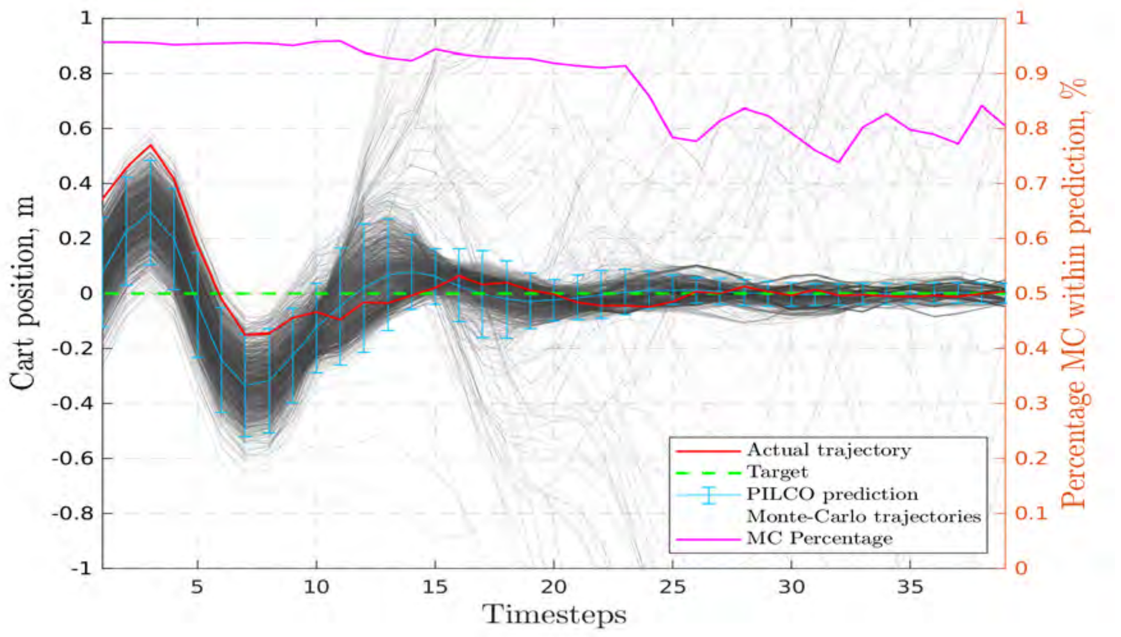
The average cost per episode exhibits a sharp decline early in the learning process (this can be seen in both panels of the figure) and by the third episode PILCO has solved the environment. This is the simplest of the environments tested and because PILCO solves it early in the learning process, we cannot gain much insight into how the cost uncertainty varies. In episode zero (not shown in the figures), PILCO performs a series of roll-outs under a random policy to generate initial state observations. It is likely, particularly in the case where the state-space is well explored during the random roll-out, that these data are enough for the algorithm to find a low cost policy that solves the environment early in the learning process. Consequently, for episodes 1-3 the selected policy contains little epistemic cost uncertainty because the model is already confident about its beliefs and a successful low cost policy exists.

Interestingly, for episodes 7 and 8, PILCO selects policies with high epistemic uncertainty in the cost and does so after it has already solved the environment. This could be due to PILCO's unintentional exploration mechanism where it favours uncertain states. After this (Fig. 3.6a episodes 9-15), the total and aleatoric uncertainties settle and the epistemic reduces, indicating that the model is now confident in its model of parts of the state-space associated with low cost.

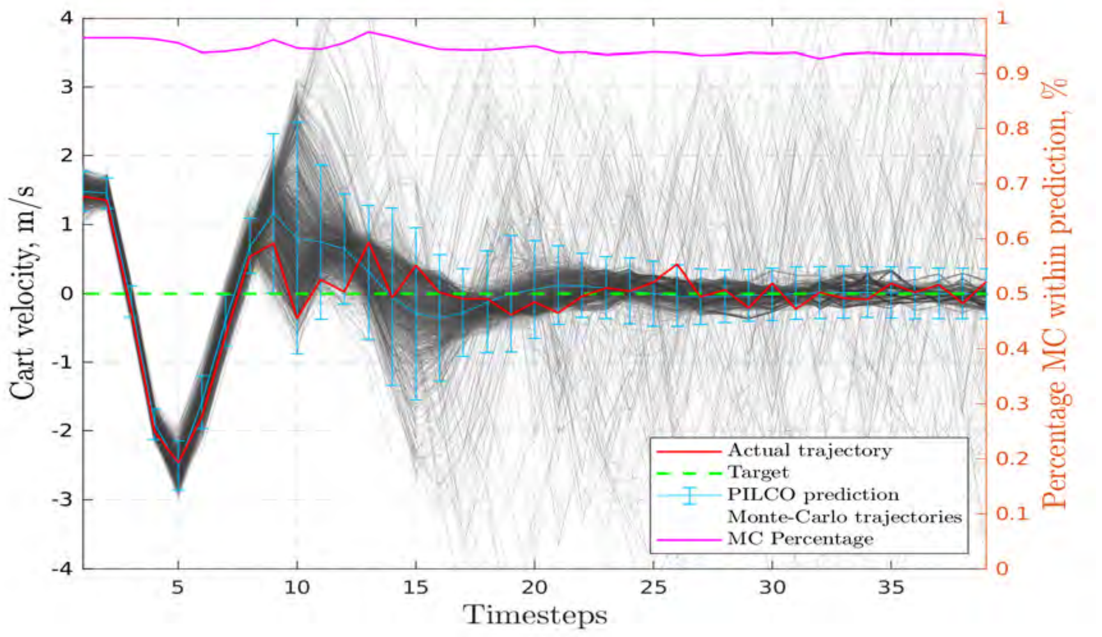
Pendubot

Fig. 3.7 and 3.8 show a random sample of the Monte Carlo trajectory roll-outs for the state variables of the pendubot environment: pendulum angular velocities $\dot{\theta}_2$ and $\dot{\theta}_3$ (fig. 3.7), and pendulum angles θ_2 and θ_3 (fig. 3.8). In both figures, the left vertical axis represents the state variable and corresponds to the trends for the Monte Carlo trajectories, the actual trajectory recorded during that episode, the target of the state variable, and PILCO's prediction of the state evolution before the episode. The right axis shows the percentage of Monte Carlo trajectories that fall within the 2 standard deviation error bars of PILCO's prediction for that time step. The trajectories shown in the figures are from episode 20, when PILCO has achieved the environment objective, and so once again represent trajectories under a mature policy π.

Up to approximately 20 time steps there is cohesion in the Monte Carlo trajectories, corresponding to the period where the pendulums are being manoeuvred to the upright position. After this the trajectories spread out, corresponding to the period where the agent attempts to balance the pendulums in the upright position. This indicates that there is more uncertainty in the policy during the balancing phase than during the swing-up phase.

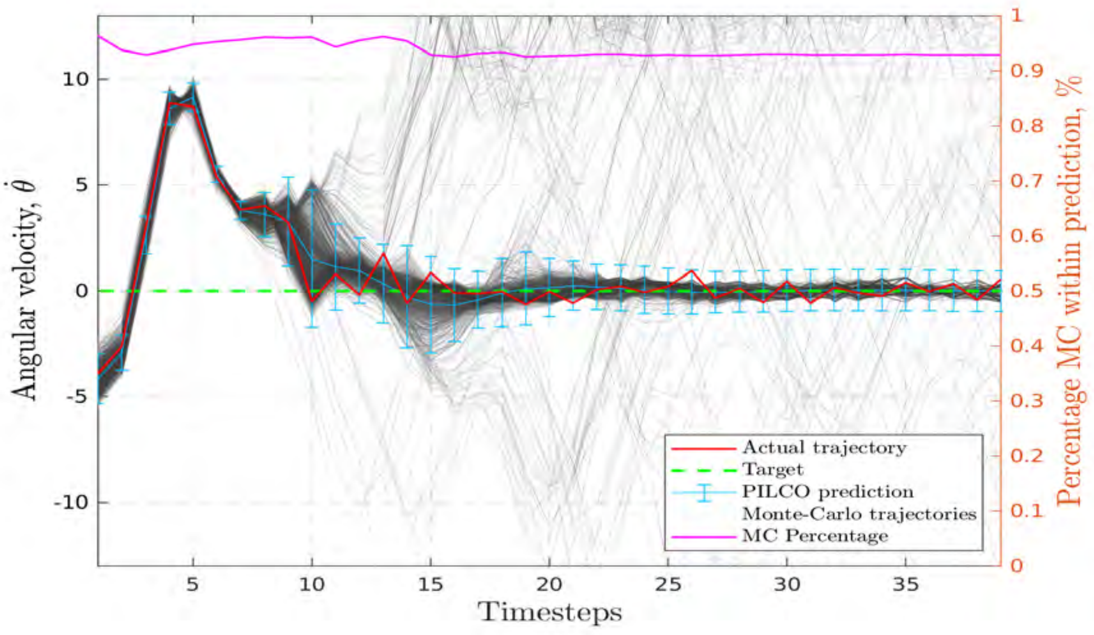


(a) Cart position x (m)

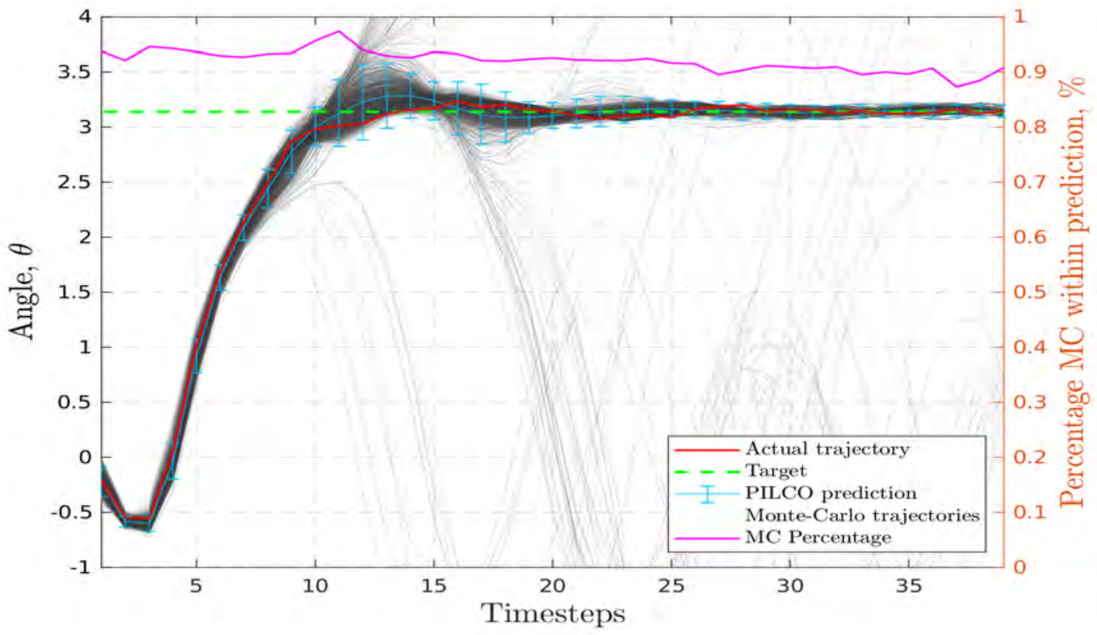


(b) Cart velocity $\dot{x}$ (m/s)

Fig. 3.4 Monte Carlo roll-outs for the **cart-pole** environment for cart position and cart velocity. Figures show a random sample of the Monte Carlo trajectories for episode 15. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.1s, roll-out horizon 4.0s

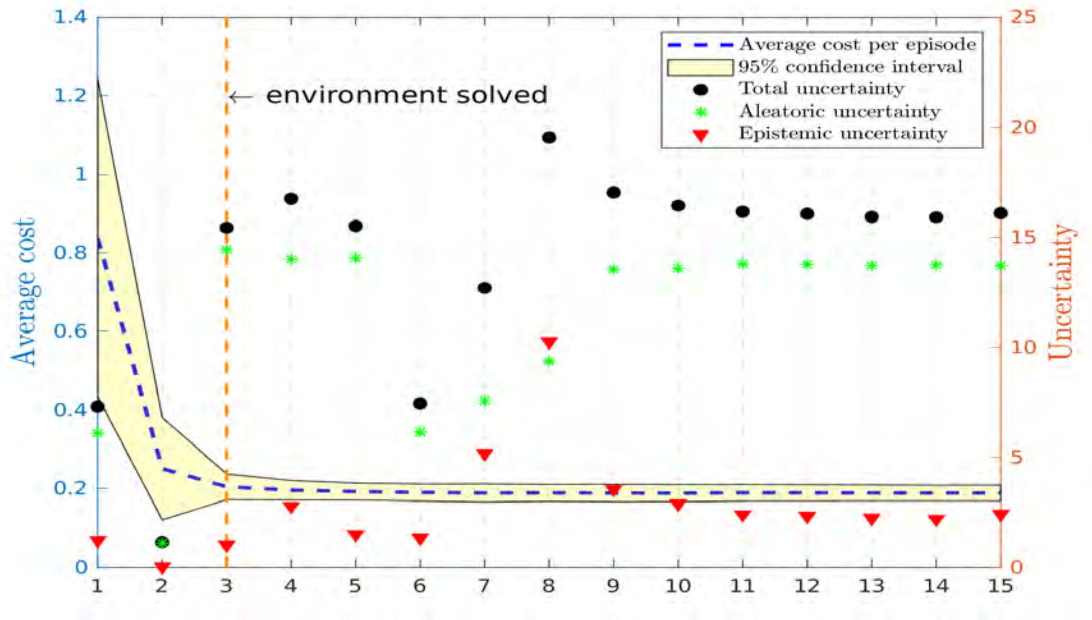


(a) Pendulum angular velocity $\dot{\theta}$ (rad/s)

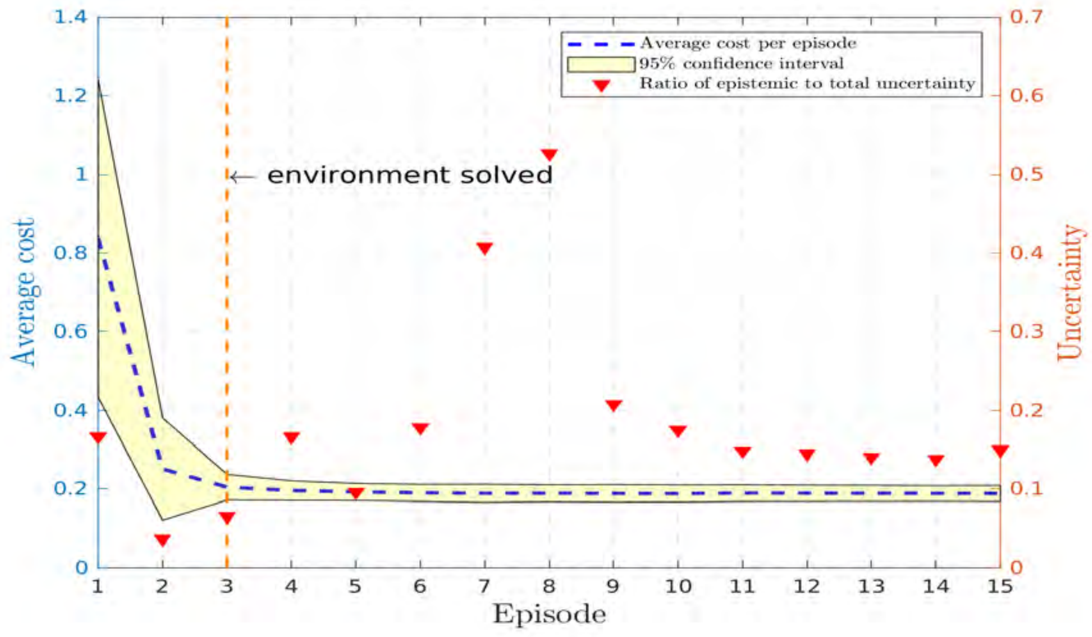


(b) Pendulum angle θ (rad)

Fig. 3.5 Monte Carlo roll-outs for the **cart-pole** environment for pendulum angular velocity and pendulum angle. Figures show a random sample of the Monte Carlo trajectories for episode 15. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.1s, roll-out horizon 4.0s



(a) Average cost (left axis) and uncertainty decomposition (right axis) for each episode over the course of learning.



(b) Average cost (left axis) and ratio of epistemic to total uncertainty (right axis) for each episode over the course of learning.

Fig. 3.6 **Cart-pole**: Average cost (left axis) and uncertainty (right axis) for each episode over the course of learning.

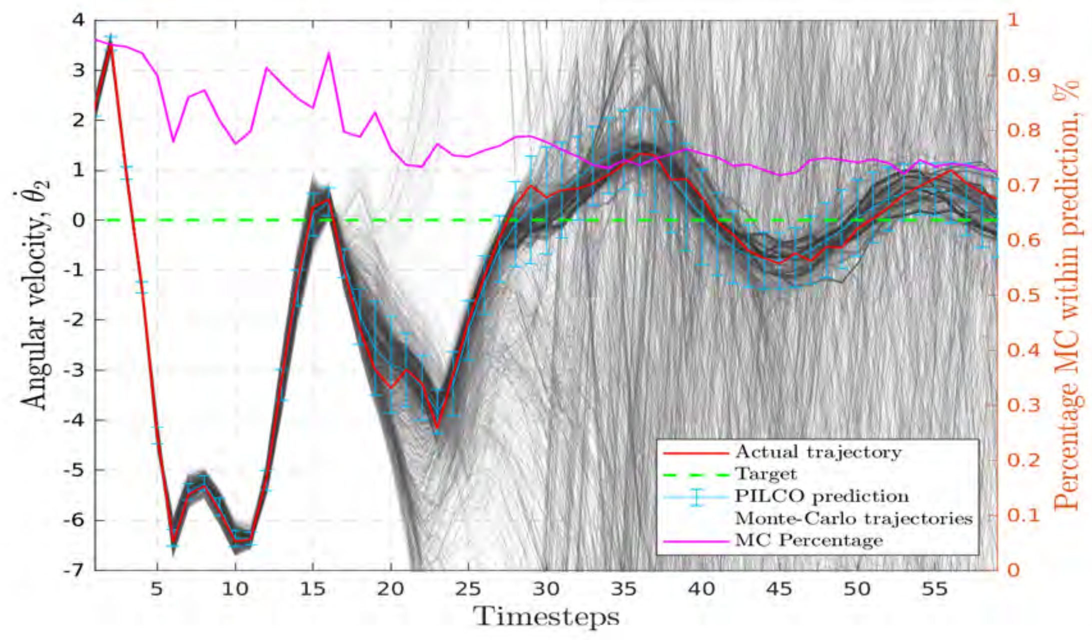
Although there is a large amount of spread in the trajectories, approximately 70% are still within PILCO's 2 standard deviation prediction.

Fig. 3.9 shows the average cost per episode and cost uncertainty decomposition under policy π over the course of learning. Once again, panel 3.9a shows the full uncertainty decomposition and panel 3.9b shows the ratio of epistemic to total uncertainty. As can be seen in panel 3.9a, over the first five episodes the agent learns to swing the pendulums to the upright position but cannot keep them balanced. During this phase, the average cost steps down from its initial value and consequently the total and aleatoric uncertainties in the cost increase while the epistemic cost uncertainty remains low. The increase in total cost uncertainty is due to the agent forming an initial belief about the environment (with no belief there is no uncertainty). Over the course of the next 10 episodes, the agent learns to maintain the balance of the pendulums. Again a step down in average cost is observed and this time an increase in aleatoric, epistemic and total cost uncertainty is observed. Once the environment is solved, the aleatoric cost uncertainty stabilises and the epistemic cost uncertainty reduces. From this representation, it is difficult to see the how the breakdown of cost uncertainties relate to learning.

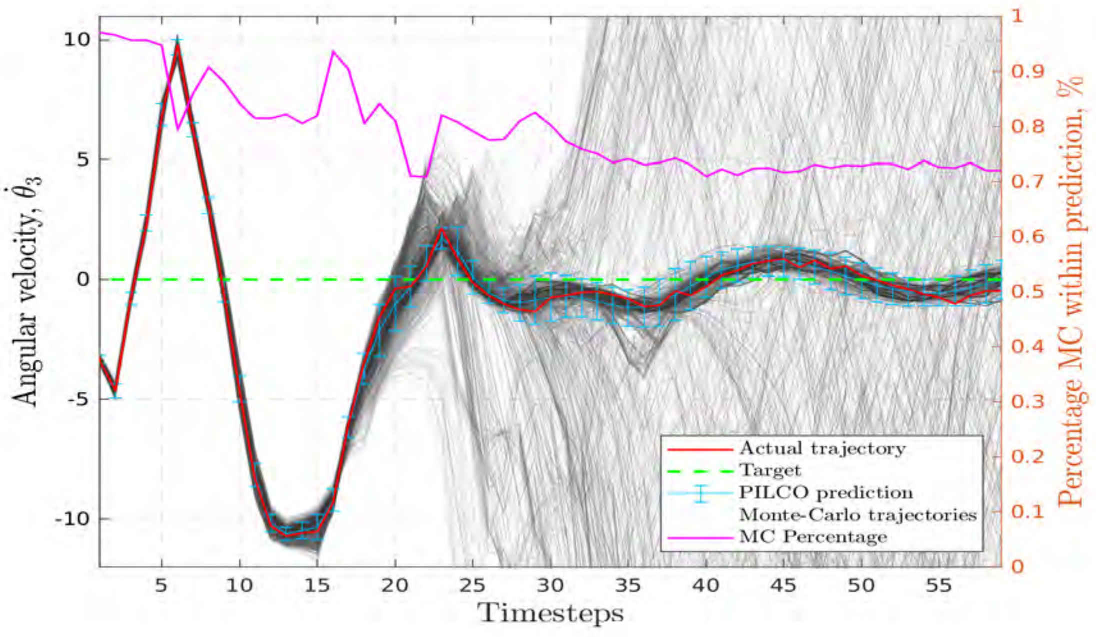
In panel 3.9b, the cost uncertainty is represented as the ratio of epistemic to total uncertainty. The first three episodes each yield an average cost of approximately 0.95 indicating that little learning is occurring over this period. During these episodes, PILCO has selected policies that have a low ratio of epistemic to total cost uncertainty (see right vertical axes). From episodes 3 to 4, the average cost reduces from 0.95 to 0.83 and this step in learning corresponds to a policy which has a higher ratio of epistemic to total cost uncertainty than the previous episodes. This trend repeats for episodes 5, 6 and 7 where the average cost plateaus for 5 and 6, corresponding to policies with a low ratio of epistemic to total cost uncertainty, but reduces for episode 7, corresponding to a policy with a high ratio of epistemic to total cost uncertainty. Perhaps the most convincing demonstration of this correlation occurs between episodes 7 and 8 where the largest observed step decrease in average cost occurs and corresponds to a policy with the largest ratio of epistemic to total cost uncertainty. Finally, after the environment is solved around episode 8, the ratio of epistemic to total cost uncertainty drops off as the agent refines the policy and model confidence increases.

Cart-Double-Pendulum

Figures 3.10 to 3.12 show a random sample of the Monte Carlo trajectory roll-outs for the state variables of the cart-double-pendulum environment: cart position x and velocity $\dot{x}$ (fig.

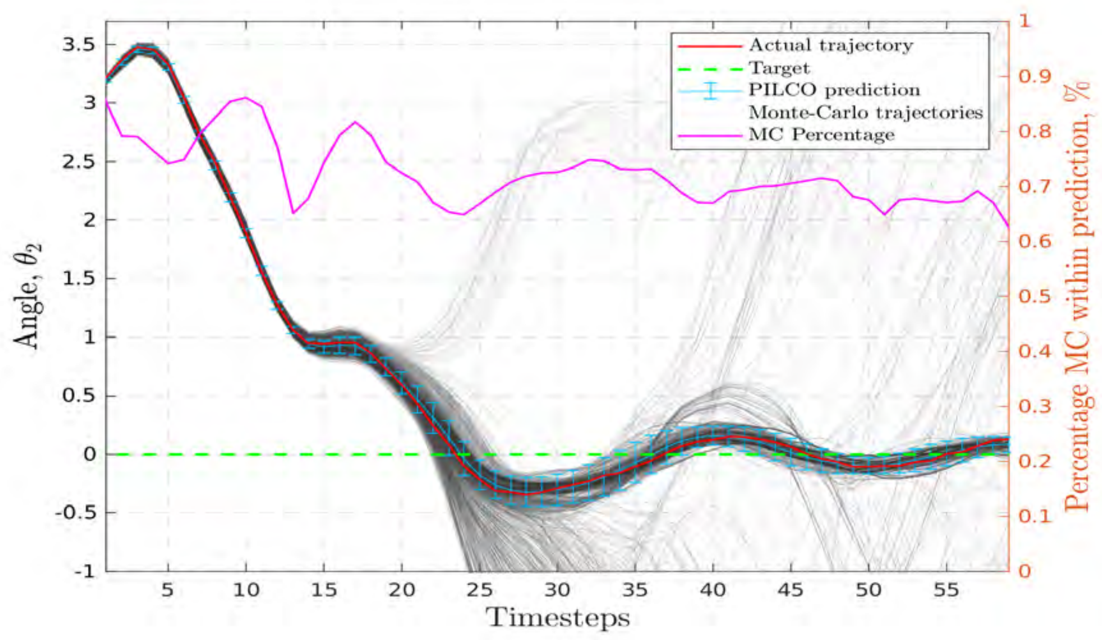


(a) Pendulum angular velocity $\dot{\theta}_2$ (rad/s)

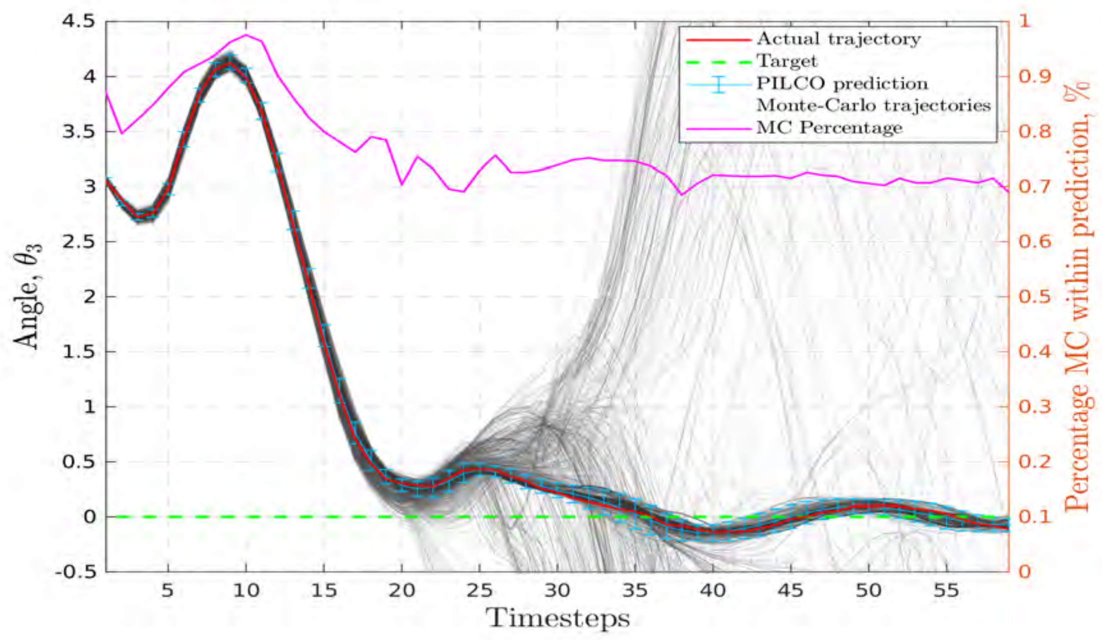


(b) Pendulum angular velocity $\dot{\theta}_3$ (rad/s)

Fig. 3.7 Monte Carlo roll-outs for the **pendubot** environment for pendulum angular velocities. Figures show a random sample of the Monte Carlo trajectories for episode 20. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.05s, roll-out horizon 3.0s.

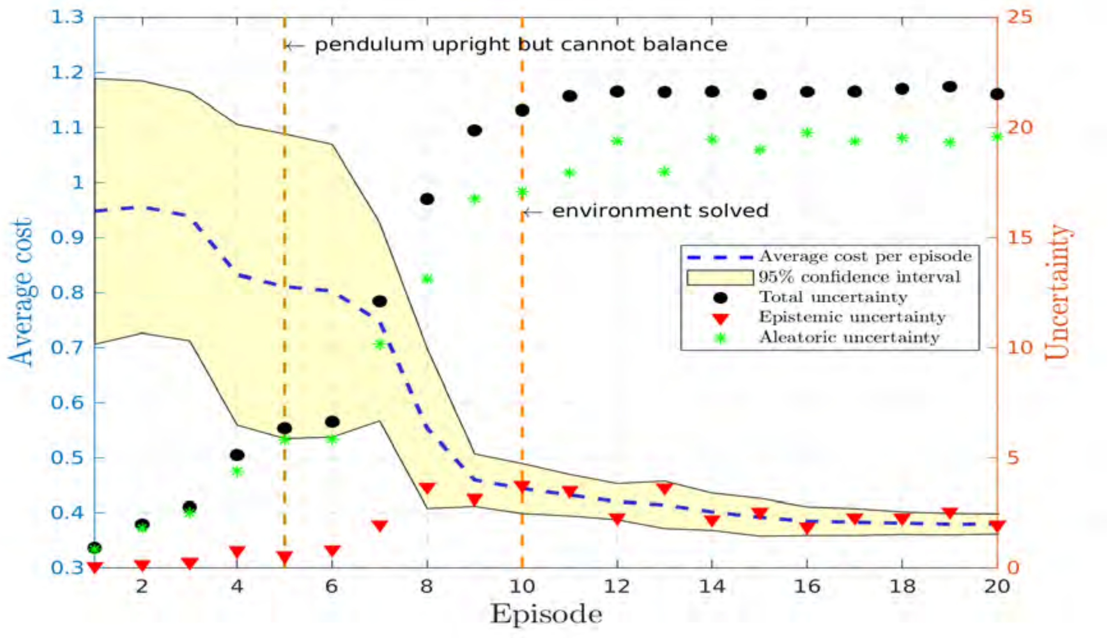


(a) Pendulum angle θ_2 (rad)

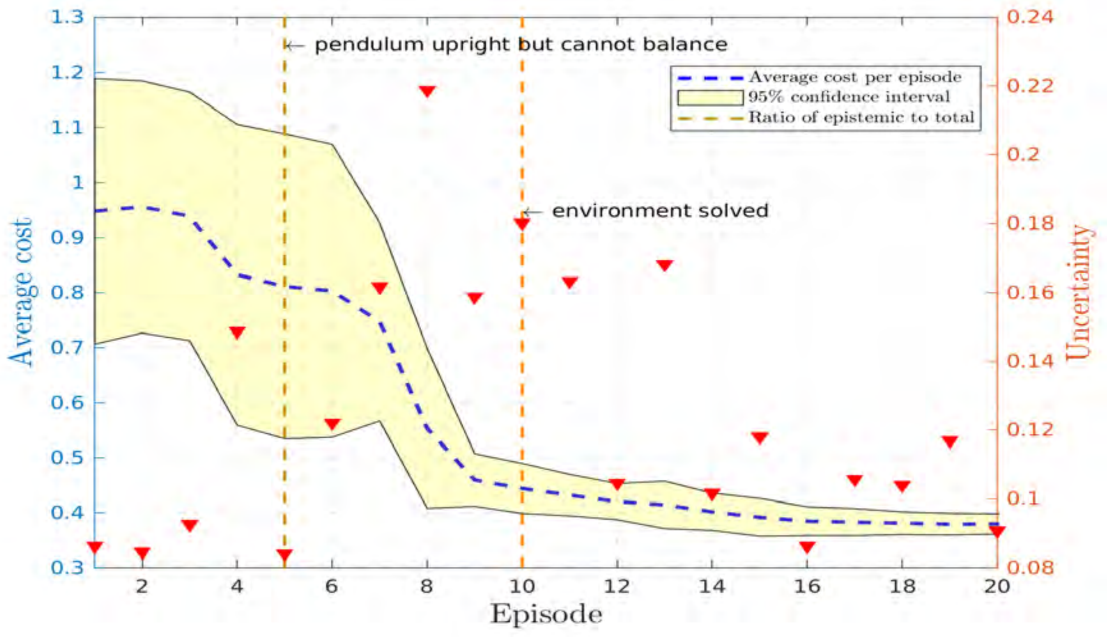


(b) Pendulum angle θ_3 (rad)

Fig. 3.8 Monte Carlo roll-outs for the **pendubot** environment for pendulum angles. Figures show a random sample of the Monte Carlo trajectories for episode 20. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.05s, roll-out horizon 3.0s.



(a) Average cost (left axis) and uncertainty decomposition (right axis) for each episode over the course of learning.



(b) Average cost (left axis) and ratio of epistemic to total uncertainty (right axis) for each episode over the course of learning.

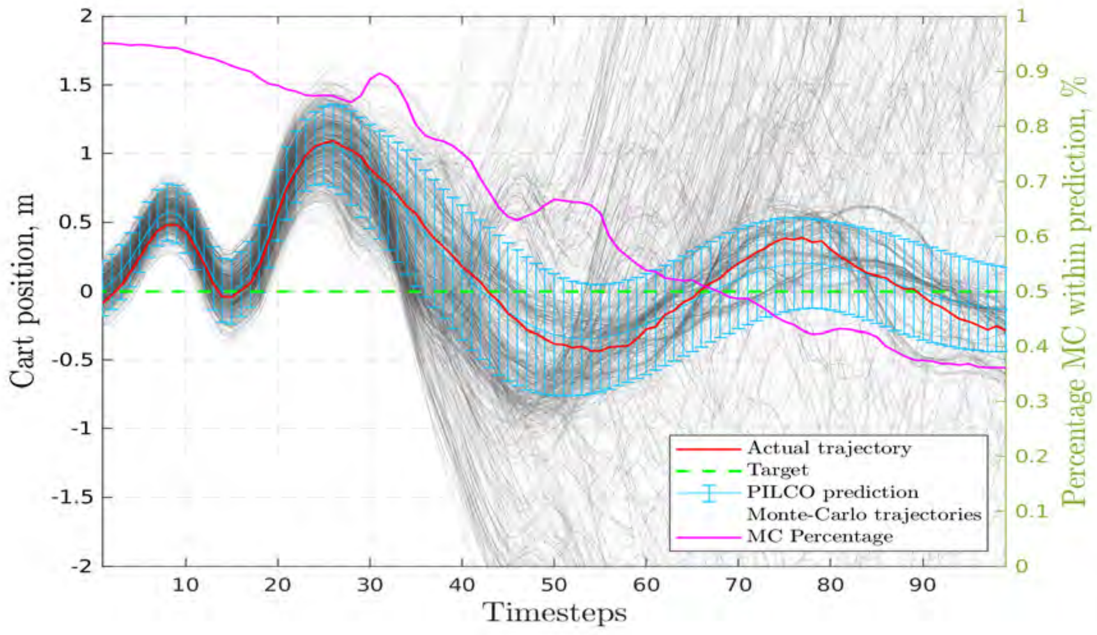
Fig. 3.9 **Pendubot**: Average cost (left axis) and uncertainty (right axis) for each episode over the course of learning.

3.10); pendulum angular velocities $\dot{\theta}_2$ and $\dot{\theta}_3$ (fig. 3.11); and pendulum angles θ_2 and θ_3 (fig. 3.12). In all three figures, the left vertical axis represents the state variable and corresponds to the trends for the Monte Carlo trajectories, the actual trajectory recorded during that episode, the target of the state variable, and PILCO's prediction of the state evolution before the episode. The right axis shows the percentage of Monte Carlo trajectories that fall within the 2 standard deviation error bars of PILCO's prediction for that time step.

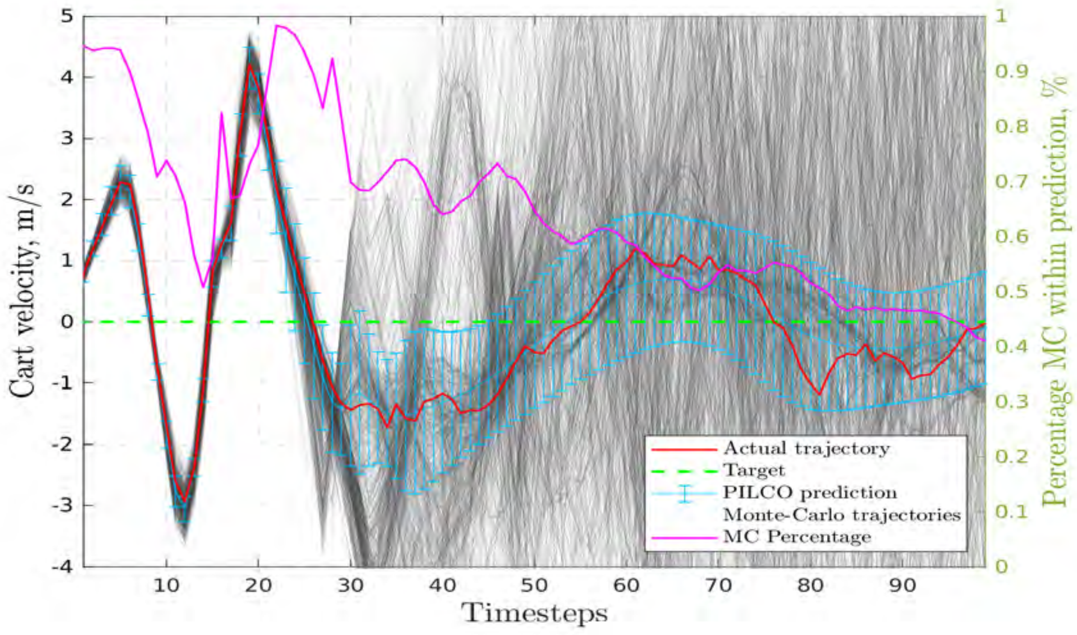
The trajectories shown in the figures are from episode 80, when PILCO has achieved the environment objective, and so represent trajectories under a mature policy π. Up to approximately 30 time steps there is cohesion in the Monte Carlo trajectories which again corresponds to the swing-up phase. During the balance phase, the trajectories spread out. By the end of the episode, only about 40% of the trajectories fall within PILCO's predictions.

Fig. 3.13 shows the average cost per episode and cost uncertainty decomposition under policy π over the course of learning. Panel 3.13a shows the full uncertainty decomposition and panel 3.13b shows the ratio of epistemic to total uncertainty. As can be seen in panel 3.13b, after 10 episodes the agent has learned to swing the pendulums into the upright position but cannot yet maintain them in a balanced state. During this phase there is a small reduction in average cost. The agent then takes a further 33 episodes to learn to maintain the pendulums in a balanced but off-centre position. During this second phase of learning, the total and epistemic cost uncertainties increase as the agent forms a belief about the environment; however, policies associated with low epistemic cost uncertainty are consistently being selected. Consequently, the learning gradient is gentle. Just after 40 episodes, there is a rapid increase in learning. During this phase, the total cost uncertainty rises further and departs from the trend of the aleatoric cost uncertainty due to the increase in epistemic cost uncertainty.

As shown in panel 3.13b, there is an even more pronounced trend in the ratio of total to epistemic cost uncertainty. Approaching the 40 episode mark, PILCO selects policies with an increasing ratio of total to epistemic cost uncertainty and as a consequence learning increases. Here there is a clear correlation between the amount of epistemic cost uncertainty present in the policy and the average cost. This shows that policies with a high epistemic cost uncertainty are associated with a steeper learning gradient.

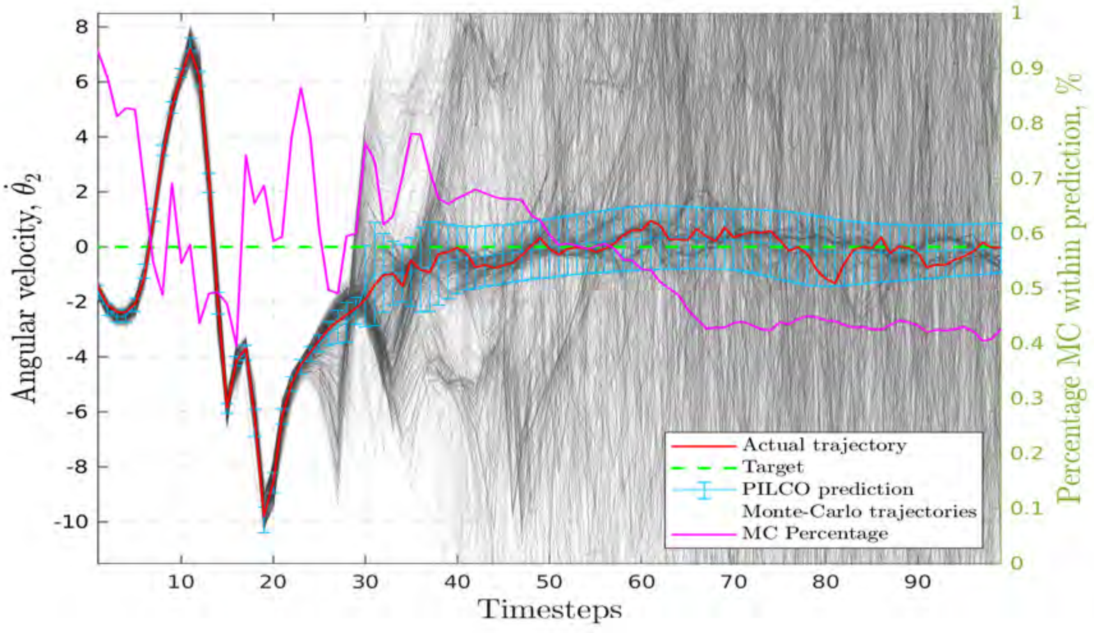


(a) Cart position x_1 (m)

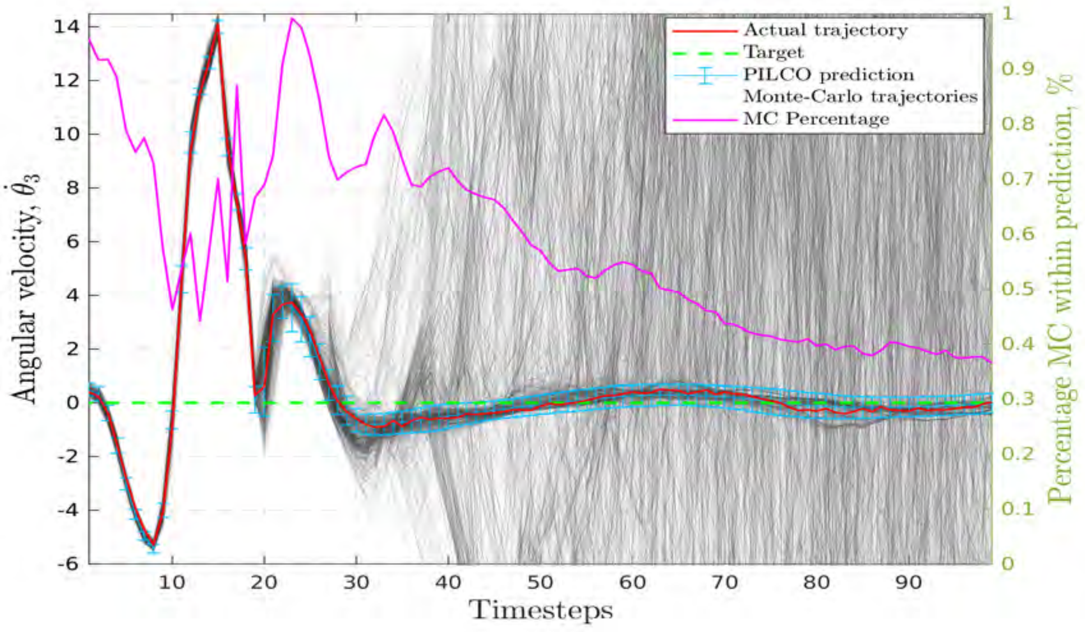


(b) Cart velocity $\dot{x}_1$ (m/s)

Fig. 3.10 Monte Carlo roll-outs for the **cart-double-pendulum** environment for cart position and cart velocity. Figures show a random sample of the Monte Carlo trajectories for episode 75. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.05s, roll-out horizon 5.0s.

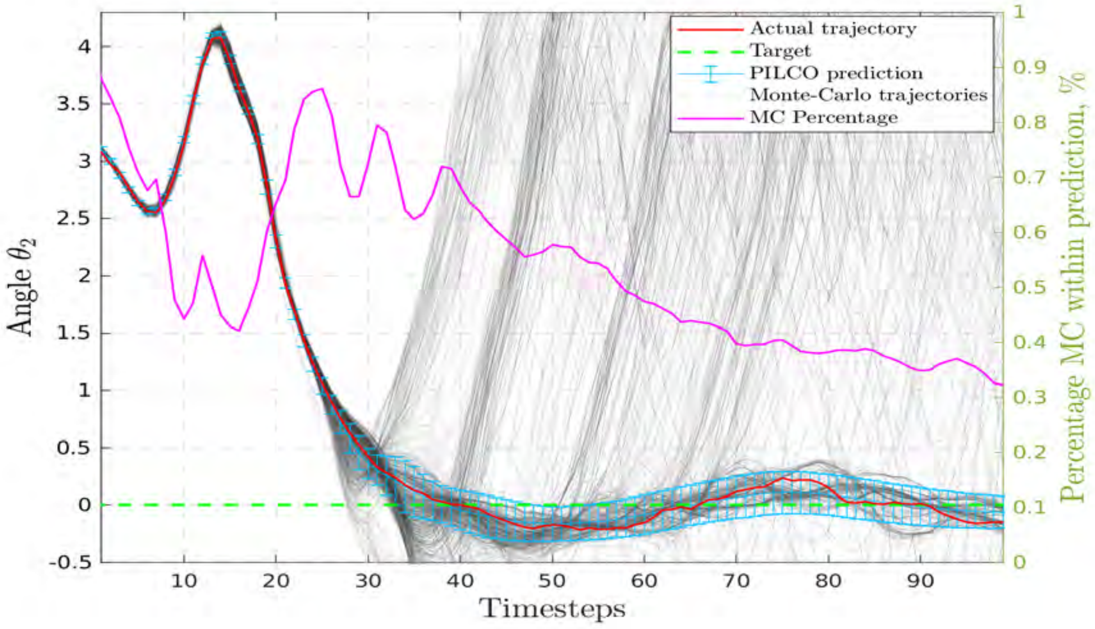


(a) Pendulum angular velocity $\dot{\theta}_2$ (rad/s)

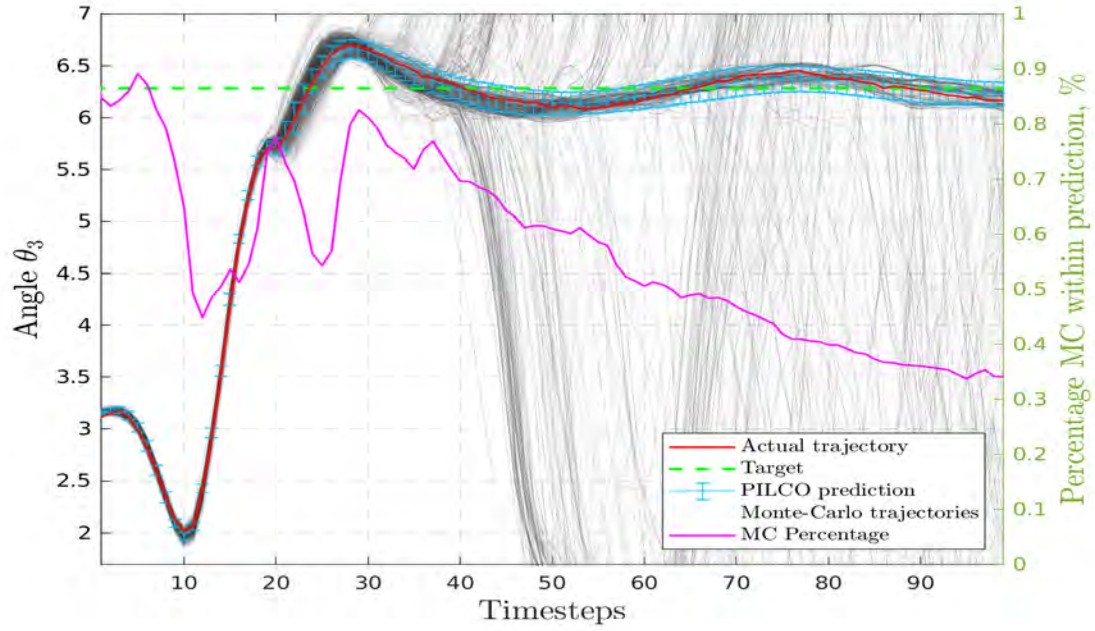


(b) Pendulum angular velocity $\dot{\theta}_3$ (rad/s)

Fig. 3.11 Monte Carlo roll-outs for the **cart-double-pendulum** environment for pendulum angular velocities]. Figures show a random sample of the Monte Carlo trajectories for episode 75. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.05s, roll-out horizon 5.0s.

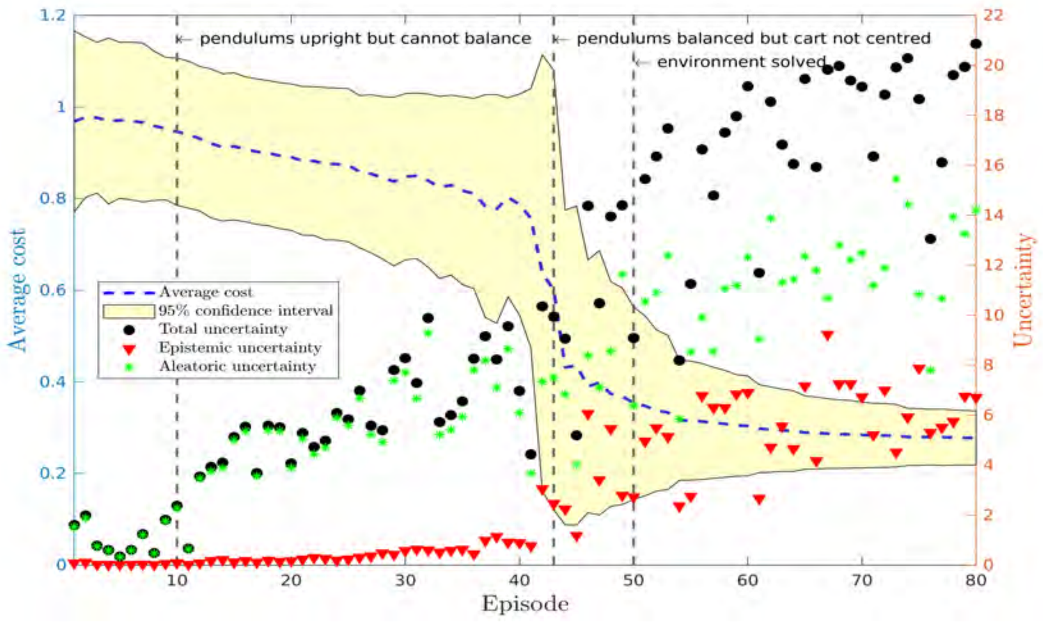


(a) Pendulum angle θ_2 (rad)

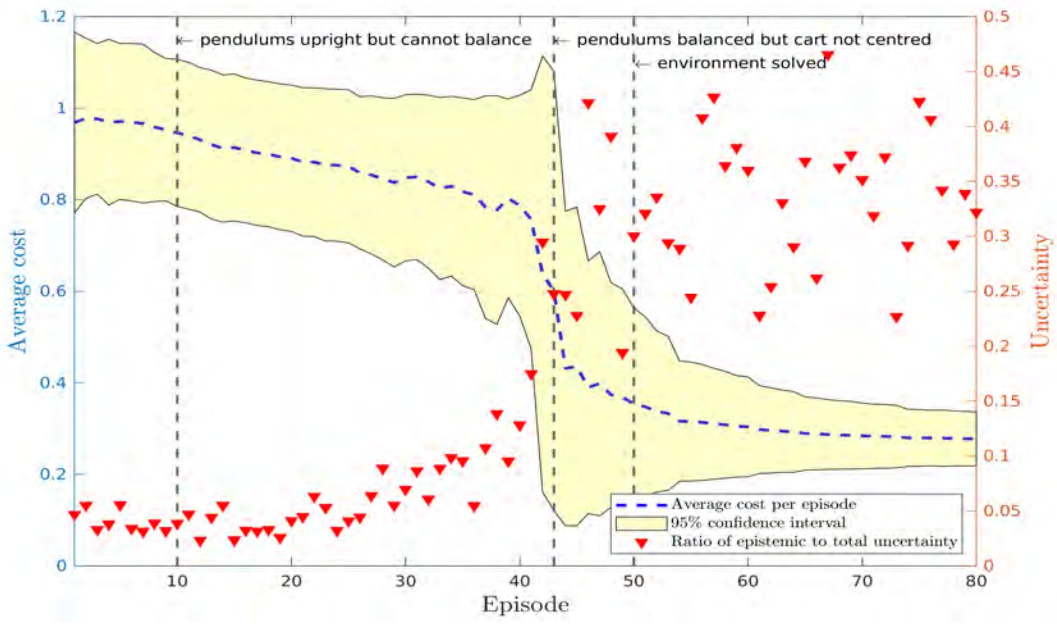


(b) Pendulum angle θ_3 (rad)

Fig. 3.12 Monte Carlo roll-outs for the **cart-double-pendulum** environment for pendulum angles]. Figures show a random sample of the Monte Carlo trajectories for episode 75. The first three legend items (actual trajectory, target, PILCO prediction and MC trajectories) correspond to the state variable axis (left axis) and the magenta line shows the percentage of MC trajectories inside PILCO's prediction (2 standard deviations) (right axis). Sample time 0.05s, roll-out horizon 5.0s.



(a) Average cost (left axis) and uncertainty decomposition (right axis) for each episode over the course of learning.



(b) Average cost (left axis) and ratio of epistemic to total uncertainty (right axis) for each episode over the course of learning.

Fig. 3.13 **Cart-double-pole**: Average cost (left axis) and uncertainty (right axis) for each episode over the course of learning.

3.2.2 The Evolution of Distributions over States and Costs

I obtain Monte Carlo uncertainty estimates by rolling out state-action values through the dynamics model using the one-step predictions given in Eq. 2.25. For each state visited, I calculate the cost according to Eq. 2.3.

PILCO's policy search method is formulated so that the cost is computed directly (see Eq. 2.2) rather than via the state (although the cost is implicitly a function of the state). It is therefore important to examine how the distributions over states and costs evolve differently as they are propagated through the model, specifically in the context of the Monte Carlo method presented in Section 2.4. I illustrate this, as shown in Fig. 3.14, by performing 9 steps through a simplified 1-dimensional transition function derived from the trigonometric model presented in section 2.2. The transition function represents the input space $\mathbf{x}_t \in [-5,5]$ and is trained on targets representing the subsequent state $\mathbf{x}_{t+1}$ rather than state differences as is done with PILCO.

Fig. 3.14 shows the predictive mean and 95% confidence interval over observed state transitions as well as 4 example functions drawn from the posterior distribution over the model weights $q(\mathbf{w})$. To illustrate the evolution of state and cost distributions for the first step through the transition function, I sample $M = 1000$ sets of weights $\mathbf{w} \sim q(\mathbf{w})$ and step through $N = 1000$ starting states drawn from $\mathbf{x}_0 \sim \mathscr{U}(-5,5)$ according to Eq. 2.25. I repeat this process 8 more times; but for each step, I feed the subsequent state $\mathbf{x}_{t+1}$, as predicted by Eq. 2.25, back into the equation as the input state $\mathbf{x}_t$. At each step I calculate the cost using Eq. 2.3 with width $\sigma_c = 1.5$ and target state $\mathbf{x}_{\text{target}} = 0$. No explicit policy is used here. The experiment can be viewed as if there is only one action and the agent selects that action at each opportunity. This enables states and costs to evolve freely under the system dynamics.

Fig 3.15 shows the evolution of the distributions over states and costs. Panels 3.15a and 3.15b show the input distributions; panels 3.15c and 3.15d show the distributions after 1 transition; and panels 3.15e and 3.15f show the distributions after 9 transitions through the transition function shown in Fig. 3.14. The distribution over costs associated with the uniform initial distribution over states reveals the locally quadratic nature of the cost function where the high density at $\mathbb{C}(x) = 1$ indicates saturation of costs associated with states further away from the target state. After the first step, the distribution over states tightens around the target state with a peak about $\mathbf{x} = -1$. The reduction in the state distribution tails means that the cost no longer saturates and the tightening around the target state has increased the cost density in the region of $\mathbb{C}(\mathbf{x}) = 0$. Finally, after 9 steps, the state distribution flattens out with small peaks around the $\pm\mathbf{x} = 2$. The high density around $\mathbb{C}(x) = 0$ reduces and a bump

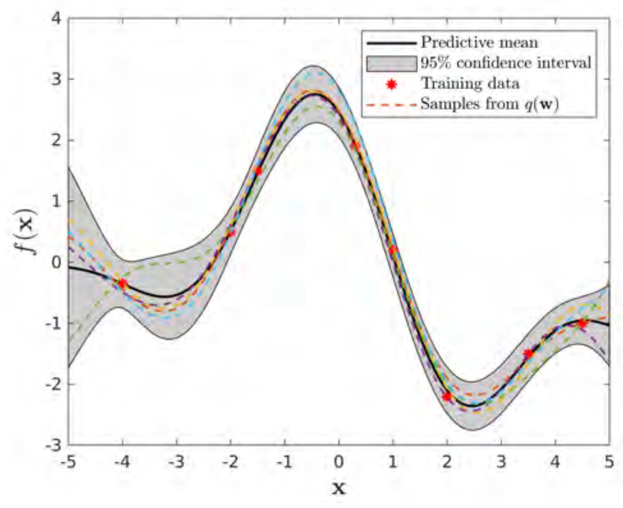
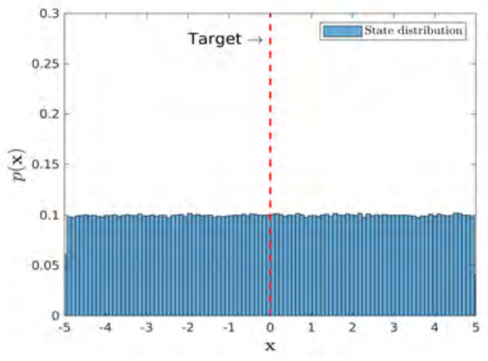


Fig. 3.14 A 1-dimensional transition function example

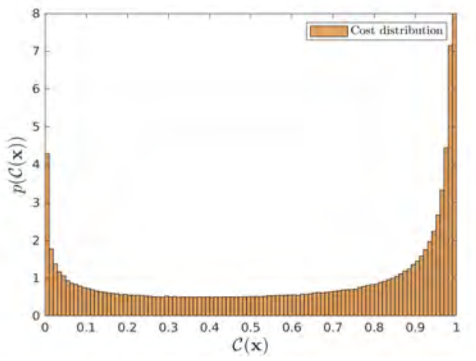
forms around the $\mathbb{C}(x) = 0.8$ position corresponding to the peaks of the state distribution at about $\pm\mathbf{x} = 2$.

3.2.3 A Decomposition of Cost Function Uncertainty

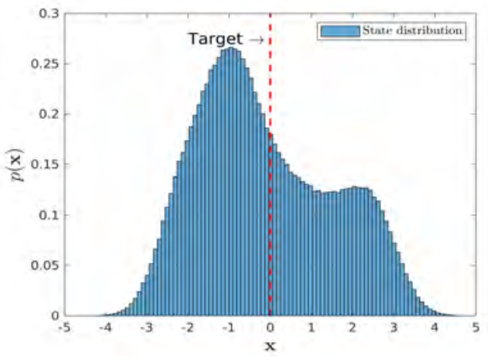
A toy data set with target noise variance 0.0625 is used to illustrate the Monte Carlo scheme for estimating the aleatoric and epistemic uncertainties present in the cost $\mathbb{C}(\mathbf{x})$ under policy π. Similar to the previous experiment, a 1-dimensional transition function is used and an explicit policy is not defined. This is viewed as if there is only one action available for selection and the agent selects that action at each opportunity which allows the system dynamics to evolve freely over time. In this experiment, the agent begins by observing 4 transition data points. It then progressively observes more data a single point at a time up to a total of 50 data points. At each observation, the trigonometric basis function model is used to form a belief over the transition dynamics of the environment. Fig. 3.16a-3.16c shows the predictive distribution over the transition data and 4 functions sampled from the posterior distribution over the parameters after 4, 25 and 50 observations, respectively. For this, the lengthscale of the spectral points is purposely set to a large value to over fit the data as can be seen in Fig. 3.16a. This is done for two reasons: to artificially induce more epistemic uncertainty into the model and to be more representative of a higher dimensional space where the observation of a single data point provides the model with more information than in the 1-dimensional case. The figures show that as more data is observed, the model becomes increasingly confident about the parameters. This can be seen in the functions drawn from the



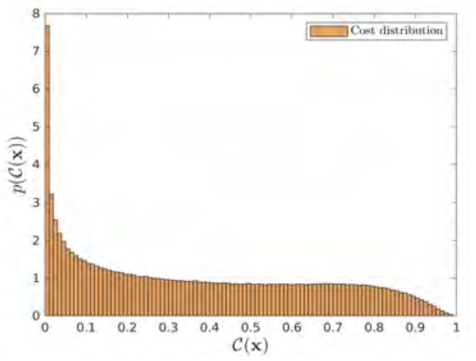
(a) Input distribution over states



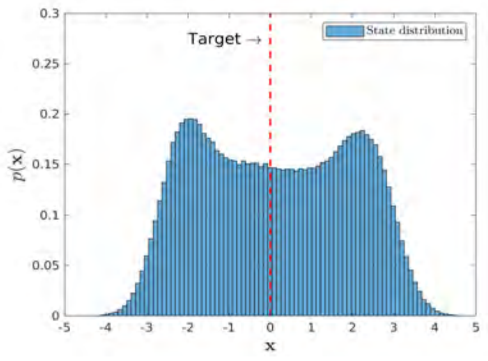
(b) Input distribution over costs



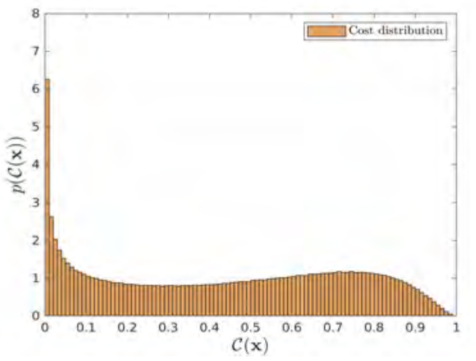
(c) Distribution over states after 1 transition



(d) Distribution over costs after 1 transition



(e) Distribution over states after 9 transitions



(f) Distribution over costs after 9 transitions

Fig. 3.15 The evolution of state and cost distributions with increasing numbers of steps through the transition function.

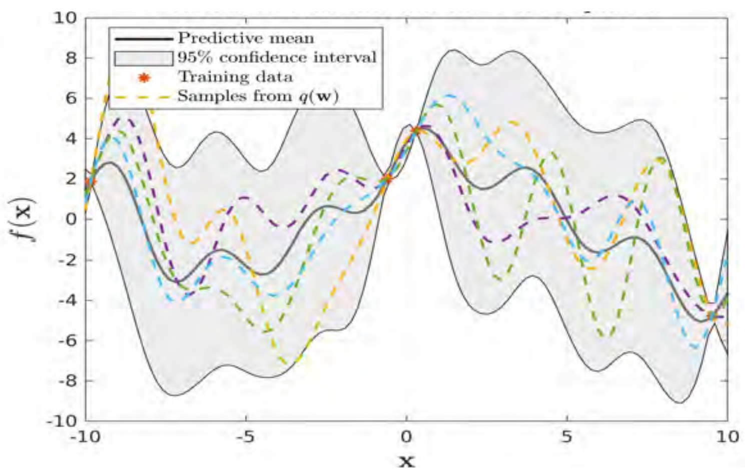
posterior distribution over the model parameters which increasingly resemble the predictive mean as confidence grows.

Each time a new data point is observed, Monte Carlo trajectory roll-outs are performed through the model using the procedure described in Sec. 2.4, the cost is calculated for a target of $x = 0$ and the uncertainty in the cost is decomposed into its aleatoric and epistemic components using Eq. 2.4. The Monte Carlo roll-outs are performed for $(M = 100, N = 100, T = 100)$ and transition noise of variance 0.09. It is important to note that the sources of uncertainty arise within the model and the transitions through the model but they are measured here with respect to the cost.

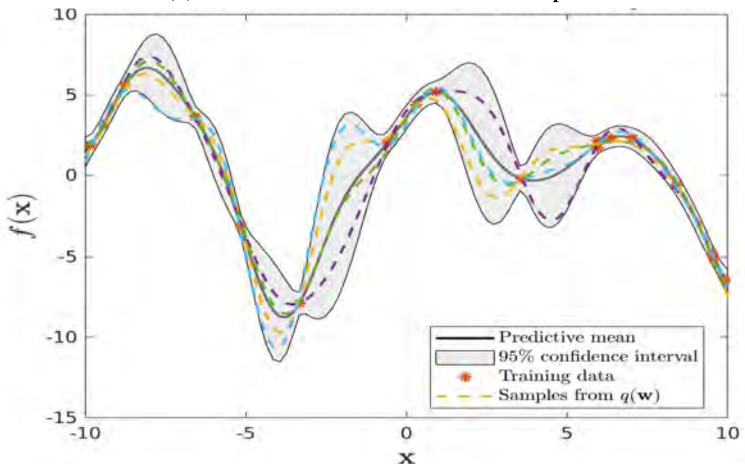
Fig. 3.17 shows the uncertainty breakdown as a function of number of data points. As the number of data points increases the model becomes more confident about the parameters and the epistemic uncertainty decreases. With the reduction in epistemic uncertainty, the total uncertainty also reduces and the gap between the aleatoric and total uncertainty decreases. This is because each time the roll-outs are performed, they are done so under the same policy. Should the policy be different after each new observation (as is typically the case after observing new data) the uncertainties would be different under each policy as will be demonstrated for the PILCO experiments. Finally, one might expect the epistemic uncertainty to decrease monotonically as more data is observed and roll-outs are performed under the same policy, but the reduction appears noisy. This could be a consequence of the Monte Carlo approach and should reduce with a larger number of roll-outs.

3.3 Discussion

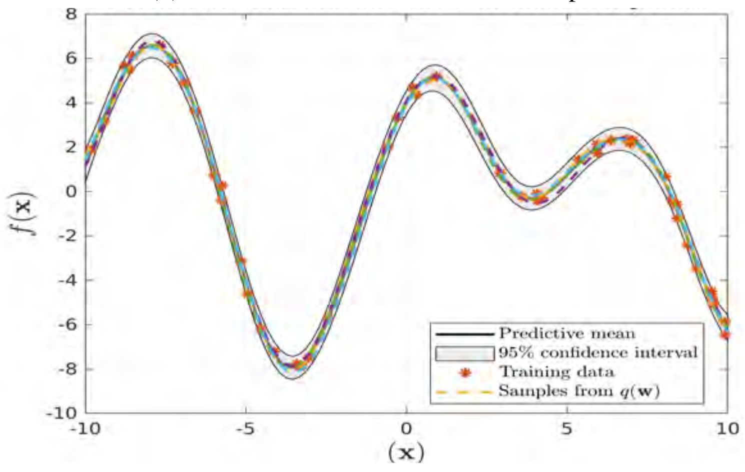
The results in section 3.2.1 demonstrate that there is a clear correlation between the ratio of epistemic to total cost uncertainty under a given policy π and the rate of PILCO's learning. Policies with higher ratios of epistemic to total cost uncertainty resulted in more efficient learning and policies with lower ratios learned more slowly. The ratio of epistemic and total cost uncertainties is an interesting metric which takes into account both the aleatoric and epistemic uncertainty values and provides a relativistic summary of cost uncertainty for a given policy. The result for the Pendubot environment (Fig. 3.9) that considered the ratio of epistemic to total cost uncertainty successfully highlighted policies that lead to large step decreases in average cost which were harder to interpret when only taking either the epistemic or aleatoric components into account. For example, in Fig. 3.9a the epistemic cost uncertainty for the policies in episodes 8 and 9 are similar but when viewed as a ratio the policy associated with episode 8 is highlighted as that which has a larger impact on learning.



(a) Predictive distribution for 4 data points



(b) Predictive distribution after 25 data points



(c) Predictive distribution after 50 data points

Fig. 3.16 : Fig 3.16a - 3.16c show the predictive distribution over the data and samples from the posterior after 4, 25 and 50 observed data points. The lengthscale of the spectral points is purposely set to a high value to over fit the data and imitate high dimensional space. As more data is observed the model becomes increasingly confident about the parameter values and as a consequence the posterior distribution narrows.

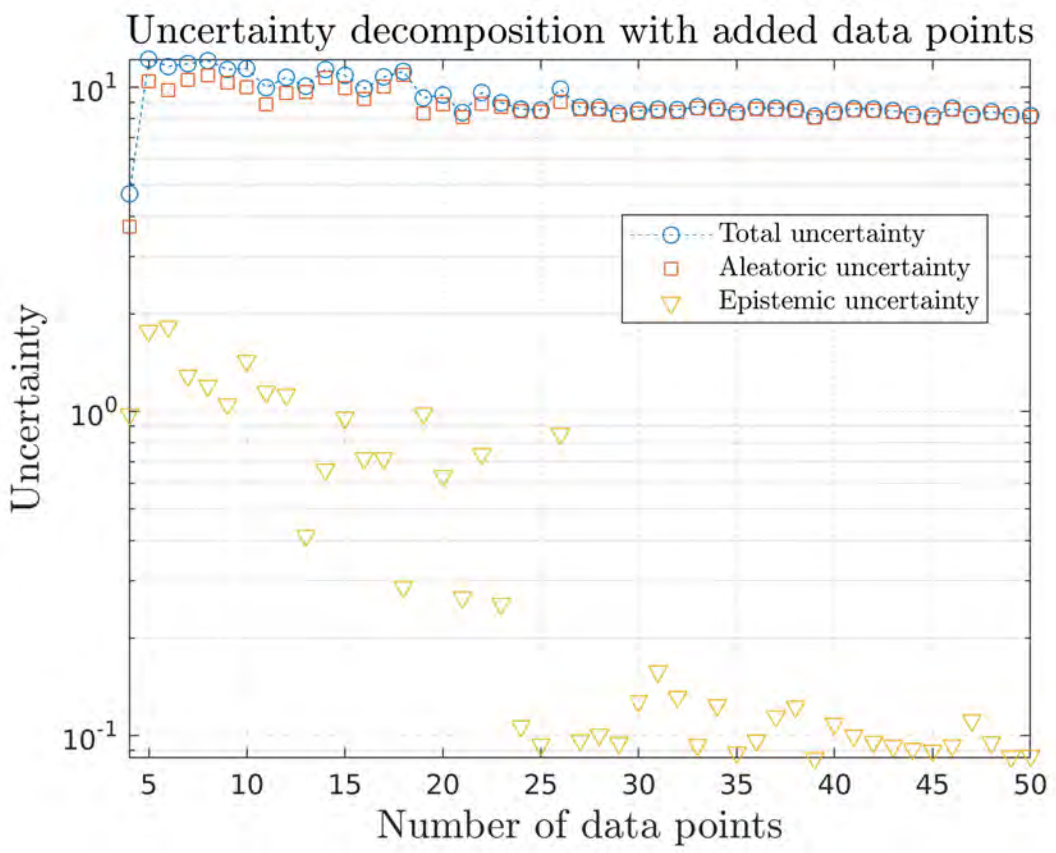


Fig. 3.17 Relating to Fig. 3.16. Trajectory roll-outs are performed through the transition function each time a new data point is observed (from 4 to 50). For each roll-out the uncertainty in the cost is decomposed. As more data is observed the model becomes increasingly confident about its parameters and consequently the epistemic uncertainty reduces due to contraction of the posterior distribution.

For the cart-double-pendulum environment it is more difficult to pin-point examples where the ratio of epistemic to total cost uncertainty clearly highlights which of two similar epistemic or aleatoric cost uncertainty values has greater impact on learning, as was done the pendubot environment. This could be because cart-double-pendulum is significantly more complex than the pendubot. There is still, however, a pronounced correlation between the ratio of epistemic to total cost uncertainty and the rate of learning (Fig. 3.13). Although the cart-double-pendulum simulation was only run for 80 episodes due to time constraints, one would expect that the ratio of epistemic to total uncertainty would begin to decrease with more episodes as the policy becomes refined and the model gains more confidence, as was the case for the pendubot environment.

For the cart-double-pendulum environment (Fig. 3.13) PILCO learns to swing the pendulums to the upright position in 10 episodes but then takes a further 33 to learn to maintain them in a balanced state. Indeed, the act of balancing the pendulums is a significantly harder problem to solve than swinging them into position. During this phase the learning rate is slow which is indicative of the absence of an exploration scheme. PILCO always exploits the policy associated with the minimum expected long-term cost which leads to the agent being overly conservative in complex environments where exploration is required to properly explore the state-space. In this setting, the use of the saturating cost function initiates a natural exploration phase where the policy seeks out uncertain states. This is because an unexplored region of the state space with a wide state distribution is more likely to have substantial tails that extend into a low-cost region than a more peaked distribution. However, the gentle learning gradient during this period indicates that learning is not taking place in an efficient manner and highlights the need for an active-exploration algorithm. Future work could incorporate the Monte Carlo method presented in Section 2.4 into an objective function to create a balance between cost and the ratio of epistemic to total cost uncertainty. The objective could then be tuned to either seek policies with higher or lower ratios of epistemic to total cost uncertainty and the effect on learning efficiency could be examined.

A closer look at propagating uncertainties:

The approximating model predictive variance in Eq. 2.24 has two terms; one representing the noise on the target data and the other, the uncertainty over of the model parameters. One might incorrectly assume that the uncertainty decomposition presented in Eq. 2.41 could be estimated in a simpler way by separately accumulating the terms of Eq. 2.24 each time a step is taken through the transition function to get estimates of the aleatoric and epistemic uncertainties in the transition function. The following sources of uncertainty need to be considered:

Observation noise: A source of aleatoric uncertainty is the observation noise. For example, the transition function is really noise free but because the equipment has a measurement tolerance it cannot reveal the true value of the transition. The equipment will never be able to reveal the true state values but if a distribution is propagated through the predictive distribution the model may be able to predict the true state.

Process noise: A further source of aleatoric uncertainty is process noise. This accounts for uncertainties that are inherent in the dynamics of the system which mean that you cannot predict the state exactly. These could arise when not all parts of the system are measured. For example, friction that changes as a function of heat.

Parameter uncertainty: The primary source of epistemic uncertainty in model-based reinforcement learning is parameter uncertainty which is reduced with the observation of more data.

Interestingly, all three types of uncertainty propagate forward through the model in different ways. In general, observation noise is irrelevant because the model can get around this by gaining access to repeated measurements. Whereas the model does not necessarily know how to deal with process noise. Since PILCO uses a GP model, it makes sense to consider the process in terms of a GP model. GP's are good at modelling observation noise but it is complicated to deal with noise present on the inputs. The reason for this is that a GP is completely defined by a mean and covariance function and the predictive distribution depends on the inverse of the covariance matrix. So building in uncertainty into the entries of the covariance matrix and then inverting it has influences throughout the model. PILCO gets around this issue by training on the difference between noisy observations, and then recognising that one source of noise on the data comes from input noise and the other from observation noise so the noise variance is approximately double and so is divided by two.

When propagating uncertainty through the transition function, all the different types of uncertainties and the way they propagate forward need to be considered, which becomes complex. So simply accumulating the the two different terms as proposed for an uncertainty would not provide a realistic estimate of the aleatoric and epistemic uncertainties in the transitions. In particular because at each step forward the total input variance must be considered in order to guarantee that the propagation spans all eventualities of propagated uncertainty. The Monte Carlo method present in Section 2.4 provides a practical way to do this without having to deal with each source of uncertainty in a different way.

3.4 Conclusion

In this dissertation I investigate how PILCO uses uncertainty in its decision making process. I show that when PILCO is learning efficiently it is selecting policies associated with a high ratio of epistemic cost uncertainty to total cost uncertainty. In contrast, when learning is slow PILCO is selecting policies associated with a low ratio of epistemic cost uncertainty to total cost uncertainty.

To do this, I disentangle and quantify the different sources of uncertainty in a model that approximates PILCO's transition function. Since PILCO is a direct policy search method, it directly consults the cost function; therefore, I examine the influence of the uncertainty present in the transition function on uncertainty in the cost. I use variance as a metric to quantify uncertainty and use the law of total variance to decompose the uncertainty into its constituents. I introduce a gold-standard Monte Carlo scheme to separately estimate the aleatoric and epistemic uncertainties present in the cost by propagating trajectories through an approximation to PILCO's dynamics model.

PILCO evaluates a particular policy by cascading uncertain inputs through its probabilistic GP dynamics model. It does this to gain insight into the variety of states that could be encountered and uses this information to search for a low cost policy. This approach only allows PILCO to make decisions based on the total uncertainty quantified by the model's predictive distribution, when there are in fact both aleatoric uncertainty and epistemic uncertainty components present in the model. This means that PILCO could be exploring regions in the state space that are irrelevant for the task of learning control because it is making decisions based on the total uncertainty without knowledge of its constituents.

Future work could incorporate the Monte Carlo estimation method into an objective function to create a balance between cost and the ratio of epistemic cost uncertainty to total cost uncertainty. The objective function could then be tuned by means of an adjustable hyperparameter so that the agent either focuses on seeking policies with higher or lower ratios of epistemic to total cost uncertainty and the effect on learning efficiency could be examined. The adjustable hyperparameter could be treated in a similar fashion to ε-greedy exploration mechanisms where the amount of exploration is adjusted to create different behavioural traits in the agent.

References

C. G. Atkeson and J. C. Santamaria. A comparison of direct and model-based reinforcement learning. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 3557–3564. IEEE, 1997.

K. Azizzadenesheli, E. Brunskill, and A. Anandkumar. Efficient exploration through bayesian deep q-networks. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9. IEEE, 2018.

J. A. Bagnell, A. Y. Ng, and J. G. Schneider. *Solving uncertain Markov decision processes*. Carnegie Mellon University, the Robotics Institute, 2001.

M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 1471–1479, 2016.

M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 449–458. JMLR. org, 2017.

C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

L. Buesing, T. Weber, Y. Zwols, S. Racaniere, A. Guez, J.-B. Lespiau, and N. Heess. Woulda, coulda, shoulda: Counterfactually-guided policy search. *arXiv preprint arXiv:1811.06272*, 2018.

C. Chatfield. *The Analysis of Time Series—An Introduction*. Chapman and Hall, London, 4 edition, 1989. pg. 94-95.

K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, pages 4754–4765, 2018.

W. R. Clements, B.-M. Robaglia, B. Van Delft, R. B. Slaoui, and S. Toth. Estimating risk and uncertainty in deep reinforcement learning. *arXiv preprint arXiv:1905.09638*, 2019.

M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

M. P. Deisenroth. *Efficient reinforcement learning using Gaussian processes*, volume 9. KIT Scientific Publishing, 2010.

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):408–423, 2013a.

M. P. Deisenroth, A. McHutchon, J. Hall, and C. E. Rasmussen. Pilco code documentation v0. 9. 2013b.

S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. *arXiv preprint arXiv:1710.07283*, 2017.

A. Der Kiureghian and O. Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105–112, 2009.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

J. Gläscher, N. Daw, P. Dayan, and J. P. O'Doherty. States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, 66(4):585–595, 2010.

Z. D. Guo and E. Brunskill. Directed exploration for reinforcement learning. *CoRR*, abs/1906.07805, 2019. URL http://arxiv.org/abs/1906.07805.

T. Jung and P. Stone. Gaussian processes for sample efficient reinforcement learning with rmax-like exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 601–616. Springer, 2010.

G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv preprint arXiv:1702.01182*, 2017.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

Z. Lipton, X. Li, J. Gao, L. Li, F. Ahmed, and L. Deng. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

M. Lopes, T. Lang, M. Toussaint, and P.-Y. Oudeyer. Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in neural information processing systems*, pages 206–214, 2012.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

T. M. Moerland, J. Broekens, and C. M. Jonker. Efficient exploration with double uncertain value networks. *arXiv preprint arXiv:1711.10789*, 2017.

T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka. Nonparametric return distribution approximation for reinforcement learning. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 799–806, 2010.

A. Nilim and L. El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

T. Pearce, N. Anastassacos, M. Zaki, and A. Neely. Bayesian inference with anchored ensembles of neural networks, and application to exploration in reinforcement learning. *arXiv preprint arXiv:1805.11324*, 2018.

L. Prashanth and M. Ghavamzadeh. Actor-critic algorithms for risk-sensitive mdps. In *Advances in neural information processing systems*, pages 252–260, 2013.

C. S. Qazaz, C. K. Williams, and C. M. Bishop. An upper bound on the bayesian error bars for generalized linear regression. In *Mathematics of Neural Networks*, pages 295–299. Springer, 1997.

J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

J. Quiñonero-Candela, C. E. Rasmussen, A. R. Figueiras-Vidal, et al. Sparse spectrum gaussian process regression. *Journal of Machine Learning Research*, 11(Jun):1865–1881, 2010.

J. G. Schneider. Exploiting model uncertainty estimates for safe dynamic control learning. In *Advances in neural information processing systems*, pages 1047–1053, 1997.

M. L. Stein. *Interpolation of spatial data*. Springer-Verlag, 1999. pg. 24.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

A. Tamar, D. Di Castro, and S. Mannor. Learning the variance of the reward-to-go. *The Journal of Machine Learning Research*, 17(1):361–396, 2016.

S. B. Thrun and K. Möller. Active exploration in dynamic environments. In *Advances in neural information processing systems*, pages 531–538, 1992.

Y. Wan, M. Zaheer, M. White, and R. S. Sutton. Model-based reinforcement learning with non-linear expectation models and stochastic environments. 2018.

C. K. Williams and C. E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Appendix A

Derivations

A.1 Model

A.1.1 Posterior Distribution

The prior distribution over the weights $p(\mathbf{w})$ and the model likelihood $p(\boldsymbol{\Delta}|\mathbf{X},\mathbf{w},\sigma_n^2)$ are

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{0},\frac{\sigma_0^2}{m}\mathbf{I}\right) \tag{A.1}$$

$$p\left(\boldsymbol{\Delta}|\mathbf{X},\mathbf{w},\sigma_n^2\right) = \prod_{i=1}^{N}\mathcal{N}\left(\Delta_i|\mathbf{w}^{\mathrm{T}}\varphi\left(\mathbf{x}_i\right),\sigma_n^2\right). \tag{A.2}$$

Here, $\mathbf{x}$ is used instead of $\tilde{\mathbf{x}}$ for notational convenience. The posterior distribution is proportional to the product of the prior and the likelihood

$$p(\mathbf{w}|\boldsymbol{\Delta}) \propto \prod_{i=1}^{N}\mathcal{N}\left(\Delta_i|\mathbf{w}^{\mathrm{T}}\varphi\left(\mathbf{x}_i\right),\sigma_n^2\right)\mathcal{N}\left(\mathbf{w}|\mathbf{0},\frac{\sigma_0^2}{m}\mathbf{I}\right). \tag{A.3}$$

Focusing on the exponential term E

$$\begin{aligned}
E &= -\frac{1}{2\sigma_n^2}\sum_{i=1}^{N}\left\{\Delta_i - \mathbf{w}^\top\varphi\left(\mathbf{x}_i\right)\right\}^2 - \frac{m}{2\sigma_0^2}\mathbf{w}^\top\mathbf{w} \\
&= -\frac{1}{2\sigma_n^2}\sum_{i=1}^{N}\left\{\Delta_i^2 - 2\Delta_i\mathbf{w}^\top\varphi\left(\mathbf{x}_i\right) + \mathbf{w}^\top\varphi\left(\mathbf{x}_i\right)\varphi\left(\mathbf{x}_i\right)^\top\mathbf{w}\right\} - \frac{m}{2\sigma_0^2}\mathbf{w}^\top\mathbf{w} \qquad (A.4) \\
&= -\frac{1}{2}\mathbf{w}^\top\left[\sum_{i=1}^{N}\frac{1}{\sigma_n^2}\varphi\left(\mathbf{x}_i\right)\varphi\left(\mathbf{x}_i\right)^\top + \frac{m}{\sigma_0^2}\mathbf{I}\right]\mathbf{w} - \frac{1}{2}\left[-\sum_{i=1}^{N}\frac{2}{\sigma_n^2}\Delta_i\varphi\left(\mathbf{x}_i\right)^\top\right]\mathbf{w} + \mathbf{C}
\end{aligned}$$

Through comparison of the quadratic term of the above equation with that of the standard Gaussian distribution, it can be seen that posterior precision matrix $\mathbf{A}$ is

$$\mathbf{A} = \frac{1}{\sigma_n^2}\Phi\Phi^\top + \frac{m}{\sigma_0^2}\mathbf{I}. \qquad (A.5)$$

Now comparing the linear term from Equation A.4 with that of a standard Gaussian distribution gives

$$-\boldsymbol{\mu}^\top\mathbf{A} = -\sum_{i=1}^{N}\frac{1}{\sigma_n^2}\Delta_i\varphi\left(\mathbf{x}_i\right)^\top. \qquad (A.6)$$

Transposing both sides (noting that $\mathbf{A}$ is symmetrical) and multiplying through by the posterior precision gives the posterior mean $\boldsymbol{\mu}_\mathbf{w}$ as

$$\boldsymbol{\mu}_\mathbf{w} = \frac{1}{\sigma_n^2}\mathbf{A}^{-1}\Phi\boldsymbol{\Delta} \qquad (A.7)$$

A.1.2 Predictive Distribution

Consider a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution for $\mathbf{y}$ given $\mathbf{x}$

$$p(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda}\right) \qquad (A.8)$$

$$p(\boldsymbol{\Delta}|\mathbf{x}) = \mathcal{N}\left(\boldsymbol{\Delta}|\mathbf{C}\mathbf{x}+\mathbf{b},\boldsymbol{\Gamma}\right) \qquad (A.9)$$

where $\mathbf{C}$, $\mathbf{b}$ and $\boldsymbol{\mu}$ are the mean parameters and $\boldsymbol{\Lambda}$ and $\boldsymbol{\Gamma}$ are the covariance matrices. The marginal distribution of $\boldsymbol{\Delta}$ is then defined as

$$p(\boldsymbol{\Delta}) = \int p(\boldsymbol{\Delta}|\mathbf{x})p(\mathbf{x})d\mathbf{x}. \qquad (A.10)$$

The standard result for a marginal Gaussian distribution under these conditions is Bishop (2006)

$$p(\boldsymbol{\Delta}) = \mathscr{N}\left(\boldsymbol{\Delta}|\mathbf{C}\boldsymbol{\mu} + \mathbf{b}, \boldsymbol{\Gamma} + \mathbf{C}\boldsymbol{\Lambda}\mathbf{C}^{\mathrm{T}}\right). \tag{A.11}$$

Reproducing Equations 2.18 and 2.16 (for a new input $\mathbf{x}_*$) here and noting that $f(\mathbf{x}_*, \mathbf{w}) = \mathbf{w}^{\mathrm{T}}\varphi(\mathbf{x}_*) = \varphi(\mathbf{x}_*)^{\mathrm{T}}\mathbf{w}$ gives

$$p\left(\mathbf{w}|\boldsymbol{\Delta}, \mathbf{X}, \sigma_0^2, \sigma_n^2\right) = \mathscr{N}\left(\mathbf{w}|\boldsymbol{\mu}_p, \mathbf{A}^{-1}\right) \tag{A.12}$$

$$p\left(\Delta_*|\mathbf{x}_*, \mathbf{w}, \sigma_n^2\right) = \mathscr{N}\left(\Delta_*|\varphi(\mathbf{x}_*)^{\mathrm{T}}\mathbf{w}, \sigma_n^2\right). \tag{A.13}$$

Comparing these to the general result given in Equation A.11 and noting that $\varphi(\mathbf{x}_*)^{\top}\mathbf{A}^{-1}\varphi(\mathbf{x}_*) = \varphi(\mathbf{x}_*)\mathbf{A}^{-1}\varphi(\mathbf{x}_*)^{\top}$ gives the predictive mean μ_{Δ_*} and covariance σ_{Δ_*}

$$\mu_{\Delta_*} = \frac{1}{\sigma_n^2}\varphi(\mathbf{x}_*)^{\top}\mathbf{A}^{-1}\Phi\boldsymbol{\Delta} \tag{A.14}$$

$$\sigma_{\Delta_*}^2 = \sigma_n^2 + \varphi(\mathbf{x}_*)^{\top}\mathbf{A}^{-1}\varphi(\mathbf{x}_*). \tag{A.15}$$

Appendix B

Code

Code to reproduce the results in this thesis can be found at:

- PILCO (MATLAB): https://github.com/linesd/PILCO-disentangling-uncertainty
- SSGPR (Python): https://github.com/linesd/SSGPR