Systems & Control Transactions

Research Article - Peer Reviewed Conference Proceeding
ESCAPE 35 - European Symposium on Computer Aided Process Engineering
Ghent, Belgium. 6-9 July 2025
Jan F.M. Van Impe, Grégoire Léonard, Satyajeet S. Bhonsale, Monika E. Polańska, Filip Logist (Eds.)

PSE PRESS

Nonmyopic Bayesian process optimization with a finite budget

José L. Pitarch[a]*, Leopoldo Armesto[b], and Antonio Sala[a]

[a] Universitat Politècnica de Valencia, Instituto de Automática e Informática Industrial (ai2), Valencia, Valencia, Spain
[b] Universitat Politècnica de Valencia, Instituto de Diseño y Fabricación (IDF), Valencia, Spain
* Corresponding Author: jlpitarch@isa.upv.es

ABSTRACT

Optimization under uncertainty is inherent to many PSE applications ranging from process design to RTO. Reaching process true optima often involves learning from experimentation, but actual experiments involve a cost (economic, resources, time) and a budget limit usually exists. Finding the best trade-off on cumulative process performance and experimental cost over a finite budget is a Partially Observable Markov Decision Process (POMDP), known to be computationally intractable. This paper follows the nonmyopic Bayesian optimization (BO) approximation to POMDPs developed by the machine-learning community, that naturally enables the use of hybrid plant surrogate models formed by fundamental laws and Gaussian processes (GP). Although nonmyopic BO using GPs may look more tractable, evaluating multi-step decision trees to find the best first-stage candidate action to apply is still expensive with evolutionary or NLP optimizers. Hence, we propose modelling the value function of the first-stage decision also with a GP, whose data will correspond to virtual evaluations of second-stage decision trees build upon myopic rollouts. Thus, the nonmyopic initial decision can be efficiently optimized via BO and the virtually learned value function. Effectiveness of the approach is demonstrated in a wide benchmark with synthetically generated functions as well as to optimize small batch production with a chemical reactor.

Keywords: Optimization, Machine Learning, Batch Process, Algorithms, Design Under Uncertainty, POMDP

INTRODUCTION

Process optimization in actual applications must deal with uncertainty in exogenous factors (such as weather predictions or market prices), partial and noisy measurements, as well as plant-model mismatch (either parametric or structural). There are widely used techniques to deal with each one of these factors in isolation: multi-stage stochastic optimization offers a good performance-robustness compromise in the case of exogenous and measurable parametric uncertainty [1]; Kalman-based filters and data reconciliation deal with partial and noisy measurements [2]; and extremum seeking [3], modifier adaptation [4], policy search [5], or Bayesian optimization (BO) [6], are typical approaches used to drive the operation point of a real process to its true steady-state optimum by acquiring experimental information. Focusing on the existence of plant-model mismatch and partial information available through measurements, the reader will note that none of the cited approaches include the cost of experimentation (either economic, resource consumption, or number of batches to produce) explicitly. In other words, their objective is just to reach the closest possible neighborhood of true optimum at any cost. But a limit on the experimental budget usually exists in practice. Moreover, with limited budgets one may be interested in finding the best trade-off between an accumulated exploration cost and exploitation performance.

The right way to formulate the problem of finding the best cumulative trade-off on process performance and experimental cost under uncertainty over a finite budget is a Partially-Observable Markov Decision Process (POMDP) whose states are uncertain process beliefs [7]. The general way to approach these problems is evaluating belief trees of candidate actions and plausible observations via dynamic programming, as Monte Carlo Tree Search (MCTS) algorithms do [8]. But their computational cost is prohibitive when the state and action spaces are of continuous nature.

If such a belief is modeled by a Gaussian process (GP), the nonmyopic BO acquisition functions developed by the machine learning community are a more tractable way to approximate POMDPs. The key idea of nonmyopic BO is to look ahead several steps and compute the best candidate decisions based on an estimation on the cumulative expected value or improvement over the budget horizon [9]. Anyway, optimizing multi-step trees with GPs as states in one shot is also challenging, and rollout algorithms with default myopic BO acquisition functions are normally used to complete the value function estimation after the first lookahead step [10], which can lead to conservative results.

Following this path, recently we proposed a variant tailored for process experimental optimization [11], in which the decision in each action node of the tree is provided by a myopic BO acquisition function from a list of the most widely used in the literature. In this way, the algorithm can dynamically select the most promising acquisition function along the sequence of decisions to optimize the expected value over the decision horizon or budget. We also used Gauss-Hermite Quadrature [12] in each observation node as an efficient way to approximate the value function. Although the approach is much more tractable than other nonmyopic BO in the literature (note that no nested optimizations are involved), its expected optimality proven to be slightly lower due to limiting the first-stage decisions on myopic BO acquisition functions.

To remove such conservativeness, hence unlocking the extra performance that nonmyopic BO can provide in theory but at lower computational costs than the existing proposals in the literature, here we propose *learning the value function of the first-stage decision* also with a GP, whose data will correspond to virtual evaluations of our POMDP decision tree in [11] for the subsequent stages, instead of typical rollouts with expected value or improvement BO. In this way, the first-stage decision can be efficiently optimized via BO with the virtually learned value-function GP. In other words, we use *myopic BO to optimize the nonmyopic BO acquisition function*. Note that evaluating a POMDP decision tree via approximate dynamic programming (ADP) is an expensive function from the computational perspective, which fits into the main hypothesis of BO.

In the next section we briefly summarize the key methods involved in our proposal. Then, we present the nonmyopic BO-POMDP strategy and a basic pseudocode. The proposal is then evaluated in a synthetic benchmark of unconstrained optimization functions, where we also provide the results gathered when comparing to some state of the art nonmyopic and myopic BO. The paper closes with an illustrative application to the Otto-Williams reactor and some remarks.

METHODS AND PROBLEM STATEMENT

Hybrid models with Gaussian Processes

The hybrid plant models in this work are understood as a basis of fundamental laws that is then customized to a particular process with gathered experimental data. In this case, instead of incorporating deterministic machine learning sub-models obtained, for instance, with sum-of-squares polynomial regression [13], we employ probabilistic Gaussian Process regression to adjust first-principles predictions to the real data, getting confidence intervals around.

GPs are characterized by a mean function $\mu(x)$ and a covariance generator $\kappa(x_a, x_b) \coloneqq \mathrm{cov}\big(f(x_a) - f(x_b)\big)$, being $f(x)$ the true function whose knowledge is partially unknown. A $GP\{X, Y\}$ is built from input-output data

$$X \coloneqq [x_0, x_1, x_2, \dots, x_n]; \quad Y \coloneqq [y(x_0), y(x_1), \dots, y(x_n)];$$

where measurements are assumed to be $y(x) = f(x) + w$, $w \approx \mathcal{N}(0, \lambda^2)$, being λ the noise standard deviation. The a priori mean $\mu(x) \coloneqq 0$ if pure experimental optimization is to be pursued [11], or set to a first-principles model.

Every time new data is gathered, the GP mean and covariance kernel are updated by [14]:

$$\mu(x) = \mu(x) + \kappa(x, X)(\kappa(X, X) + \lambda^2 I)^{-1}\big(Y - \mu(X)\big) \qquad (1)$$

$$\kappa(x_a, x_b) = \kappa(x_a, x_b) - \kappa(x_a, X) \cdot (\kappa(X, X) + \lambda^2 I)^{-1} \kappa(x_b, X)^T \qquad (2)$$

Moreover, GPs can themselves contain parametric components with adjustable parameters θ, such that $\phi(x)\theta$ is the deterministic prediction by these components. In such a case, the GP covariance kernel becomes

$$\bar{\kappa}(x_a, x_b) \coloneqq \kappa(x_a, x_b) + \phi(x_a)\Sigma_\theta \phi(x_b)^T \qquad (3)$$

where Σ_θ is the a priori parameter covariance matrix.

Myopic Bayesian optimization

The goal of BO is to reach the optimum of a static mapping $f(x)$. In particular, BO proposes the next experiment to actually test by means of optimizing an acquisition function $F(\mathrm{GP}, x)$ with a surrogate model (cheaper to evaluate than $f(x)$), usually a GP built from process data.

The acquisition function $F(\mathrm{GP}, x)$ is the cornerstone of the method, because it defines a certain tradeoff between exploitation of the already acquired knowledge and exploration to improve such a knowledge. There are several well-known heuristics in the BO literature that perform really well and are cheap to optimize [15]: Expected Value; Probability of Improvement; Expected Improvement; Lower-Upper Confidence Bound.

Given a GP, the basic BO algorithm is a one-step optimization and subsequent GP update, as follows:

$$x = \arg\min_{x \in \mathbb{X}} F(\mathrm{GP}, x) \text{ or } x = \arg\max_{x \in \mathbb{X}} F(\mathrm{GP}, x) \qquad (4)$$

$$GP \leftarrow GP \oplus \{x, y(x)\} = \{X \cup x, Y \cup y(x)\} \tag{5}$$

This process repeats until some criteria is met, for instance, no improvement observed, number of experiments, o time exhausted. Of course, constraints $g(x) < 0$ can be also realized in BO in straightforward ways [11].

However, the exploitation-exploration tradeoff that these standard functions define is just a myopic heuristic that does not account for the potential gains of optimizing at several steps. There are improved heuristics that look two steps ahead, such as Knowledge Gradient or Predictive Entropy Search [16]. These usually show better performance than the classic ones, but at much higher computational cost. Anyway, none of these considers problem-specific cumulative costs and budgets.

Multi-stage scenario optimization

Multi-stage scenario optimization (MSSO) [1] is often the approach to approximate complex Markov decision processes. It combines decision optimization with simulation steps on potential scenarios that may realize (sampling from a probability distribution) within a planning horizon. Basically, MSSO builds a tree of sample paths of depth N through possible states and observations. This is the forward-in-time propagation step. Then, state-value functions can be estimated by backward propagation, providing the best decision for each stage.

If such a decision tree is iteratively built, *value functions can be learned* through the evaluations, which might allow generalization across states, i.e., the already visited states or evaluated decisions might tell us something about the value of the ones not visited yet [17].

In this paper we recall the Gauss-Hermite quadrature (GHQ) method [18] to approximate integrals as an efficient way to build sample-paths trees and to estimate state-value functions in POMDPs.

Problem statement

Assume that a statistical belief of the actual $f(x)$ and $g(x)$ given by GPs is available, constructed from a prior inaccurate model and a set of past data X_0,Y_0 (e.g. the optimum of the first-principles model tested in the actual plant). Then, there is a finite budget of N experiments or production batches to realize, so that the task is deciding which will be the most promising sequence of decisions x to perform such that they are optimal in some sense related to a POMDP. After each decision x_k ($k = 1, \dots, N-1$) is taken, a process observation y_k will be available, so that an immediate reward or cost $r([X_{k-1}, x_k], [Y_{k-1}, y_k])$ is computed. Compact notation $r(X_k, Y_k)$ is used from now on when the order of the samples does not matter. A terminal cost $J_N(X_N, Y_N)$ will be attained as a result of the last decision x_N. Then, the goal is to optimize the expected overall cumulative cost (being $\gamma > 0$ a discount factor)

$$J \coloneqq \mathbb{E}\{\gamma^N J_N(X_N, Y_{f,N}) + \textstyle\sum_{k=1}^{N-1} \gamma^k r(X_k, Y_{f,k})\} \tag{6}$$

over a set of N future observations of $f(x)$ denoted by $Y_{f,N} \coloneqq \{y_f(x_1), \dots, y_f(x_N)\}$, such that the observations of process constraints $Y_{g,N} \coloneqq \{y_g(x_1), \dots, y_g(x_N)\} < 0$ hold with desired probability threshold, e.g., 2σ confidence.

NONMYOPIC OPTIMIZATION STRATEGY

Our proposal of nonmyopic acquisition function approximates the value function of the first-stage decision set x_0 by ADP, building scenario trees that branch with some easy-to-compute acquisition functions from myopic BO at the decision nodes and with GHQ points at the observation ones [11]. Evaluating this tree backwards for each first-stage candidate $\tilde{x}_i$ provides the corresponding value-function estimate $\hat{V}_i$, data that serves to fit a GP model of the first-stage value function. The proposed exploration strategy is illustrated in Figure 1 on next page, where the root node has a fixed current belief on the process to be optimized after incorporating previous actual observations, and a value-function belief that is learned dynamically with each subsequent tree evaluation.

Hence, the idea is to progressively improve such a value function model $GP_V\{\tilde{x}, \hat{V}\}$ while performing myopic BO on it to propose new candidates $\tilde{x}_i$. When this learning-optimization process converges (e.g. reaching an input-deviation tolerance or maximum iteration number), the best candidate decision found is tested in the real process and everything repeats for the remaining budget horizon. Algorithm 1 summarizes de proposed procedure.

Algorithm 1: Nonmyopic BO acquisition function.

Global: $X_0, Y_0, N, H_r, V_0, \text{optol}$	%Initial data
1: Create $GP_c\{X_0, Y_0\}$	%Build root cost GP
2: For $k \coloneqq 1$ **to** N **do:**	
3: If $k = 1$ **then** $\tilde{x} \leftarrow X_0$**; otherwise** $\tilde{x} \leftarrow \arg\min_{\mathrm{x} \in \mathbb{X}} F(GP_V, \mathrm{x})$	%Value-function BO
4: $b = \text{NodeV}(GP_c)$	%Create the root node
5: $\bar{H} \leftarrow N - k$	%Reduce prediction horizon
6: $b.\text{Expand}(\tilde{x})$	%Branch second-stage tree
7: $\hat{V} \leftarrow b.\text{ValueFunction}()$	%Estimate V($\tilde{x}$)
8: $GP_V \leftarrow GP_V \oplus \{\tilde{x}, \hat{V}\}$	%Update GP of V(x)
9: If $\lvert V_0 - \hat{V} \rvert > \text{optol}$ **then** $V_0 \leftarrow \min_{\hat{\mathrm{V}}} GP_V$**; Goto 3**	
10: Realize $x \leftarrow \arg_{\tilde{\mathrm{x}}} \min_{\hat{\mathrm{V}}} GP_V$	%Actual experiment
11: Get $y_f \leftarrow f(x) + w$	%New observation
12: $GP_c \leftarrow GP_c \oplus \{x, y_f\}$;	%Update GP of r(X,Y)

Where, abusing notation, $\min_{\hat{\mathrm{V}}} GP_V$ finds the lowest value $[V^*, i_d] = \min GP_V.\hat{\mathrm{V}}$ already in the GP dataset, and $\arg_{\tilde{\mathrm{x}}} \min_{\hat{\mathrm{V}}} GP_V$ stands for the x-coordinate in the GP data corresponding to such a V^*, i.e., $x^* = GP_V.\tilde{\mathrm{x}}[i_d]$.

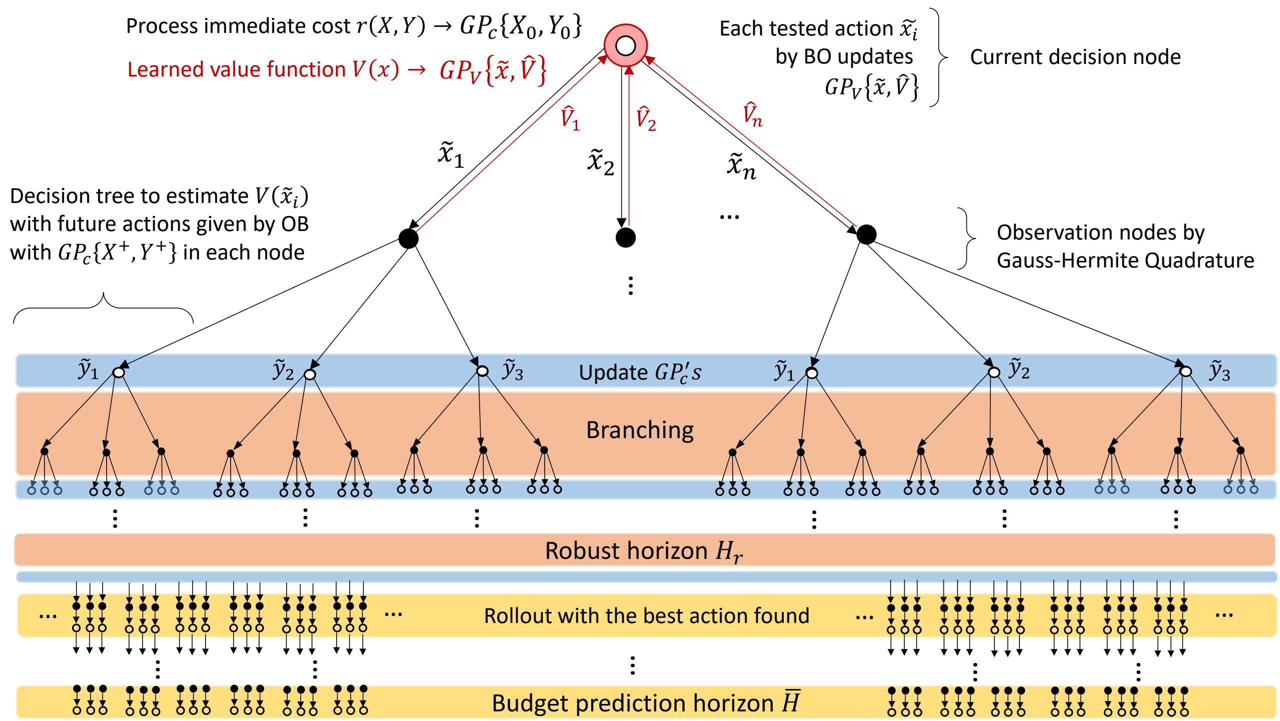

Figure 1: Nonmyopic POMDP decision tree strategy via myopic BO of the first-stage value function.

Note that Algorithm 1 here presented is a simple version that does not include process constraints for the sake of clarity, but considering them is straightforward. See our previous work [11] (and references therein) for ideas to deal with constraints in BO-POMDP as well as details and pseudocode on the exploration-decision tree. Moreover, Algorithm 1 assumes a very basic myopic BO in statements 3 and 9, but there is no inconvenience in calling sophisticated software BO packages instead. In fact, Bayesian adaptive direct search (BADS) [19] turned to be a good implementation to realize this higher first-stage BO without worrying on hyperparameter tuning and right covariance-kernel choice for the value-function GP.

Performance evaluation

To evaluate the proposed nonmyopic optimization strategy, we tested it to optimize the cumulative cost (6) on 100 functions $f(x)$ of two decision variables (normalized to the squared range [-1,1]) and zero mean, synthetically generated from the exponential covariance kernel

$$\kappa(x_a, x_b) = \Sigma \cdot e^{-\frac{\|x_b - x_a\|^2}{2 \cdot \sigma^2}}, \qquad (7)$$

with $\sigma = 0.25$, $\Sigma = 1$, and considering a measurement noise level $\lambda = 0.002$. The problem budget is $N = 10$. Algorithm 1 as well as the proposals of [10] and [11] are both set to branch up to a horizon $H_r = 2$, and the discount factor is set to no discount ($\gamma = 1$).

The average performance got by Algorithm 1, references [10] and [11], as well as myopic BO with typical acquisition functions of negligible computation time (expected improvement, probability of improvement, and expected value) are presented in Table 1 below.

Table 1: Average regrets with respect to the best achievable performance (always knowing the true optimum).

Method:	**[10]**	**Alg 1**	**[11]**	**PI**	**EV**	**EI**
Regret:	7.92	7.77	8.28	9.93	9.73	11.47
CPU (s):	22.3	13	0.68	-	-	-

Algorithm 1 is the one that got lower average regret in the benchmark, followed by our implementation of [10] consisting in a multi-start strategy of a regular grid on the first-stage input space that provides the initial guess to an interior-point NLP optimizer. Furthermore, the average computation load of the proposed Algorithm 1 is about half of the required by [10], as expected because BO is an efficient approach to optimize expensive-to-evaluate functions. Note also that our reduced first-stage exploration strategy [11], although gets a slightly higher regret, it does the work at much lower computation time. Moreover, it outperforms by 15% the best myopic BO tested, and by 30% the usually used expected improvement acquisition function. The evaluation was run in a standard laptop equipped with an Intel Core i7-4510U CPU.

SMALL-BATCH PRODUCTION EXAMPLE

Next, Algorithm 1 is tested in a simulated chemical batch production process with the Otto-Williams stirred tank reactor. There are two reactants, A and B, whose stream F_A is not controlled but known and F_B is decision variable. The reactor temperature T_R can be set so it is also decision variable. These two define the quality of the product at the reactor outlet once the three chemical reactions are completed. In total, six compounds (or species) exist inside the reactor: 4 products (C, E, G, and P) and the reactants A and B. The equations of the chemical dynamic model and associated parameters are omitted for brevity. See [20] and references therein.

As the mass fraction of the product C is one order of magnitude below the rest of the compounds, a usual gross representation of the process is to consider only the other five species, with the corresponding structural and parametric plant-model mismatch. Then, only two parallel reactions inside the reactor are considered:

$$A + 2B \overset{r_1}{\rightarrow} P + E; \quad A + B + P \overset{r_2}{\rightarrow} G + E$$

The mass-fraction equations in steady state to act as prior mean $\mu(F_B, T_R)$ for the process belief are:

$$\begin{aligned} \frac{dX_A}{dt} &:= \mathrm{F_A} - (\mathrm{F_A} + F_B)\, X_A - (r_1 + r_2)\, \mathrm{V_R} = 0 \\ \frac{dX_B}{dt} &:= F_B - (\mathrm{F_A} + F_B)\, X_B - (2\, r_1 + r_2)\, \mathrm{V_R} = 0 \\ \frac{dX_E}{dt} &:= -(\mathrm{F_A} + F_B)\, X_E + 2\, r_1 \mathrm{V_R} = 0 \\ \frac{dX_G}{dt} &:= -(\mathrm{F_A} + F_B)\, X_G + 3\, r_2 \mathrm{V_R} = 0 \\ \frac{dX_P}{dt} &:= -(\mathrm{F_A} + F_B)\, X_E + (r_1 - r_2)\, \mathrm{V_R} = 0 \end{aligned} \tag{8}$$

where $\mathrm{F_A} = 1.8275$, $\mathrm{V_R} = 2105$, and the reaction rates:

$$r_1 := 1.65 \cdot 10^8\, e^{-\frac{8097}{273+T_R}}\, X_A\, X_B^2$$

$$r_2 := 2.61 \cdot 10^{13}\, e^{-\frac{12398}{273+T_R}}\, X_A\, X_B\, X_P$$

The optimization is to sequentially decide the production setpoints $x := [F_B, T_R]$ that get the highest cumulative economic benefit at the end of producing $N = 7$ batches of the products P and E with the actual reactor:

$$\max_{F_B, T_R} J := \textstyle\sum_{k=1}^{10} \left(X_{P,k}\, \mathrm{P_P} + X_{E,k}\, \mathrm{P_E}\right)\left(\mathrm{F_A} + F_{B,k}\right) - \mathrm{F_A C_A} - F_{B,k}\mathrm{C_B}$$

$$\text{s.t.: } X_{G,k} < 9.5\%;\ 3 \le F_{B,k} \le 6\text{kg/s};\ 70 \le T_{R,k} \le 100°\text{C} \tag{9}$$

where $\mathrm{P_p} = 1143.38$ and $\mathrm{P_E} = 25.92$ are product selling prices, $\mathrm{C_A} = 76.23$ and $\mathrm{C_B} = 114.34$ are reactant costs.

The simple non-parametric exponential kernel with $\Sigma_{\kappa_f} = 25$ and $\sigma_f = \sqrt{0.2}$ is chosen to build the GP belief for the immediate economic benefit plant-model mismatch:

$$\kappa_f(x_a, x_b) := \Sigma_{\kappa_f}\, e^{-\frac{\|x_b - x_a\|^2}{2 \cdot \sigma_f^2}} \tag{10}$$

Inputs are normalized to $x \in$ [-1,1] to easily interpret the length scale σ_f. Same kernel class is set up for $\kappa_g(x_a, x_b)$ to model the limit on X_G, but with $\sigma_g = \sqrt{0.2}$ and $\Sigma_{\kappa_g} = 0.5$. These values are chosen by mere intuition in accordance with model (8), and no GP-hyperparameter tuning is then performed from run to run with the reactor due to the small batch assumption.

The given measurement noise is $\lambda_f = 0.5$ for the immediate benefit and $\lambda_g = 0.1$ for the constraint on X_G.

Algorithm 1 has been set up with $H_r = 3$, optol equal to BADS default tolerance. The first batch is run at $x_0 = [4.62\text{kg s}^{-1}, 81.9°\text{C}]$, which is the mean model (8) optimum without violating constraints, whose predicted immediate economic benefit is 157€. However, the actual benefit (noise free) at such a x_0 is 176.17€. Note that model (8) is pessimistic so, under non-negligible measurement noise, greedy BO algorithms are not able to see any potential benefit in testing nearby x_0 instead of remaining on it.

The actual-reactor optimum batch satisfying constraints provides 188.92€ at $x^* = [4.89\text{kg s}^{-1}, 87.64°\text{C}]$, but the goal is not to reach it at any cumulative cost (9).

Table 2 below shows the performance obtained by Algorithm 1 to produce the 7 batches. Myopic BO has been also applied with four widely used acquisition functions: expected improvement, probability of improvement, upper confidence bound, and expected value.

Table 2: Regrets with respect to the best achievable performance (computed noise free with the "actual" reactor).

Method:	Alg 1	EI	PI	UCB	EV
Regret (€):	44.37	56.64	74.7	48.38	76.45

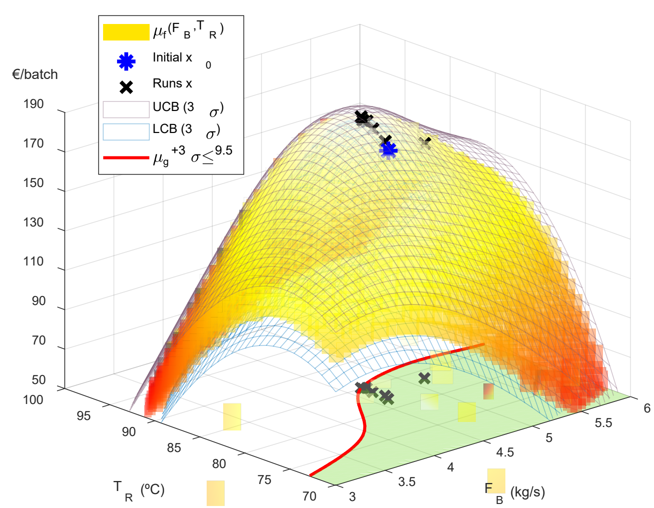

Figure 2. Reactor knowledge after producing 7 batches.

For completeness, Figure 2 shows the sequence of batches decided by Algorithm 1 and the final a posteriori belief on the process (mean model (8) plus the GPs fitted to the measured data).

CONCLUDING REMARKS

This paper proves the advantages of non-myopic BO in process optimization problems of POMDP nature. In contrast to other computationally intensive strategies, we proposed a non-myopic acquisition function that uses efficient myopic BO to evaluate rollout POMDP trees.

In the test with the chemical reactor, our proposal performs better than standard BO, as expected. However, a single run with a particular reactor is not statistically conclusive. Anyway, this "one-chance run" is often the situation in multiproduct industrial plants.

The main performance limitation in practice comes from the ability of the probabilistic belief to represent the actual plant-model mismatch.

ACKNOWLEDGEMENTS

This work is part of Grants PID2023-146747NB-C21 (SIECMA) and PID2020-116585GB-I00 (LOCPU) funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU.

REFERENCES

1. Li C, Grossmann IE. A review of stochastic programming methods for optimization of process systems under uncertainty. *Front. Chem. Eng.* 2:622241 (2021) https://doi.org/10.3389/fceng.2020.622241
2. Bai S, Thibault J, McLean DD. Dynamic data reconciliation: Alternative to Kalman filter. *J. Proc. Cont.*, 16(5):485-498 (2006) https://doi.org/10.1016/j.jprocont.2005.08.002
3. Ariyur KB, Krstic M. Real-time optimization by extremum-seeking control. Wiley & Sons (2003)
4. Bottari A. Real-time optimization modifier adaptation approach using quadratic approximation of the plant-model mismatch function. *J. Proc. Cont.* 136:103180 (2024) https://doi.org/10.1016/j.jprocont.2024.103180
5. Deisenroth MP, Neumann G, Peters J. A survey on policy search for robotics. *Found. & Trends in Rob.* 2(1–2):1-142 (2013) http://dx.doi.org/10.1561/2300000021
6. del Rio Chanona EA, Petsagkourakis P, Bradford E, Graciano JA, Chachuat B. Real-time optimization meets Bayesian optimization and derivative-free optimization: A tale of modifier adaptation. *Comp. & Chem. Eng.* 147:107249. https://doi.org/10.1016/j.compchemeng.2021.107249
7. Smallwood R, Sondik E. The optimal control of partially observable Markov decision processes over a finite horizon. *Oper. Res.* 21:1071–1088 (1973) https://doi.org/10.1287/opre.21.5.1071
8. Świechowski M, Godlewski K, Sawicki B, Mańdziuk J. Monte Carlo tree search: A review of recent modifications and applications. *Artif. Intell. Rev.* 56(3):2497-2562 (2023) https://doi.org/10.1007/s10462-022-10228-y
9. Jiang S, Jiang D, Balandat M, Karrer B, Gardner J, Garnett R. Efficient nonmyopic Bayesian optimization via one-shot multi-step trees. *Adv. Neur. Inform. Proc. Sys.* 33:18039-18049 (2020) https://doi.org/10.48550/arXiv.2006.15779
10. Lam R, Willcox K. Lookahead Bayesian optimization with inequality constraints. *Adv. Neur. Inform. Proc. Sys.* 30:1888 - 1898 (2017)
11. Pitarch JL, Armesto L, Sala A. POMDP non-myopic Bayesian optimization for processes with operation constraints and a finite budget. *Rev. Iber. Autom. Inform. Ind.* 21(4):328-338 (2024) https://doi.org/10.4995/riai.2024.21142
12. Stroud AH, Secrest D. Gaussian Quadrature Formulas. Prentice Hall (1966)
13. Pitarch JL, Sala A, de Prada C. A systematic grey-box modeling methodology via data reconciliation and SOS constrained regression. *Processes* 7(3):170 (2019) https://doi.org/10.3390/pr7030170
14. Rasmussen CE, Williams CK. Gaussian processes for machine learning. Cambridge, MIT press (2006) https://doi.org/10.7551/mitpress/3206.001.0001
15. Frazier PI. Bayesian optimization. *Informs* 255-278 (2018) https://doi.org/10.1287/educ.2018.0188
16. Hernández-Lobato JM, Hoffman MW, Ghahramani Z. Predictive entropy search for efficient global optimization of black-box functions. *Adv. Neur. Inf. Proc. Sys.* 27:918-926 (2014) https://doi.org/10.48550/arXiv.1406.2541
17. Mern J, Yildiz A, Sunberg Z, Mukerji T, Kochenderfer MJ. Bayesian optimized monte carlo planning. *AAAI Conf. Artific. Intell.* 35(13):11880-11887 (2021) https://doi.org/10.1609/aaai.v35i13.17411
18. Abramowitz M, Stegun IA. Handbook of mathematical functions, [Equation 25.4.46], Dover Publications (1972), ISBN: 978-0-486-61272-0.
19. Acerbi L, Ma WJ. Practical Bayesian optimization for model fitting with Bayesian adaptive direct search. *Adv. Neur. Inf. Proc. Sys.* 30:1834-1844 (2017) https://doi.org/10.48550/arXiv.1705.04405
20. Rodríguez-Blanco T. Modifier Adaptation for process optimization with uncertainty. PhD Thesis, Universidad de Valladolid (2017). http://uvadoc.uva.es/handle/10324/28631