Adaptive Model Predictive Control by Learning Classifiers

Rel Guzman[2] REL.GUZMANAPAZA@SYDNEY.EDU.AU
Rafael Oliveira[2] RAFAEL.OLIVEIRA@SYDNEY.EDU.AU
Fabio Ramos[1,2] FABIO.RAMOS@SYDNEY.EDU.AU
[1]*NVIDIA, USA,* [2]*the University of Sydney, Australia*

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

Stochastic model predictive control has been a successful and robust control framework for many robotics tasks where the system dynamics model is slightly inaccurate or in the presence of environment disturbances. Despite the successes, it is still unclear how to best adjust control parameters to the current task in the presence of model parameter uncertainty and heteroscedastic noise. In this paper, we propose an adaptive MPC variant that automatically estimates control and model parameters by leveraging ideas from Bayesian optimisation (BO) and the classical expected improvement acquisition function. We leverage recent results showing that BO can be reformulated via density ratio estimation, which can be efficiently approximated by simply learning a classifier. This is then integrated into a model predictive path integral control framework yielding robust controllers for a variety of challenging robotics tasks. We demonstrate the approach on classical control problems under model uncertainty and robotics manipulation tasks.

Keywords: Bayesian methods, Gaussian process, model predictive control, optimisation

1. Introduction

Reinforcement learning, as a framework, concerns learning how to interact with the environment through experience, while optimal control emphasises sequential decision making and optimisation methods. The boundaries between both fields have been diminished due to deeper understanding and typical applications. Model predictive control (MPC) is an optimisation strategy for behaviour generation that consists of planning actions ahead by minimising costs throughout a horizon. Reinforcement learning and robotics can benefit from MPC by correcting behaviours while constantly estimating hyper-parameters. This controller learning capability can be achieved with data-driven approaches for MPC optimisation (Görges, 2017).

We are particularly interested in path integral (PI) control (Kappen, 2005), which is a methodology for solving nonlinear stochastic optimal control problems by sampling trajectories and computing costs. Using such methodology, model predictive path integral control (MPPI) was introduced in (Williams et al., 2016). MPPI enables robots to navigate in stochastic and partially observable environments, for example, in-car racing (Williams et al., 2018b). MPPI is a sampling-based and derivative-free method which makes it a simple yet powerful strategy to simulate actions.

Within data-driven approaches, deep reinforcement learning has been successful in solving high-dimensional control problems in simulation (Duan et al., 2016). The main limitation of deep RL is the need for many interactions with the environment, which can be impractical with a physical system due to costly evaluations (Peng et al., 2018). An alternative to reduce evaluations is to have a model of the system dynamics, also called a dynamics model or transition model. Modeling an accurate transition model inevitably leads to errors. Even so, by using data-driven approaches, it is possible to reduce the error or adapt to the expected errors in the environment (Lee et al., 2020).

Data-driven approaches have been proposed for automatic MPC tuning, which can be seen as an intersection between machine learning and control since they make use of the transition model in combination with a learnt model. For example, (Sorourifar et al., 2021) presents MPC under uncertainty over model parameters with Bayesian optimisation (BO) that handles constraints of system parameters in a tank reactor. (Lee et al., 2020) addresses different environment contexts where a robot's dynamics could change due to a component malfunctioning. Other approaches propose inferring simulation parameters based on data instead of uniform parameter randomisation (Peng et al., 2018; Ramos et al., 2019). In another example, the controller optimisation is able to handle heteroscedastic noise for control tasks (Guzman et al., 2020). Intuitively, heteroscedastic noise is a type of noise that changes with input variables. For example, in stochastic MPC, the noise associated with the stochastic process changes significantly with the temperature hyper-parameter (Guzman et al., 2020), making hyper-parameter tuning quite challenging from an optimisation perspective.

We propose a data-driven approach to optimise a stochastic controller by adapting the transition model parameters to the environment. For costly system evaluations, we use surrogate-based optimisation. Instead of merely optimising objective functions, we optimise an alternative function approximating the more complex, higher-order model to quickly find the local or global optimum. These are used when function evaluations are expensive and noisy due to the environment characteristics. In this context, a surrogate model can amortise the optimisation. Surrogate-based modelling is usually associated with Gaussian processes (Rasmussen and Williams., 2006). We consider single-objective optimisation problems with continuous inputs and noisy observations. We work on the case where the noise variance depends on the input. That leads us to heteroscedastic optimisation, which is a more realistic approach than the typical homoscedastic assumption. Heteroscedastic noise can cause problems for the surrogate model and make the optimisation method deviate from the maximum (Wang and Ierapetritou, 2017). To solve this issue, we aim at finding an optimisation method for stochastic simulations under heteroscedastic noise.

BO uses a Gaussian process (Rasmussen and Williams., 2006) as a surrogate model commonly used for costly black-box functions. BO proposes solutions according to an acquisition function that encodes the trade-off between exploration and exploitation of the objective function. GPs have excellent generalisation properties, but their computational cost can be prohibitive for big data. Additionally, standard GPs provide analytical solutions for posteriors under homoscedastic noise, while heteroscedastic approximations might require computationally expensive approximations.

An alternative formulation to BO, which allows the utilisation of simple classifiers within the optimisation loop, was proposed in (Tiao et al., 2021) as Bayesian optimisation by Density Ratio Estimation (BORE). The method introduces the concept of relative density ratio, which is used to estimate the expected improvement acquisition function (Bull, 2011). The main advantage of this formulation is that density ratios are bounded between 0 and 1 and can be estimated using any off-the-shelf probabilistic classifier. Classifiers are easy to train and can handle a variety of input noise types, including heteroscedastic, without major modifications to the classification function.

Contributions: The main contribution of this work is a new robust and adaptive MPC method that automatically estimates distributions of model parameters and MPC hyper-parameters such as the temperature by continuously updating a classifier that acts as a proxy for a Bayesian optimisation step. We demonstrate that the approach provides superior performance in general control problems and manipulation tasks under model uncertainty.

2. BACKGROUND

2.1. Stochastic Model Predictive Control

Model predictive control resides on the idea of optimising an action sequence up to certain horizon T while minimising the sum of action costs, starting from a current state. MPC returns a next optimal action a^* that is sent to the system actuators. Unlike classical deterministic MPC, stochastic MPC allows disturbances over the states. A stochastic MPC method models disturbances as random variables. At each time step t, stochastic MPC generates sequences of perturbed actions $V_t = \{v_i\}_{i=t}^{t+T}$ where $v_i = a_i^* + \epsilon_i$ and $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$, based on a roll-over sequence of optimal actions $\{a_i^*\}_{i=t}^{t+T}$ that start at $t = 0$. Each action results in a state produced by a transition or dynamics model $s_{t+1} = f(s_t, a_t)$, and action sequences result in a state trajectory $\mathrm{S}_t = \{s_{t+i}\}_{i=1}^{T}$. Each trajectory has a cumulative cost C determined by instant costs c and a terminal cost q:

$$C(\mathrm{S}_t) = q(s_{t+T}) + \sum_{i=1}^{T-1} c(s_{t+i}) \,. \tag{1}$$

In stochastic MPC, the goal is to minimise the expected $C(S_t)$. The stochastic MPC method known as model predictive path integral (MPPI) and its variations (Williams et al., 2016, 2018b) provide optimal actions for the entire horizon following an information-theoretic approach. Constraints over the states are determined by the transition model, and the actions are constrained according to their limits. After M simulated rollouts, MPPI updates the sequence of optimal actions and weights:

$$a_i^* \leftarrow a_i^* + \sum_{j=1}^{M} w(\mathrm{V}_t^j)\epsilon_i^j \,, \qquad w(\mathrm{V}_t) = \frac{1}{\eta} \exp\left(-\frac{1}{\lambda}\left(C(\mathrm{S}_t) + \frac{\lambda}{\sigma_\epsilon^2}\sum_{i=t}^{t+T} a_i^* \cdot v_i\right)\right) \,, \tag{2}$$

where $j \in \{1, \ldots, M\}$ and η is a normalisation constant so that $\sum_{j=1}^{M} w(\mathrm{V}_t^j) = 1$. The hyper-parameter temperature $\lambda \in \mathbb{R}^+$, $\lambda \to 0$ leads to more peaked distributions for actions (Williams et al., 2018a). The hyper-parameter σ_ϵ^2 is the control variance, which leads to more varying and forceful actions when it is increased.

2.2. Classic Bayesian Optimisation

Bayesian optimisation (BO) has been widely applied in robotics and control to optimise black-box (i.e., derivative-free) functions that are costly to evaluate in applications such as robotics. That is due to the usual cost of running experiments in real robots. We use BO to perform global optimisation in a given search space $\mathcal{X}$. BO uses a Gaussian process (GP) (Rasmussen and Williams., 2006) as a surrogate model $\mathcal{M}$ to internally approximate an objective function $g : \mathcal{X} \to \mathbb{R}$. BO defines an optimisation problem $\mathbf{x}^* \in \mathrm{argmax}_{\mathbf{x} \in \mathcal{X}}\, g(\mathbf{x})$. Then, given a set of collected observations $\mathcal{D}_{1:t}$, BO constructs a surrogate model that provides a posterior distribution over the objective function g. This posterior is used to construct the acquisition function h, which measures both performance and uncertainty of unexplored points. The next step is optimising the acquisition function, obtaining a sample $(\mathbf{x}_t, y_t)$. BO is summarised in Algorithm 1.

A popular acquisition function in the BO literature is the expected improvement (EI) (Bull, 2011). At iteration t, one can define $y_t^* := \max_{i<t} y_t$ as an optimal incumbent. The expected improvement is then defined as:

$$h_{\mathrm{EI}}(\mathbf{x}|\mathcal{D}_{t-1}) := \mathbb{E}[\max\{0, f(\mathbf{x}) - y_t^*\} | \mathcal{D}_{t-1}] \,. \tag{3}$$

In the case of a GP prior on $f|\mathcal{D}_{t-1} \sim \mathcal{GP}(\mu_{t-1}, k_{t-1})$, for any point $\mathbf{x}$ where the predictive standard deviation of the GP is non-zero, i.e., $\sigma_{t-1}(\mathbf{x}) = \sqrt{k_{t-1}(\mathbf{x}, \mathbf{x})} > 0$, the EI is given by:

$$h_{\mathrm{EI}}(\mathbf{x}|\mathcal{D}_{t-1}) = (\mu_{t-1}(\mathbf{x}) - y_t^*)\Psi(s_t) + \sigma_{t-1}(\mathbf{x})\psi(s_t)\,, \tag{4}$$

where $s_t := \frac{\mu_{t-1}(\mathbf{x}) - y_t^*}{\sigma_{t-1}(\mathbf{x})}$, if $\sigma_{t-1}^2(\mathbf{x}) := k_{t-1}(\mathbf{x}, \mathbf{x}) > 0$. For points where $\sigma_{t-1}(\mathbf{x}) = 0$, i.e., there is no posterior uncertainty, we simply have $h_{\mathrm{EI}}(\mathbf{x}|\mathcal{D}_{t-1}) = 0$. Here $\Psi(s_t)$ and $\psi(s_t)$ denote the cumulative and probability density functions of the standard normal distribution evaluated at s_t.

2.3. Bayesian Optimisation by Learning Classifiers

BO is hindered by the GP surrogate model, for which it has limitations such as cubic computational cost in training and not being directly amenable to handle variable noise structures such as heteroscedasticity. The extensions proposed to address those issues are restrained by the necessity to ensure analytical tractability and typically make strong and oversimplifying assumptions. For example, (Roy et al., 2013) proposed an heteroscedastic BO approach that uses a variational approximation that can be expensive to compute. BO by Density Ratio Estimation (BORE) (Tiao et al., 2021) was proposed based on a reformulation of the expected improvement acquisition function (Equation 3) and bypassed the challenges of analytical tractability in GP-based approaches. BORE works by selecting points according to a density ratio similar to the Tree-structured Parzen Estimator (TPE) proposed by Bergstra et al. (2011). TPE divides the observations, based on some quantile hyper-parameter, into a first group that gave the best scores and a second group containing the rest. Then, the goal is to find inputs that are more likely to be in the first group. In order to propose a new sampling point, TPE computes the ordinary density ratio from Equation 5 between the probability $a(\mathbf{x})$ of being in the first group, and the probability $b(\mathbf{x})$ of being in the second group. Instead, BORE uses the γ-relative density ratio r_γ. The ordinary and γ-relative density ratios are specified by:

$$r_0(\mathbf{x}) = a(\mathbf{x})/b(\mathbf{x})\,, \tag{5}$$

$$r_\gamma(\mathbf{x}) := \frac{a(\mathbf{x})}{\gamma a(\mathbf{x}) + (1-\gamma)b(\mathbf{x})}\,. \tag{6}$$

BORE approximates the γ-relative density ratio to a binary class posterior probability as $r_\gamma(\mathbf{x}) \simeq \gamma^{-1}\pi(\mathbf{x})$, where π computes the probability of $\mathbf{x}$ belonging to a positive class $\pi(\mathbf{x}) = p(z = 1 \mid \mathbf{x})$. The binary label z introduced here denotes a negative or positive class, and its meaning corresponds to whether the point should be selected or not. In BORE, given a maximisation objective, we set $z := \mathbb{I}[y \geq \tau]$, indicating of whether the corresponding observation y at a point $\mathbf{x}$ is above the γth quantile τ of the (empirical) observations distribution, i.e., $\gamma = p(y \geq \tau)$. In the end, computing the acquisition function h from the classical BO method (Algorithm 1) is reduced to classifier training.

As shown in Algorithm 2, the optimisation depends on the hyper-parameter $\gamma \in (0, 1)$, which influences the exploration-exploitation trade-off. A smaller γ encourages exploitation. In this work, we anneal γ until it reduces to a minimum value close to 0. The reason is to induce more exploration initially and avoid local minima as much as possible.

3. Methodology

3.1. Dynamics Model Uncertainty

The dynamics of the environment is modelled as a Markovian transition model. We consider a transition model with states $\mathbf{s} \in \mathcal{S}$ and admissible actions $\mathbf{a} \in \mathcal{A}$. The state follows Markovian

Algorithm 1: Bayesian Optimisation

```
input  : Sampling iterations $n$; search space $\mathcal{X}$;
         hyper-parameters of $h$
output : $(\mathbf{x}^*, y^*)$
for $t = 1$ to $n$ do
    Fit a GP model $\mathcal{M}$ on the data $\mathcal{D}_{t-1}$
    Find $\mathbf{x}_t = \text{argmax}_{\mathbf{x}\in\mathcal{X}} h(\mathbf{x}, \mathcal{M}, \mathcal{D}_{t-1})$
    $y_t \leftarrow f(\mathbf{x}_t)$
    $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$
end
```

Algorithm 2: BORE

```
input  : Sampling iterations $n$; search space $\mathcal{X}$;
         $\gamma \in (0,1)$; classifier to train $\pi : \mathcal{X} \to [0,1]$
output : $(\mathbf{x}^*, y^*)$
for $t = 1$ to $n$ do
    Train the probabilistic classifier $\pi$
    Find $\mathbf{x}_t = \arg\max_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})$
    $y_t \leftarrow f(\mathbf{x}_t)$
    $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$
end
```

dynamics, $\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t)$, with a transition function f and a reward function r that evaluates the system performance given a state and action $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. That transition model can be learned or assumed from expert knowledge.

We propose adapting stochastic MPC to the environment by exposing the robot to different possible scenarios by defining a transition model parameterised by a random variable θ. To find an optimum θ, we add randomisation at each MPC trajectory rollout. We define a random vector of transition model parameters $\boldsymbol{\theta}$ and adapt them to the stochastic MPC controller. Each transition model parameter follows a probability distribution parameterised by $\boldsymbol{\psi}$:

$$\boldsymbol{\theta} \sim p_{\boldsymbol{\theta}}(\boldsymbol{\theta}; \boldsymbol{\psi}), \qquad \mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t + \boldsymbol{\epsilon}_t, \boldsymbol{\theta}) \ , \tag{7}$$

where $\mathbf{s}_t$ is the state obtained at time t, and $\mathbf{a}_t + \boldsymbol{\epsilon}_t$ is the perturbed action as described in stochastic MPC (subsection 2.1). Note that $\boldsymbol{\theta}$ is now an input to the dynamics model. Finally, optimal actions found by MPC are sent to the system using the dynamics model f.

3.2. An Adaptive Control Formulation

We do not directly aim at finding parameters that match the observed dynamics as we would do in system identification (Romeres et al., 2016). Instead, we adapt model parameter distributions to the controller, which means those resulting distributions may be close to their true values. The uncertainty of those parameters would allow the controller to adapt under different environment circumstances or characteristics, such as the size of an obstacle or the length of a robot component.

We use the optimal MPC hyper-parameters by adapting them to the transition model parameters. In the case of MPPI (Williams et al., 2018a), the hyper-parameters considered are λ and σ_ϵ as described in subsection 2.1. All controller hyper-parameters are collectively described as $\boldsymbol{\phi}$. We introduce the reinforcement learning objective of optimising the episodic cumulative reward R. An overview of the proposed framework is shown in Figure 1. The objective is to solve the reward optimisation problem where we jointly estimate $\boldsymbol{\psi}$, which are hyper-parameters for the dynamics model parameter distribution $p_{\boldsymbol{\psi}}(\boldsymbol{\theta})$, and the controller hyper-parameters $\boldsymbol{\phi}$:

$$\boldsymbol{\psi}^\star, \boldsymbol{\phi}^\star = \underset{\{\boldsymbol{\psi}, \boldsymbol{\phi}\}}{\text{argmax}} \ R(\boldsymbol{\psi}, \boldsymbol{\phi}) \ . \tag{8}$$

To perform the optimisation in Equation 8, we adopt a surrogate-based optimisation method that can handle noisy gradient-free functions that are common in control tasks. To this end, we use BORE since it provides a number of advantages over traditional GP-based BO, as discussed

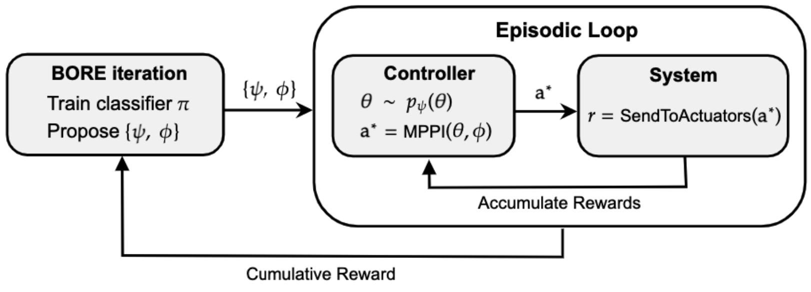

Figure 1: An overview of adaptive MPC by learning classifiers.

previously, and it can handle heteroscedasticity without modifications to the core method. Heteroscedastic noise is common when tuning stochastic MPC in control problems (Guzman et al., 2020). A difference between GP-based BO and BORE is that BO filters noise directly via the GP model, while BORE relies on the classifier to account for label noise. Therefore, to mitigate noise effects, we propose to optimise the objective function averaged over a small number n_e of episodes.

To better understand the optimisation framework, Algorithm 3 describes how to estimate the optimal controller and dynamics hyper-parameters $\mathbf{x}^* = \{\boldsymbol{\psi}^*, \boldsymbol{\phi}^*\}$ using a binary classifier. Following the reinforcement learning literature, we define the cumulative reward $R = \sum_{i=1}^{n_s} r_i$, where n_s is the number of time steps in an episode, and set our goal as maximising the expected cumulative reward $g = \mathbb{E}[R]$. We compute an empirical expected cumulative reward g by averaging R over n_e episodes. The classifier π_t is trained by first assigning labels $\{z_k\}_{k=1}^t$ to the data observed up to the current iteration t. For training, the classifier uses the auxiliary dataset:

$$\{(\boldsymbol{\psi}_k, \boldsymbol{\phi}_k, z_k)\}_{k=1}^t \ , \tag{9}$$

where the labels are obtained by separating the observed data according to $\gamma \in (0, 1)$ by computing the γth quantile of $\{g_k\}_{k=1}^t$:

$$\tau \leftarrow \Phi^{-1}(\gamma), \qquad z_k \leftarrow \mathbb{I}[g_k \geq \tau] \quad \text{for } k = 1, \dots, t \ . \tag{10}$$

The exploration-exploitation trade-off is balanced by γ, with small γ encouraging more exploitation. Instead of keeping γ fixed, we define a strategy that first explores the search space with an initial high γ_1 that decays linearly across the iterations until a final γ_n. Inputs predicted as positive labels are considered to have a higher reward, and one of them is selected by maximising the classifier output:

$$\boldsymbol{\psi}_t, \boldsymbol{\phi}_t = \underset{\{\boldsymbol{\psi},\boldsymbol{\phi}\}\in\mathcal{X}}{\text{argmax}}\, \pi_{t-1}(\boldsymbol{\psi}, \boldsymbol{\phi}) \ . \tag{11}$$

Note that this is equivalent to acquisition function maximisation in conventional BO and allows the method to suggest candidate solutions efficiently. For better performance, the maximisation can be carried out with a global optimisation method (e.g., Arnold and Hansen, 2010).

3.3. Choosing the Classifier

The probabilistic binary classifier in Equation 11 has to be chosen considering the observation noise in the task. Some methods used in Tiao et al. (2021) are XGBoost, multi-layer perceptron (MLP), and Random Forests (RF). For example, as an ensemble method, RF combines decision trees via

Algorithm 3: Adaptive MPC by learning classifiers

```
input  : Sampling iterations $n$
         Search space $\mathcal{X}$
         Initial $\gamma_1$ and final $\gamma_n$
         Probabilistic binary classifier $\pi : \mathcal{X} \to [0,1]$
output : $(\boldsymbol{\psi}^*, \boldsymbol{\phi}^*, g^*)$
for $t = 1$ to $n$ do
    $\gamma_t = \gamma_1 - \frac{t-1}{n-1}(\gamma_1 - \gamma_n)$            // linear $\gamma$ decay
    $\tau \leftarrow \Phi^{-1}(\gamma_t)$            // compute $\gamma_t$-th quantile of $\{g_k\}_{k=1}^{t}$
    $z_k \leftarrow \mathbb{I}[g_k \geq \tau]$ for $k = 1, \ldots, t-1$            // assign labels to the observed data points
    Train the classifier $\pi_{t-1}$ using $\{(\boldsymbol{\psi}_k, \boldsymbol{\phi}_k, z_k)\}_{k=1}^{t-1}$            // acquisition function according to BORE
    $\boldsymbol{\psi}_t, \boldsymbol{\phi}_t = \arg\max_{\{\boldsymbol{\psi},\boldsymbol{\phi}\}\in\mathcal{X}} \pi_{t-1}(\boldsymbol{\psi}, \boldsymbol{\phi})$            // estimate new hyper-parameters and model parameters
    for $j = 1$ to $n_e$ do
        $R_j^{(t)} = 0$
        for $i = 1$ to $n_s$ do
            $\boldsymbol{\theta} \sim p_{\boldsymbol{\theta}}(\boldsymbol{\theta}; \boldsymbol{\psi_t})$            // sample from the parameter distributions
            $a_i^* = \text{MPC}(f, \boldsymbol{\theta}_t, \boldsymbol{\phi}_t)$            // use estimated parameter distributions in the new transition model
            $r_i = \text{SendToActuators}(a_i^*)$            // evaluate the first action to take in the optimal trajectory
            $R_j^{(t)} \mathrel{+}= r_i$            // accumulate rewards
        end
    end
    Decrease $\gamma$ to reduce explorability
    $g_t = 1/n_e \sum_{j=1}^{n_e} \left[R_j^{(t)}\right]$
    $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(\boldsymbol{\psi}_t, \boldsymbol{\phi}_t, g_t)\}$
end
```

bagging. The number of decision trees should be sufficiently large to reduce classification variance without increasing the bias. We highlight the study of this method since it is an out-of-the-box classification method.

4. Experiments

We consider the problem of optimising the function $R(\mathbf{x}) = g(\mathbf{x}) + \epsilon$ with heteroscedastic input-dependent noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2(\mathbf{x}))$, where $\sigma_\epsilon^2(\mathbf{x})$ denotes an input-dependent noise variance. In these experiments the variables we optimise are the controller hyper-parameters $\boldsymbol{\phi}$, and the variable $\boldsymbol{\psi}$ that parameterises transition model parameter distributions. We aim at optimising the true noise-free function g although we only have access to the cumulative reward R. We use R as the objective to optimise in the simulator experiments. We evaluate the performance of the proposed adaptive model predictive control framework in several experiments described below.

4.1. Simulation Experiments

In this section, we assess the performance of the adaptive controller framework in control and robotic tasks. Specifically, we ran tests in simulated environments, such as Pendulum, Half-Cheetah, and Fetchreach from OpenAI[1] with dense reward functions. The same functions taken from Guzman et al. (2020) are used to compute instant costs for MPPI. We also experimented with the reaching task for the Panda robot environment from Bhardwaj et al. (2021), with a single obstacle, a fixed

1. OpenAI Gym: https://gym.openai.com

Environment	n_e	T	M	Control hyp.	Distribution parameter range		True parameter
Pendulum	1	10	10	$\lambda \in [0.01, 50]$ $\sigma_\epsilon \in [1.0, 10]$	$\mu_l \in [0.5, 1.6]$	$\sigma_l \in [0.001, 0.1]$	$l = 1.0$
Half-Cheetah	18	15	10	$\lambda \in [0.01, 1.0]$ $\sigma_\epsilon \in [0.05, 2.0]$	$\kappa_{m,\mu} \in [0.2, 2.0]$	$\kappa_{m,\sigma} \in [0.001, 0.1]$	$\kappa_m = 1.0$
Fetchreach	90	12	3	$\lambda \in [0.01, 0.03]$ $\sigma_\epsilon \in [0.001, 0.5]$	$\kappa_{d,\mu} \in [1.0, 50]$	$\kappa_{d,\sigma} \in [0.001, 0.6]$	$\kappa_d = 1.0$
Panda	10	150	20	$\lambda \in [0.01, 2.0]$	$x_\mu \in [0.3, 0.32]$ $y_\mu \in [0.1, 0.12]$ $z_\mu \in [0.6, 0.62]$	$x_\sigma \in [0.001, 0.05]$ $y_\sigma \in [0.001, 0.01]$ $z_\sigma \in [0.001, 0.03]$	$x = 0.3$ $y = 0.1$ $z = 0.6$

Table 1: Ranges of hyper-parameters and model parameters.

target location, and a fixed initial robot position. MPPI trajectory evaluations are done on the GPU, which helped overcome efficiency issues.

The purpose of the Panda task shown in Figure 2 is to reach the yellow target while avoiding obstacle collision. The obstacle has true length $x = 0.3$, width $y = 0.1$, and height $z = 0.6$. We assume partial observability for such obstacle dimension sizes and attempt to infer them as part of the transition model parameters for which we define search spaces shown in Table 1. The transition model parameter l is the rod length for Pendulum, κ_m is the mass scaling factor for all the links in Half-Cheetah, and κ_d is a damping ratio scaling factor for all components in Fetchreach. For Panda, we optimise the obstacle dimensions x, y, and z. Each transition model parameter is a random variable parameterised by ψ, for example $\psi = \{\kappa_{d,\mu}, \kappa_{d,\sigma}\}$ are damping ratio mean and damping ratio standard deviation for the Fetchreach.

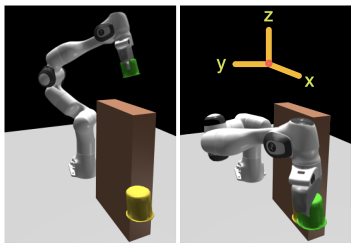

Figure 2: Panda robot setup

4.2. Method Configuration

We compare the proposed adaptive MPC framework with other surrogate-based methods used in robotics. All the compared methods are configured as follows.

Adaptive MPC framework: First, we have to determine a classifier that can deal with the stochasticity of robotic tasks. We compare two probabilistic classifiers: Random Forest (RF) with 50 decision trees denoted as BORE-RF, and Multi-layer Perceptron (MLP) classifier denoted as BORE-MLP with 2 hidden layers, each with 32 units, ReLU activations and sigmoid for the output layer. The weights were optimised for 1000 epochs using binary cross-entropy loss and ADAM optimiser with a batch size of 32. Second, we start by exploring with a $\gamma_1 = 0.5$ that decays linearly across the iterations until a reasonable final $\gamma_n = 0.05$. We compare different initial γ_1 values in Figure 3 for the Panda task. With a low γ_1, BORE could stay stuck in some local minimum, and with a higher γ_1, BORE would do more exploration first before exploiting some region. A reasonable initial value is $\gamma_1 = 0.5$, corresponding to the median, which has shown an optimal compromise according to the preliminary results in Figure 3. Finally, we set a parameter distribution p_θ with positive support

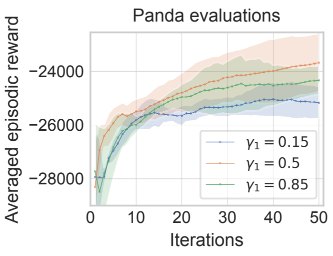

Figure 3: Some γ evaluations

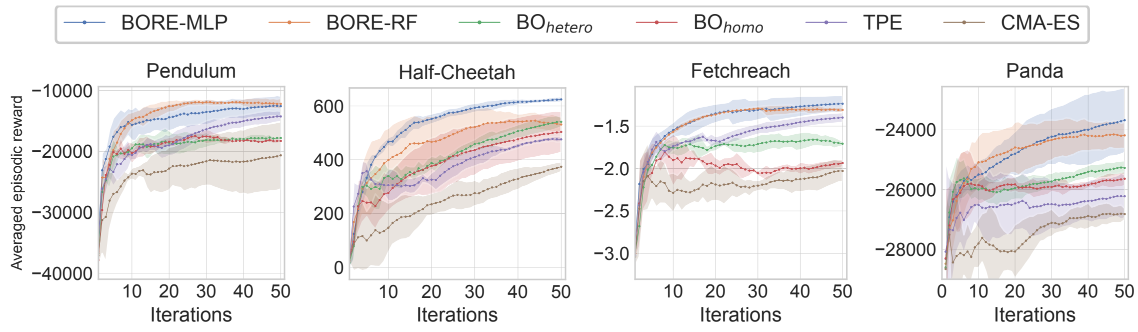

Figure 4: Averaged cumulative rewards per iteration where the shaded areas correspond to 1.5 standard deviations. Each method started at a point with minimum expected cumulative reward.

since we deal with physics variables (mass, damping ratio) and sizes. We choose the gamma distribution $\Gamma(\alpha, \beta)$, and we transform the provided mean μ and standard deviation σ via $\alpha = \frac{\mu^2}{\sigma^2}$ and $\beta = \frac{\mu}{\sigma^2}$.

BO methods: We compare the proposed method with the traditional homoscedastic B0$_{\text{homo}}$ and a heteroscedastic BO$_{\text{hetero}}$ version from Guzman et al. (2020). We collected 400 data points for the control and robotic tasks over the search spaces shown in Table 1. Then, using such data, we optimise BO's hyper-parameters via maximum GP marginal likelihood optimisation. That marginal likelihood and the acquisition function are optimised using multi-start L-BFGS-B. We used a UCB acquisition function $h_{\text{UCB}}(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \delta\sigma(\boldsymbol{x})$ with balance factor $\delta = 3.0$. For both BO, we used the anisotropic squared exponential kernel $k(\mathbf{x}, \mathbf{x}') = \sigma_n \exp\left(-1/2(\mathbf{x} - \mathbf{x}')^T \operatorname{diag}(\boldsymbol{\ell})^2(\mathbf{x} - \mathbf{x}')\right)$.

Other methods: We use TPE optimisation (Bergstra et al., 2011) with quantile value 0.5 since it is what BORE is based on, and finally, we use covariance matrix adaptation evolution strategy (CMA-ES) (Arnold and Hansen, 2010) as a non-BO baseline set with $\sigma_0 = 10$ and population size 2. CMA-ES has been widely used for hyper-parameter tuning in robotics (Modugno et al., 2016; Sharifzadeh et al., 2021).

4.3. Optimisation Assessment

To quantitatively assess optimisation performance, we report averaged cumulative rewards, defined as $\frac{1}{t'} \sum_{i=1}^{t'} g_i$ from $t' = 1$ to $t' = n$. To account for the stochasticity of evaluations, we repeat those 50 iterations 3 times. The result of repeating gives averaged cumulative rewards with their respective standard deviations. We use the cumulative rewards R to approximate g by averaging over $n_e = 10$ episodes. Each episode consists of 480 time steps for Panda and 200 time steps for the other tasks. We compare the averaged cumulative reward against the number of iterations. Figure 4 shows that BO$_{\text{homo}}$ and BO$_{\text{hetero}}$ perform similarly mainly in Pendulum because of homoscedastic noise across the search space. However, BO$_{\text{homo}}$ tends to converge to a local minimum in other tasks, which is expected since it does not account for heteroscedasticity. It is possible to achieve better or equal results with TPE, although it seems to also get stuck since it only divides observations based on the output and chooses the best next point without considering unseen regions. Both BO versions

Method	R_{max}	R_{σ}	λ	x_{μ}	x_{σ}	y_{μ}	y_{σ}	z_{μ}	z_{σ}
BORE-MLP	-21106.36	724.10	1.65	0.3104	0.0010	0.1000	0.0010	0.6000	0.0058
BORE-RF	-21891.82	291.64	1.67	0.3036	0.0431	0.1001	0.0013	0.6145	0.0234
BO$_{\text{hetero}}$	-23025.47	162.22	1.73	0.3185	0.0500	0.1200	0.0028	0.6145	0.0118
BO$_{\text{homo}}$	-22870.14	158.46	2.00	0.3200	0.0010	0.1000	0.0010	0.6200	0.0010
TPE	-23438.34	121.26	1.63	0.3124	0.0159	0.1152	0.0045	0.6047	0.0067
CMA-ES	-23779.35	481.55	1.47	0.3200	0.0010	0.1200	0.0010	0.6200	0.0142

Table 2: Maximum reward found at the last iteration for the Panda task.

are being outperformed by BORE-MLP and BORE-RF. BORE-MLP converges quite faster to an optimum in most tasks. In the PANDA environment, the difference is higher, and it suggests that the proposed framework performs better in higher-dimensional problems.

4.4. Evaluating the Optima

The previous section emphasised the proposed MPC framework and its ability to explore efficiently compared to other optimisation methods. This section describes the optima found by the methods in the Panda environment, where the improvement is more noticeable. In Table 2, we show the control hyper-parameters $\boldsymbol{\phi} = \{\lambda\}$ and the transition model parameters $\boldsymbol{\psi} = \{x_{\mu}, x_{\sigma}, y_{\mu}, y_{\sigma}, z_{\mu}, z_{\sigma}\}$ that give the maximum reward $R_{\max}$ at the last iteration after running each method for 50 iterations. R_{σ} is the observed standard deviation of the reward at the respective iteration. BORE-MLP is able to find an optimum close to the one found by BORE-RF. R_{σ} is higher for BORE-MLP as the method is still exploring new unseen regions at the end, and it can still improve its current maximum. The table also shows the optimised parameters for the distribution-based sizes: orange for the length x, green for the width y, and cyan for the height z. BORE-MLP and almost all the other methods found that considering more uncertainty in the obstacle height z would provide a higher reward, which is understandable considering that the gripper could find convenient trajectories by moving over the obstacle. The most relevant dimension size is the width y, since a wrong y would result in obstacle collision. Meanwhile, all methods can allow more uncertainty about the length of the obstacle as it does not affect the collision. Most methods converge to a similar controller hyper-parameter λ.

5. Conclusion

This paper presented an adaptive variant of model predictive control that automatically estimates model parameter distributions and tunes MPC hyper-parameters within a Bayesian optimisation framework. In contrast to previous approaches, our formulation is the first to show that global optimisation can be accomplished by learning a classifier that estimates density ratios. We studied the empirical performance of the framework with different classifiers and against benchmark BO versions. The proposed method was able to surpass the performance of the traditional BO and a heteroscedastic BO variation. Our results indicate the flexibility of using density-ratio estimation to optimise MPC and how it can impact the performance of MPC in control and robotic tasks under dynamics model uncertainty. Future research directions include obtaining theoretical results on the effects of heteroscedasticity, and we could explore alternative non-normal distributions for the actions that could be more suitable for the tasks.

References

Dirk V. Arnold and Nikolaus Hansen. Active covariance matrix adaptation for the (1+1)-CMA-ES. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO '10*, page 385, Portland, OR, 2010. ACM.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for Hyper-parameter Optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011.

Mohak Bhardwaj, Balakumar Sundaralingam, Arsalan Mousavian, Nathan D Ratliff, Dieter Fox, Fabio Ramos, and Byron Boots. Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation. In *5th Annual Conference on Robot Learning*, 2021.

Adam D. Bull. Convergence Rates of Efficient Global Optimization Algorithms. *Journal of Machine Learning Research (JMLR)*, 12:2879–2904, 2011.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. *33rd International Conference on Machine Learning, ICML 2016*, 3:2001–2014, 2016.

Daniel Görges. Relations between Model Predictive Control and Reinforcement Learning. *IFAC-PapersOnLine*, 50(1):4920–4928, 2017. ISSN 24058963. doi: 10.1016/j.ifacol.2017.08.747.

Rel Guzman, Rafael Oliveira, and Fabio Ramos. Heteroscedastic Bayesian Optimisation for Stochastic Model Predictive Control. *IEEE Robotics and Automation Letters*, 6(1):1–1, 2020. doi: 10.1109/lra.2020.3028830.

Hilbert J Kappen. Linear theory for control of nonlinear stochastic systems. *Physical review letters*, 95(20):200201, 2005.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware Dynamics Model for Generalization in Model-Based Reinforcement Learning. 2020.

Valerio Modugno, Gerard Neumann, Elmar Rueckert, Giuseppe Oriolo, Jan Peters, and Serena Ivaldi. Learning soft task priorities for control of redundant robots. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 221–226. IEEE, 2016.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, Brisbane, Australia, 2018.

Fabio Ramos, Rafael Carvalhaes Possas, and Dieter Fox. BayesSim : adaptive domain randomization via probabilistic inference for robotics simulators. In *Robotics: Science and Systems (RSS)*, Freiburg im Breisgau, Germany, 2019.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. 2006. ISBN 026218253X.

D. Romeres, G. Prando, G. Pillonetto, and A. Chiuso. On-line bayesian system identification. In *2016 European Control Conference (ECC)*, pages 1359–1364, 2016. doi: 10.1109/ECC.2016.7810478.

Nicholas Roy, Paul Newman, and Siddhartha Srinivasa. Variational bayesian optimization for runtime risk-sensitive control. 2013.

Mohammad Sharifzadeh, Yuhao Jiang, Amir Salimi Lafmejani, Kevin Nichols, and Daniel Aukes. Maneuverable gait selection for a novel fish-inspired robot using a cma-es-assisted workflow. *Bioinspiration & Biomimetics*, 16(5):056017, 2021.

Farshud Sorourifar, Georgios Makrygirgos, Ali Mesbah, and Joel A Paulson. A data-driven automatic tuning method for mpc under uncertainty using constrained bayesian optimization. *IFAC-PapersOnLine*, 54(3):243–250, 2021.

Louis C Tiao, Aaron Klein, Cédric Archambeau, Edwin V Bonilla, Matthias Seeger, and Fabio Ramos. Bayesian optimization by density ratio estimation. In *Proceedings of the 38th International Conference on Machine Learning (ICML2021), Virtual (Online)*, 2021.

Zilong Wang and Marianthi Ierapetritou. A novel surrogate-based optimization method for black-box simulation with heteroscedastic noise. *Industrial & Engineering Chemistry Research*, 56 (38):10720–10732, 2017. doi: 10.1021/acs.iecr.7b00867.

Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Aggressive driving with model predictive path integral control. *Proceedings - IEEE International Conference on Robotics and Automation*, 2016-June:1433–1440, 2016. ISSN 10504729. doi: 10.1109/ICRA.2016.7487277.

Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and IEvangelos A. Theodorou. Information-Theoretic Model Predictive Control: Theory and Applications to Autonomous Driving. *IEEE Transactions on Robotics*, 34(6):1603–1622, 2018a. ISSN 15523098. doi: 10.1109/TRO.2018.2865891.

Grady Williams, Brian Goldfain, Paul Drews, Kamil Saigol, James Rehg, and Evangelos Theodorou. Robust Sampling Based Model Predictive Control with Sparse Objective Information. (c), 2018b. doi: 10.15607/rss.2018.xiv.042.