12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

Efficient Performance-based MPC Tuning in High Dimensions using Bayesian Optimization over Sparse Subspaces

Akshay Kudva * **Melanie T. Huynh** ** **Ali Mesbah** **
Joel A. Paulson *

* *Department of Chemical and Biomolecular Engineering, The Ohio State University, Columbus, 43210, OH, USA*
** *Department of Chemical and Biomolecular Engineering, University of California, Berkeley, Berkeley, CA 94720*

Abstract: Model predictive control (MPC) is one of the most effective technologies for optimal control of constrained multivariable systems. The closed-loop performance of MPC, however, can be sensitive to the choice of several tuning parameters that can appear in the prediction model, constraints, and/or cost function. Due to inherent limitations of manual tuning and the performance function depends on these parameters in an unknown manner, there has been increasing interest in "auto-tuning" using derivative-free optimization (DFO) methods. Bayesian optimization (BO) is a particularly powerful framework for data-efficient DFO of noisy, black-box functions Several recent works have shown the effectiveness of BO for MPC tuning when the number of tuning parameters is relatively small; however, MPC problems often involve a much larger number of parameters for which BO tends to struggle. In this paper, we propose to exploit a new type of Gaussian process surrogate model defined on sparse axis-aligned subspaces to mitigate the curse of dimensionality in BO. The approach is effective when closed-loop performance is sensitive to a small subset of tuning parameters, which is often the case in task-specific tuning problems. We demonstrate an order-of-magnitude performance improvement can be obtained with the proposed method compared to standard BO on a benchmark inverted pendulum on a cart problem controlled by MPC with twenty independent tuning parameters.

Keywords: Controller tuning; High-dimensional Gaussian processes; Bayesian optimization.

1. INTRODUCTION

Model predictive control (MPC) is a powerful technology for advanced control of constrained multivariable systems (Rawlings et al., 2017). MPC has been successfully applied in a wide-variety of complex engineering applications, including robotic, aerospace, automotive, biomedical, sustainability, and chemical process systems (Schwenzer et al., 2021). The actual closed-loop performance achieved in practice by MPC is known to strongly depend on several factors, including the quality of the dynamic prediction model, the choice of objective and constraint functions, the choice of real-time optimization routine, and the selection of any free "hyperparameters" that may appear in these functions. For example, it is common to identify a prediction model using limited open-loop data collected around a single operating condition. The uncertainty in the identified nominal model, however, can lead to overly aggressive or conservative responses, resulting in reduced closed-loop performance. This is because the model that produces the smallest output prediction errors may not be the one that provides the best system performance when operated in a closed-loop fashion (Piga et al., 2019). This argument, often referred to as identification for control (I4C) (Gevers, 2005), can be extended to the other components of MPC such that the goal is to select/design/tune the model, optimization formulation, and solution method simultaneously to produce the best closed-loop performance while meeting computation constraints.

The need for sample-efficient MPC tuning has recently led to the use of Bayesian optimization (BO), which is a powerful black-box optimization framework designed for noisy, expensive-to-evaluate functions (Frazier, 2018). Recent contributions have demonstrated substantial performance gains with BO over alternative black-box optimization algorithms (Paulson et al., 2023b). Furthermore, BO can be straightforwardly adapted to handle many variations of the tuning problem, including those with mixed continuous-discrete search spaces (Paulson and Mesbah, 2020), multiple performance objectives (Makrygiorgos et al., 2022), safety constraints (Sorourifar et al., 2021; Paulson et al., 2023a), and approximate gradient information (Makrygiorgos et al., 2023).

While BO has shown great promise on low-dimensional tuning problems (fewer than ten dimensions), its application to high-dimensional problems remains a significant challenge. Realistic forms of MPC tuning problems often involve a large number of parameters, as they can appear in any component of the algorithm. This challenge is cir-

⋆ This work was supported by the National Science Foundation (NSF) under award number 2237616.

cumvented in previous work by assuming sufficient prior knowledge is available to *a priori* identify a reasonably small set of key parameters before running the optimization procedure. This type of assumption is likely not valid in the early phases of tuning wherein we can easily have many tens to hundreds of parameters whose impact on performance is unknown. The challenge with BO in such cases can be traced back to the difficulty of building well-calibrated surrogate models in high dimensions. BO leverages a probabilistic surrogate model to decide the optimal location of the next sample, meaning it will perform poorly if the model class is too flexible or too rigid.

Gaussian process (GP) surrogate models defined on sparse axis-aligned subspaces (SAAS) are an example of such a model class that strikes a useful balance between flexibility and parsimony in the low-data regime (Eriksson and Jankowiak, 2021). In this paper, we propose to use SAAS-GPs to tackle high-dimensional MPC tuning problems. The key assumption underpinning SAAS-GP models is that only a small number of features have a strong impact on the objective function, which we conjecture holds in many task-specific MPC tuning problems. We discuss how inference (i.e., posterior prediction) can be efficiently carried out using Hamiltonian Monte Carlo (HMC). HMC enables fast identification of a sparse subspace of tuning parameters most relevant to modeling the closed-loop performance function. The proposed approach is validated on a challenging benchmark problem wherein an inverted pendulum on a cart is controlled using hierarchical MPC with more than twenty tuning parameters. Our results show that BO using SAAS-GP can discover parameters leading to an order-of-magnitude improvement in cost compared to BO with standard GP and random search.

The remainder of this paper is organized as follows. The MPC tuning problem of interest in this work is defined in Section 2. The SAAS-GP method and its integration within the BO framework is summarized in Section 3. Section 4 illustrates the performance of the proposed method on the benchmark problem. We then conclude the paper and describe some possible directions for future work in Section 5.

2. PROBLEM FORMULATION

In this work, we are interested in closed-loop MPC tuning problems of the form

$$\boldsymbol{\theta}^\star \in \underset{\boldsymbol{\theta}\in\mathcal{D}}{\operatorname{argmin}} \; J(\boldsymbol{\theta}), \tag{1}$$

where $\boldsymbol{\theta} \in \mathcal{D} \subset \mathbb{R}^D$ denotes the set of tuning parameters that are restricted to a space $\mathcal{D}$ and $J : \mathcal{D} \to \mathbb{R}$ is scalar objective function derived from some simulator or experiment in closed loop with the MPC controller. For simplicity, we assume that the parameter space has been scaled to the D-dimensional unit cube $\mathcal{D} = [0, 1]^D$. As discussed in detail in (Paulson et al., 2023b), the closed-loop performance function is generally defined according to an expected cost of the form

$$J(\boldsymbol{\theta}) = \mathbb{E}\{\psi(\boldsymbol{X}(\boldsymbol{\theta}), \boldsymbol{U}(\boldsymbol{\theta}), \boldsymbol{W}(\boldsymbol{\theta}))\}, \tag{2}$$

where $\boldsymbol{X}(\boldsymbol{\theta})$, $\boldsymbol{U}(\boldsymbol{\theta})$, and $\boldsymbol{W}(\boldsymbol{\theta})$ denote the state, control input, and disturbance trajectories over some finite horizon (that are completely parametrized by the tuning parameters), $\psi(\cdot)$ is a function that maps these trajectory realizations to a performance measure, and $\mathbb{E}\{\cdot\}$ denotes the expectation over some probability distribution for the initial state values and disturbance sequence. Typically, these closed-loop sequences are generated by a stochastic nonlinear dynamic process such as

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{w}_t), \tag{3a}$$
$$\boldsymbol{u}_t = \boldsymbol{\pi}_t(\boldsymbol{x}_t; \boldsymbol{\theta}), \tag{3b}$$
$$\boldsymbol{w}_t \sim p(\boldsymbol{w}_t \mid \boldsymbol{x}_t, \boldsymbol{u}_t), \tag{3c}$$

where $\boldsymbol{x}_t$ is the system state, $\boldsymbol{u}_t$ is the control input, $\boldsymbol{w}_t$ is the disturbance, $\boldsymbol{f}(\cdot)$ is the state transition function, $\boldsymbol{\pi}_t$ is the MPC policy fully parametrized by tuning parameters $\boldsymbol{\theta}$, and $p(\cdot \mid \boldsymbol{x}_t, \boldsymbol{u}_t)$ denotes the probability distribution for the disturbance at time t. In general, we cannot compute the expectation in (2) exactly and must resort to some approximation procedure such as Monte Carlo (MC) integration. Although in principle the proposed framework can handle noisy observations introduced by MC, the purpose of this paper is investigate the impact of increasing dimensionality on MPC tuning. As such, we assume that noise-free evaluations of J can be obtained by running K independent closed-loop simulations, i.e.,

$$J(\boldsymbol{\theta}) = \frac{1}{K}\sum_{i=1}^{K} \psi(\boldsymbol{X}^i(\boldsymbol{\theta}), \boldsymbol{U}^i(\boldsymbol{\theta}), \boldsymbol{W}^i(\boldsymbol{\theta})), \tag{4}$$

where $\{\boldsymbol{X}^i, \boldsymbol{U}^i, \boldsymbol{W}^i\}_{i=1}^K$ denote the closed-loop trajectories generated by (3). This is equivalent to assuming that the probability distribution of the disturbance sequence is discrete with K unique realizations.

We are interested in the case that evaluations of J are expensive, meaning we are limited to at most a few hundred evaluations in total. This situation is quite common in MPC tuning applications since MPC is a particularly expensive controller to evaluate as it requires the solution to an optimization problem at every sampling time instance. This issue is only further compounded when $\boldsymbol{f}$ is defined in terms of a high-fidelity simulator or experimental system and/or the number of disturbance realizations K is large.

3. PERFORMANCE-BASED MPC TUNING USING SPARSE AXIS-ALIGNED BAYESIAN OPTIMIZATION

In this section, we describe the high-dimensional BO strategy that can be used to solve (1) when D is large. We start with a brief review of traditional GP models and then describe a strategy to effectively extend GPs to high-dimensional problems when the dimensions of $\boldsymbol{\theta} \in \mathcal{D}$ exhibit a hierarchy of relevance for prediction of J (i.e., a subset of dimensions are the most sensitive to the outcome). We then describe the data acquisition strategy by which new tuning parameters can be queried sequentially by averaging posterior samples generated from the high-dimensional GP.

3.1 Standard Gaussian Processes

Gaussian processes (GPs) are an increasingly popular method for non-parametric regression where the goal is to find an approximation to the unknown mapping J that provides built-in uncertainty quantification for predictions $J(\boldsymbol{\theta}_\star)$ where $\boldsymbol{\theta}_\star \in \mathcal{D}$ denotes an arbitrary test input. GPs achieve this goal by assuming that the function values

at different inputs are random variables, with any finite subset of these random variables having a joint Gaussian distribution (Williams and Rasmussen, 2006). As such, a GP defined over input space $\mathcal{D}$ is fully specified by a prior mean function $\mu^\gamma : \mathcal{D} \to \mathbb{R}$ and a prior covariance function (or kernel) $k^\gamma : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ where γ denotes the collective set of hyperparameters that define the GP prior. Without loss of generality, we assume the mean function is uniformly zero, which can be achieved in practice with appropriate data normalization (Paulson and Lu, 2022).

In this work, we focus exclusively on the squared exponential kernel of the form

$$k^\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') = \zeta^2 \exp\left(-\tfrac{1}{2}\textstyle\sum_{i=1}^{D} \rho_i(\theta_i - \theta_i')^2\right), \qquad (5)$$

where θ_i denotes the i^{th} dimension of $\boldsymbol{\theta}$, $\{\rho_i\}_{i=1}^D$ are the inverse lengthscales, and ζ^2 is a scaling factor for the output variance. The hyperparameters then correspond to $\gamma = \{\rho_1, \ldots, \rho_D, \zeta^2\}$, which have intuitive interpretations. The inverse lengthscale ρ_i specifies how quickly the covariance changes between neighboring values along the dimension θ_i. Small values for ρ_i imply the covariance decays slowly in θ_i such that this dimension contributes little to variation in J (prediction is flat along θ_i). On the other hand, a large value for ρ_i implies the covariance decays quickly in the direction of θ_i such that J can strongly vary along this dimension (prediction can be highly nonlinear along θ_i). Lastly, the prior variance ζ^2 roughly represents that expected magnitude of the function value, meaning $|J(\boldsymbol{\theta})| \leq \zeta$ for $\boldsymbol{\theta} \in \mathcal{D}$ with around 68% probability.

Given N observations of the performance function $\mathbf{y} \in \mathbb{R}^N$ at N specific input values $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_i\}_{i=1}^N$, we can condition on these observations, which leads to a posterior normal distribution $\mathcal{N}(\mu_N(\boldsymbol{\theta}_\star|\gamma), \sigma_N^2(\boldsymbol{\theta}_\star|\gamma))$ at query point $\boldsymbol{\theta}_\star \in \mathcal{D}$ with mean and variance functions given by

$$\mu_N(\boldsymbol{\theta}_\star|\gamma) = {k^\gamma_{\star,\boldsymbol{\Theta}}}^\top \left(K^\gamma_{\boldsymbol{\Theta},\boldsymbol{\Theta}} + \sigma^2 I_N\right)^{-1}\mathbf{y}, \qquad (6a)$$

$$\sigma_N^2(\boldsymbol{\theta}_\star|\gamma) = k^\gamma_{\star,\star} - {k^\gamma_{\star,\boldsymbol{\Theta}}}^\top \left(K^\gamma_{\boldsymbol{\Theta},\boldsymbol{\Theta}} + \sigma^2 I_N\right)^{-1} k^\gamma_{\star,\boldsymbol{\Theta}}, \qquad (6b)$$

where $k^\gamma_{\star,\star} = k^\gamma(\boldsymbol{\theta}_\star, \boldsymbol{\theta}_\star)$, $K^\gamma_{\boldsymbol{\Theta},\boldsymbol{\Theta}}$ is the $N \times N$ kernel matrix with elements $[K^\gamma_{\boldsymbol{\Theta},\boldsymbol{\Theta}}]_{(i,j)} = k^\gamma(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, $k^\gamma_{\star,\boldsymbol{\Theta}} = [k^\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}_1), \ldots, k^\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}_N)]^\top$ is a column vector of the kernel evaluated between the query point $\boldsymbol{\theta}_\star$ and the observed inputs $\boldsymbol{\Theta}$, I_N is the $N \times N$ identity matrix, and σ^2 is the variance of a Gaussian noise term. Since we are interested in noise-free evaluations of J, we set σ^2 to a small constant just to improve the numerical stability.

3.2 Sparse Axis-Aligned Subspace (SAAS) Function Prior

A key challenge one has to deal with in GP modeling is the choice of hyperparameters γ. Standard GPs are fit using maximum likelihood estimation (MLE) (Williams and Rasmussen, 2006, Section 2.7), however, since there is no mechanism to regularize the lengthscales, this approach generally produces to non-negligible ρ_i for all $i = 1, \ldots, D$, leading to significant overfitting in the high-dimensional case. One way to overcome this challenge is to take a fully Bayesian perspective of the hyperparameters such that γ also has a prior that can also be conditioned on the available data when performing inference tasks. To be useful in practice, the prior must have a strong sparsifying effect so we can hope to identify a reasonable model for J given limited data. We propose to use the following SAAS function prior that creates a sparse structure on the inversed lengthscales (Eriksson and Jankowiak, 2021):

1. Kernel variance: $\zeta^2 \sim \mathcal{LN}(0, 10^2)$;
2. Global shrinkage: $\tau \sim \mathcal{HC}(\alpha)$;
3. Lengthscales: $\rho_i \sim \mathcal{HC}(\tau)$, $\forall i = 1, \ldots, D$;
4. Function values: $\mathbf{J} \sim \mathcal{N}(\mathbf{0}, K^\gamma_{\boldsymbol{\Theta},\boldsymbol{\Theta}})$;
5. Observations: $\mathbf{y} = \mathcal{N}(\mathbf{J}, \sigma^2 I_N)$;

where $\mathcal{LN}$ denotes the log-normal distribution and $\mathcal{HC}(s)$ is the half-Cauchy distribution with scale parameter s. The value $\alpha > 0$ is a hyperparameter that controls the level of shrinkage in the model – we use the suggested default value of $\alpha = 0.1$. The half-Cauchy prior for τ implies $p(\tau|\alpha) \propto (\tau^2 + \alpha^2)^{-1}\mathbf{1}(\tau > 0)$ where $\mathbf{1}(\tau > 0)$ is the indicator function equal to 1 if $\tau > 0$ and 0 otherwise. Thus, the prior enforces that the density of τ concentrates near zero such that, based on $\{\rho_i\}_{i=1}^D$ also having half-Cauchy priors, we expect most dimensions to be "turned off" (unimportant for prediction) initially. As observations $\mathbf{y}$ are gathered, the posterior for τ can be pushed to higher values (as long as sufficient evidence is available), which enables the posterior density of more ρ_i values to increase. Therefore, the SAAS prior is sparse by nature (must have strong evidence to unlock any given dimension θ_i in the model) and adaptive in the sense that more dimensions can be unlocked as enough evidence is gathered to push the posterior for τ to larger values.

We conjecture that the SAAS prior is particularly useful in task-specific MPC tuning contexts (1), as it is unlikely that all considered tuning parameters contribute equally to closed-loop performance. This conjecture is supported by the case study results presented in Section 4. Since it is often non-obvious which parameters are important, SAAS gracefully addresses this issue by automatically identifying important parameters from relatively small observation sets. Even though one could use intuition to select a subset of $\boldsymbol{\theta}$ to consider in (1), unless chosen very precisely, this approach is subject to significant performance losses due to neglecting an unknowingly important parameter perceived to be unimportant by the user. It is worth noting, however, that in cases that we want the derived tuning parameters to generalize between different tasks (e.g., results in high-performance across many different initial state and/or setpoint values), this conjecture may not be valid and we may not achieve the same level of performance. We argue that this is not a fundamental limitation of the method but more that a tuning problem of this type (one with a large number of sensitive parameters) is extremely difficult to solve. Such cases are likely to require large simulation budgets and potentially new algorithms, which we plan to explore more in our future work.

3.3 The SAASBO Algorithm

The main challenge with the SAAS-GP model defined in Section 3.2 is the inference step needed to compute the predictive posterior distribution $p(J(\boldsymbol{\theta}_\star)|\boldsymbol{\theta}_\star, \mathbf{y}, \boldsymbol{\Theta})$, which is no longer Gaussian and must be calculated by marginalizing over the posterior distribution $p(\gamma, \tau|\mathbf{y}, \boldsymbol{\Theta})$. Since the marginalization cannot be done analytically, we resort to generating samples from $p(\gamma, \tau|\mathbf{y}, \boldsymbol{\Theta})$. The No-U-Turn sampler (NUTS) (Hoffman et al., 2014) is an efficient

variant of Hamiltonian Monte Carlo (HMC) that exploits the fact that the SAAS-GP model has a differentiable joint density function. NUTS is then capable of targeting the un-normalized joint density

$$p(\mathbf{y}|\boldsymbol{\Theta},\gamma)p(\gamma|\tau)p(\tau) \propto p(\gamma,\tau|\mathbf{y},\boldsymbol{\Theta}), \tag{7}$$

where the marginal likelihood of the observed data given fixed hyperparameters can be computed in closed form, i.e., $p(\mathbf{y}|\boldsymbol{\Theta},\gamma) = \mathcal{N}(\mathbf{y}, K^{\gamma}_{\boldsymbol{\Theta},\boldsymbol{\Theta}} + \sigma^2 I_N)$. By running the NUTS algorithm, we can obtain approximate posterior samples for the kernel hyperparameters at a cost of $O(N^3D)$ per sample for N datapoints and D dimensions.

To extend this to the BO setting, suppose that we have collected a data history of $\mathcal{H}_n = \{\boldsymbol{\theta}_{1:N}, y_{1:N}\}$ in the past n steps. Our goal is then to use $\mathcal{H}_n$ to decide the next query point $\boldsymbol{\theta}_{n+1}$ using some acquisition (or expected utility) function $\alpha_n : \mathcal{D} \to \mathbb{R}$ that quantifies the potential benefit of evaluating $J(\boldsymbol{\theta})$ at any unseen $\boldsymbol{\theta} \in \mathcal{D}$. BO can then be interpreted as the sequential learning process defined by

$$\boldsymbol{\theta}_{n+1} \in \operatorname*{argmax}_{\boldsymbol{\theta}\in\mathcal{D}} \alpha_n(\boldsymbol{\theta}). \tag{8}$$

Several acquisition functions have been proposed in the literature including, for example, expected improvement (EI) and lower confidence bound (LCB), which both have analytic expressions for fixed hyperparameter values γ. For example, EI is defined as follows (Jones et al., 1998)

$$\mathrm{EI}_n(\boldsymbol{\theta}|J_n^\star,\gamma) = \varphi(J_n^\star - \mu_n(\boldsymbol{\theta}|\gamma), \sigma_n(\boldsymbol{\theta}|\gamma)), \tag{9}$$

where

$$\varphi(y,s) = \begin{cases} y\Phi(y/s) + s\phi(y/s), & s > 0, \\ \max(y,0), & s = 0, \end{cases}$$

$\Phi(\cdot)$ and $\phi(\cdot)$ denote the standard normal cumulative density function and probability density function, respectively, and $J_n^\star = \min_n y_n$ denotes the incumbent solution that refers to the best function evaluation observed so far. Since γ is a latent variable in the SAAS-GP context, the acquisition function is computed by averaging over the posterior samples $\{\gamma_n^{(l)}\}_{l=1}^L \sim p(\gamma|\mathcal{H}_n)$

$$\alpha_n(\boldsymbol{\theta}) = \frac{1}{L}\sum_{l=1}^{L} \mathrm{EI}_n(\boldsymbol{\theta}|J_n^\star, \gamma_n^{(l)}) \tag{10}$$

where L is the number of posterior samples obtained with NUTS. Since the GP prediction model (6) and the EI function are differentiable with respect to $\boldsymbol{\theta}$, they can be efficiently optimized using gradient-based methods (see (Balandat et al., 2020, Appendix F) for further details).

4. CASE STUDY: AUTOMATED MPC TUNING IN A HIGH-DIMENSIONAL SPACE

4.1 System Description

We are interested in MPC tuning for an inverted pendulum on a cart problem whose dynamics are governed by the standard equations of motion. We use the same problem setup as described in (Piga et al., 2019) (same dynamics, parameters, and constraints). The key outputs of interest are the position of the cart p and the pendulum angle with respect to the upright vertical position ϕ. The force acting on the cart F is the only manipulatable input in the problem. The output measurements and the input force are all assumed to be corrupted with a relatively small amount of independent Gaussian noise.

Table 1. Overview of tuning parameters and their corresponding minimum and maximum values for the hierarchical MPC system.

	Min.	Max.
PID controller gains		
$\theta_1, \theta_2, \theta_3$	-600	0
MPC state-space parameters, see (13)		
$\theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9$	-600	600
θ_{10}, θ_{12}	-2	1
θ_{11}, θ_{13}	-2	0
θ_{14}	-1	1
θ_{15}	-2	10
MPC prediction horizon (integer)		
θ_{16}	10	20
Initial state values		
$\theta_{17} = p(0)$	-0.15	0.5
$\theta_{18} = \dot{p}(0)$	-0.15	0.5
$\theta_{19} = \phi(0)$	$\frac{\pi}{5} - 0.1$	$\frac{\pi}{5} + 0.3$
$\theta_{20} = \dot{\phi}(0)$	-2	2

4.2 Hierarchical MPC Design

We also use the hierarchical MPC control structure from (Piga et al., 2019), which consists of two main parts. An inner controller $\mathcal{K}$ is used to control the fast dynamics of the system (the angle ϕ in this case) while an outer MPC is used to improve performance and enforce constraints on the cart position p and the input force F. A complete list of the 20 tuning parameters for this control system is given in Table 1. The details of how these parameters enter the controller are briefly discussed below. Note that, for implementation purposes, all parameters are scaled by their bounds to be within the unit cube $[0,1]^{20}$.

Here, $\mathcal{K}$ takes the form of a discrete-time proportional-integral-derivative (PID) controller with three parameters $\{\theta_1, \theta_2, \theta_3\}$ corresponding to the gains of the P, I, and D terms. This controller sets the applied force to the cart, i.e., $u = F = \mathcal{K}(\boldsymbol{\theta})(g - \phi)$. The MPC controller, on the other hand, has several additional tuning parameters. In particular, it uses a parametrized continuous-time linear state-space model to predict the inner loop dynamics

$$\dot{\boldsymbol{\xi}}_M = \boldsymbol{A}_M(\boldsymbol{\theta})\boldsymbol{\xi}_M + \boldsymbol{B}_M(\boldsymbol{\theta})g, \tag{11}$$

where $\boldsymbol{\xi}_M \in \mathbb{R}^2$ is the state vector, $g \in \mathbb{R}$ is the MPC command that corresponds to the setpoint of the inner loop, and $\boldsymbol{A}_M(\boldsymbol{\theta}) \in \mathbb{R}^{2\times 2}$ and $\boldsymbol{B}_M(\boldsymbol{\theta}) \in \mathbb{R}^2$ are system matrices that define the output prediction model. The state vector is assumed to equal the model outputs $\boldsymbol{y}_M = \boldsymbol{\xi}_M$, leading to the following MPC prediction model

$$\begin{bmatrix} u_M \\ \boldsymbol{y}_M \end{bmatrix} = \begin{bmatrix} \mathcal{K}(\boldsymbol{\theta})(1 - [0,1]M_y(\boldsymbol{\theta})) \\ M_y(\boldsymbol{\theta}) \end{bmatrix} g, \tag{12}$$

where $M_y(\boldsymbol{\theta})$ denotes the prediction model from the MPC command g to the outputs $\boldsymbol{y} = (p, \phi)$.

Although $\boldsymbol{A}_M(\boldsymbol{\theta})$ and $\boldsymbol{B}_M(\boldsymbol{\theta})$ only have six independent dimensions in total, we double the number of parameters considered to increase the complexity of the problem

$$\boldsymbol{A}_M(\boldsymbol{\theta}) = \begin{bmatrix} \theta_4 & \theta_5 \\ \theta_6 & \theta_7 \end{bmatrix} + \begin{bmatrix} 0.1\theta_{10}^2 & 0.01\theta_{11}^2 \\ 0.1\theta_{12}^2 & 0.01\theta_{13}^2 \end{bmatrix}, \tag{13}$$

$$\boldsymbol{B}_M(\boldsymbol{\theta}) = \begin{bmatrix} \theta_8 \\ \theta_9 \end{bmatrix} + \begin{bmatrix} 0.01\theta_{14}^2 \\ 0.01\theta_{15}^2 \end{bmatrix}. \tag{14}$$

Notice that $\theta_{10}, \ldots, \theta_{15}$ are redundant tuning parameters that cannot be uniquely identified in general – they have

been intentionally added to understand the impact of such unimportant variables on the BO search process. Lastly, we also treat the prediction horizon θ_{16} and the initial states as tuning parameters, denoted by $\theta_{17}, \ldots, \theta_{20}$. A detailed discussion on the exact form of the MPC optimization problem and the choice of sampling time and weight parameters are given in (Piga et al., 2019).

4.3 Closed-loop Performance Cost

The closed-loop performance cost is assumed to be of the form (2) where the function $\psi(\cdot)$ is defined as follows

$$\psi(\boldsymbol{X}(\boldsymbol{\theta}), \boldsymbol{U}(\boldsymbol{\theta}), \boldsymbol{W}(\boldsymbol{\theta})) = \log\left[\frac{1}{T}\sum_{t=1}^{T} b(p(t)) + 1\right] \quad (15)$$
$$+ \log\left[\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{10}|r_p - p(t)|\right) + \frac{9}{10}|r_\phi - \phi(t)|\right],$$

where

$$b(p) = \begin{cases} 10(|p| - 1), & \text{if } |p| > 1, \\ 0, & \text{otherwise,} \end{cases}$$

is a barrier function accounting for violation of constraints on the cart position, T is the final time of the closed-loop experiment (equal to 10 seconds), $t \in \{1, \ldots, T\}$ denotes the discrete time index taken at 5 millisecond sampling times, and $r_p = r_\phi = 0$ are the reference values for the outputs of interest (which captures the engineering objectives of controlling the angle ϕ to 0 so the pole remains upright while limiting the horizontal displacement). The cost J is then estimated as an average over 10 disturbance realizations using (4).

4.4 Results and Discussion

We compare SAASBO to two baseline methods: EI computed using a standard GP prior fit using MLE hyperparameter estimation and Random Search (RS). All methods are provided 10 initial random samples and are given an additional budget of 35 expensive evaluations. The results of the best observed closed-loop performance over the 45 total iterations for all three methods are shown in Fig. 1. Since the objective function (15) is defined in terms of logarithms, we plot the results in terms of a scaled objective $\tilde{J}(\boldsymbol{\theta}) = e^{J(\boldsymbol{\theta})}$ to better visually show the differences in performance. We see SAASBO finds tuning parameter values that lead to at least an order of magnitude improvement in $\tilde{J}(\boldsymbol{\theta})$ compared to standard BO and RS. This can also be seen in the closed-loop system trajectories for the best found $\boldsymbol{\theta}$ values with each method (Fig. 2). The SAASBO recommended tuning parameters $\boldsymbol{\theta}^\star_{\text{SAASBO}}$ were the only ones that accomplish all the desired control objectives.

To gain insight into why SAASBO outperforms standard BO in this case, we investigate the quality of the SAAS-GP model trained on closed-loop performance values obtained at 90 randomly sampled $\boldsymbol{\theta}$ values across $\mathcal{D}$. Specifically, we look at the magnitude of the lengthscale values identified using NUTS averaged over a large number of posterior samples (see Section 3 for details). The results are shown in Fig. 3 wherein we see that θ_4 and θ_2 are found to be the most important parameters that impact closed-loop performance. θ_2, which corresponds to the integral

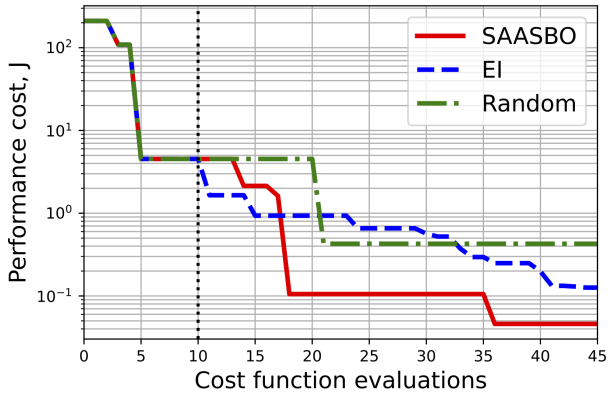


Fig. 1. Best found (transformed) closed-loop performance cost $\tilde{J}(\boldsymbol{\theta}) = e^{J(\boldsymbol{\theta})}$ using SAASBO, EI, and Random Search as a function of the number of evaluations. The first 10 samples represent the initialization points, which are drawn uniformly at random from $\mathcal{D}$.

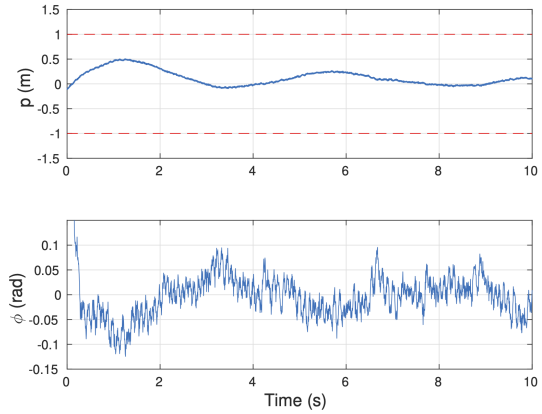
gain, would be expected to be very important since poor selections can destabilize the closed-loop system. θ_4 corresponds to the $(1, 1)$ element of the $\boldsymbol{A}_M$ matrix (part of the internal MPC model). It is not obvious that θ_4 would have such a strong impact on performance, however, the SAAS prior is surprisingly able to identify the behavior from a small number of evaluations. Several other parameters emerge in the next tier of importance, so this problem does not have obvious unimportant dimensions. However, by exploiting the clearly dominant dimensions, we can identify much better $\boldsymbol{\theta}$ values with substantially less resources (expensive closed-loop simulations in this case), which highlights the value that the SAAS perspective can provide in MPC tuning contexts.

5. CONCLUSIONS

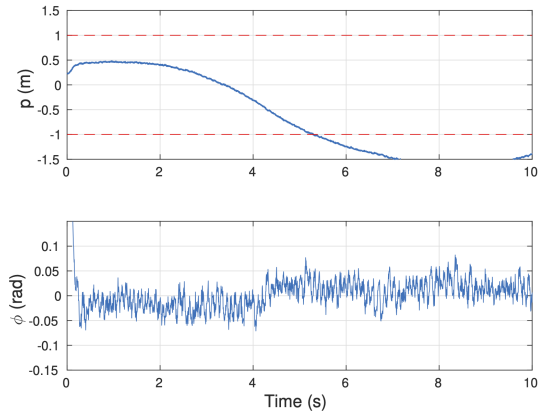
In this work, we presented a high-dimensional Bayesian optimization (BO) framework for practical (sample-efficient) MPC tuning with tens to hundreds of parameters. The key observation is that standard BO, which uses Gaussian process (GP) surrogate models to plan the next query point, is prone to over-exploration when a high-dimensional GP is fit using classical maximum likelihood estimation. As such, we propose to replace standard GP priors with a sparse axis-aligned subspace (SAAS) prior. The SAAS prior is effective when the objective function depends on a subset of (unknown) dominant dimensions. We argue that task-specific MPC tuning is a class of real-world problems that fit this assumption. We investigate this claim empirically on a challenging benchmark problem related to an inverted pendulum on a cart system controlled by a hierarchical MPC strategy with twenty tuning parameters. Not only do we observe significant performance gains with SAASBO, we also show how useful insights into the problem structure can be automatically discovered with SAAS.

REFERENCES

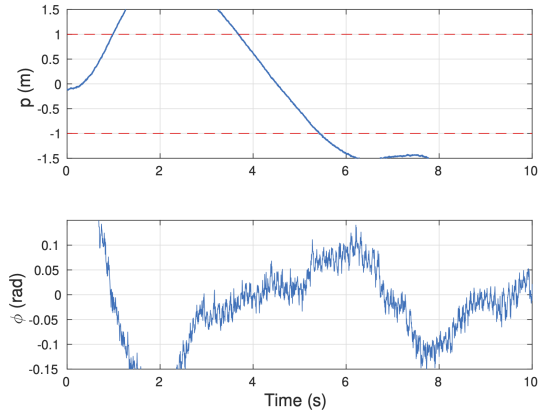
Balandat, M., Karrer, B., Jiang, D., Daulton, S., Letham, B., Wilson, A.G., and Bakshy, E. (2020). BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. *Advances in neural information processing systems*, 33, 21524–21538.



(a) SAASBO



(b) Expected Improvement (EI)



(c) Random Search (RS)

Fig. 2. Closed-loop simulations of inverted pendulum on cart system with the best MPC controller parameters discovered by (a) SAASBO, (b) EI, and (c) RS.

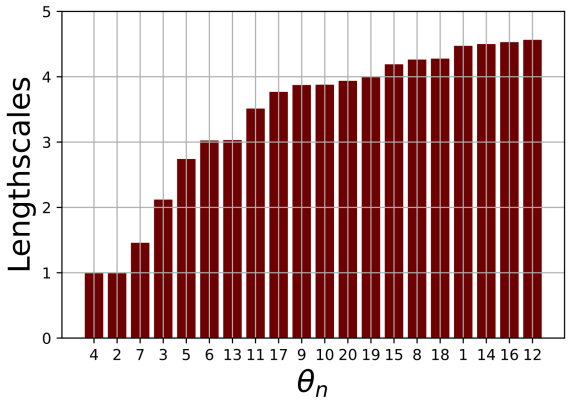


Fig. 3. MPC tuning parameters ranked using SAAS-GP lengthscales with 90 closed-loop cost datapoints.

Eriksson, D. and Jankowiak, M. (2021). High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In *UAI*, 493–503.

Frazier, P.I. (2018). A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Gevers, M. (2005). Identification for control: From the early achievements to the revival of experiment design. *European Journal of Control*, 11(4-5), 335–352.

Hoffman, M.D., Gelman, A., et al. (2014). The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1), 1593–1623.

Jones, D.R., Schonlau, M., and Welch, W.J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4), 455.

Makrygiorgos, G., Bonzanini, A.D., Miller, V., and Mesbah, A. (2022). Performance-oriented model learning for control via multi-objective Bayesian optimization. *Computers & Chemical Engineering*, 162, 107770.

Makrygiorgos, G., Paulson, J.A., and Mesbah, A. (2023). Gradient-enhanced Bayesian optimization via acquisition ensembles with application to reinforcement learning. *IFAC-PapersOnLine*, 56(2), 638–643.

Paulson, J., Sorourifar, F., Laughman, C., and Chakrabarty, A. (2023a). Self-optimizing vapor compression cycles online with Bayesian optimization under local search region constraints. *Journal of Dynamic Systems, Measurement, and Control*, 1–14.

Paulson, J.A. and Lu, C. (2022). COBALT: COnstrained Bayesian optimizAtion of computationaLly expensive grey-box models exploiting derivaTive information. *Computers & Chemical Engineering*, 160, 107700.

Paulson, J.A. and Mesbah, A. (2020). Data-driven scenario optimization for automated controller tuning with probabilistic performance guarantees. *IEEE Control Systems Letters*, 5(4), 1477–1482.

Paulson, J.A., Sorourifar, F., and Mesbah, A. (2023b). A tutorial on derivative-free policy learning methods for interpretable controller representations. In *Proceedings of the American Control Conference*, 1295–1306.

Piga, D., Forgione, M., Formentin, S., and Bemporad, A. (2019). Performance-oriented model learning for data-driven mpc design. *IEEE Control Systems Letters*, 3(3), 577–582.

Rawlings, J.B., Mayne, D.Q., and Diehl, M. (2017). *Model predictive control: Theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI.

Schwenzer, M., Ay, M., Bergs, T., and Abel, D. (2021). Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6), 1327–1349.

Sorourifar, F., Makrygirgos, G., Mesbah, A., and Paulson, J.A. (2021). A data-driven automatic tuning method for mpc under uncertainty using constrained bayesian optimization. *IFAC-PapersOnLine*, 54, 243–250.

Williams, C.K. and Rasmussen, C.E. (2006). *Gaussian Processes for Machine Learning*, volume 2. MIT Press.