Parallel Bayesian Optimization for Optimal Scheduling of Underground Pumped Hydro-Energy Storage Systems

Maxime Gobert
Mathematics and Operational Research
University of Mons
Mons, Belgium
maxime.gobert@umons.ac.be

Jan Gmys
INRIA Lille - Nord Europe
Université de Lille, CNRS/CRIStAL
Lille, France
jan.gmys@inria.fr

Jean-François Toubeau
Power Systems and Markets Research
University of Mons
Mons, Belgium
jean-francois.toubeau@umons.ac.be

Nouredine Melab
Université de Lille
CNRS/CRIStAL, INRIA Lille - Nord Europe
Lille, France
nouredine.melab@univ-lille.fr

Daniel Tuyttens
Mathematics and Operational Research
University of Mons
Mons, Belgium
daniel.tuyttens@umons.ac.be

François Vallée
Power Systems and Markets Research
University of Mons
Mons, Belgium
francois.vallee@umons.ac.be

***Abstract*—Underground Pumped Hydro-Energy Storage stations are sustainable options to enhance storage capacity and thus the flexibility of energy systems. Efficient management of such units requires high-performance optimization algorithms able to find solutions in a very restricted timing to comply with the responsive energy markets. In this context, parallel computing offers a valuable solution to ensure appropriate decisions that maximize the profit of the station operator, while guaranteeing the safety of the energy network. This study investigates the use of three existing algorithms in Parallel Bayesian Optimization, namely q-EGO, BSP-EGO and TuRBO. The three algorithms have different inherent behaviors in terms of parallel potential and, even though TuRBO scales better, q-EGO remains the best choice regarding the final outcomes for all investigated batch sizes and manages to get up to 5 times more profits than other approaches.**

***Index Terms*—Bayesian Optimization, Gaussian Process, Batch-based Parallelism, Optimization, Electrical Engineering.**

I. Introduction

Integrating renewable energy resources is a key challenge to ensure the transition towards a low-carbon energy system. Electricity storage systems provide a valuable solution to compensate the uncertain production, thus offering sustainable means to increase the flexibility of the system [1]. An appropriate option regarding storage technologies is offered by Underground Pumped Hydro-Energy Storage (UPHES). However, in modern competitive energy networks, individual actors rely on efficient operational strategies, enabling them to hedge the uncertainty of renewable energy resources. It is thus essential to dispose of efficient tools to take informed decisions at the different time steps of the energy markets (e.g., from long-term towards real-time) [2]. Let us assume that for a decision $\boldsymbol{x} \in \mathbb{R}^d$, the expected profit of a UPHES operator is given by $f : \mathbb{R}^d \to \mathbb{R};\ \boldsymbol{x} \mapsto y = f(\boldsymbol{x})$. The optimization problem writes:

$$\max_{\boldsymbol{x} \in \Omega \subset \mathbb{R}^d} f(\boldsymbol{x}) \tag{1}$$

The simulator, f in Eq. 1, is a time-consuming Black-Box function, which is further described in Section II. Surrogate-assisted optimization is extensively used in this context. Indeed, surrogate models allow to reduce the computational burden of the optimization by (partially) replacing the time-consuming simulator and/or by evaluating the desirability of a candidate point.

More precisely, Parallel Bayesian Optimization (PBO) is emerging as a powerful framework for such problems. This surrogate-guided optimization approach relies on a surrogate model - often Gaussian Process (GP) regression. Based on the model, an Acquisition Function (AF) is defined to assess the value of any candidate point. Optimizing the AF yields the best candidate point according to this specific AF. In this way, only valuable (in the sense of the AF) points are evaluated in parallel using the time-consuming simulator. Consequently, the overall optimization time can be considerably reduced. This is essential due to the time constraint arising from both the organization of energy markets and the complex modelling of physical and economic constraints of pumped-hydro systems.

More generally speaking, we refer to the process responsible of selecting the next candidate points as the Acquisition Process (AP). Clearly, the choice of the AP is of crucial importance to balance the exploration and exploitation during the optimization process. However, no clear recommendation can be extracted from the literature. In addition, the main potential for exploiting parallel computing lies in the batch-parallel evaluation of the candidates. Therefore, the AP must be able to provide a batch of valuable candidates. Various approaches are presented in Shahriari *et al.* [3], however the batch selection remains a challenging question especially for large batch sizes.

In this paper, we investigate three different parallel batch-based algorithms using different APs, and we compare the optimized average profit: q-Efficient Global Optimization (q-EGO) from Ginsbourger, Le Riche and Carraro [4] (2008) and a revisited version allowing multi-criteria sampling; Binary Space Partitioning EGO (BSP-EGO) from Gobert *et al.* [5] (2020); and TrUst Region BO (TuRBO) from Eriksson *et al.* [6] (2019). Experiments are carried out with different batch sizes to evaluate and compare the scalability of the approaches. The main objective is to identify the most suitable algorithm that responds to the specificities of the UPHES optimization problem (short time budget and acquisition time not negligible compared to simulation time).

The conducted study reveals that even though q-EGO seems less suited for parallel computing than TuRBO or BSP-EGO, it achieves the best performance regarding the final outcomes for moderate batch sizes (≤ 8).

In Section II, a detailed explanation of the complex UPHES operation is provided along with a mathematical formulation of the resulting optimization problem. In Section III, the PBO algorithms designed to optimize the station management are presented with a focus on their specificities. Section IV is dedicated to experiments and results, where the outcomes from different algorithms are compared. In Section V, we discuss the experimental results. Finally, conclusions are drawn in Section VI.

II. Underground Pumped Hydro-Energy Storage

Due to their ability to quickly and cost-effectively mitigate energy imbalances, Pumped Hydro-Energy Storage (PHES) stations offer an appropriate storage solution. PHES plants are composed of (at least) a lower and an upper reservoirs from which water is exchanged to either produce or store energy. In off-peak periods, production might exceed consumption such that energy is saved by pumping water from lower basin into the upper one. It then provides a substantial reserve of energy that can be later released when needed, e.g., to maintain the transmission grid stability. Recent progress in power electronics have enabled PHES units to operate with a reliable variable-speed feature in both pump and turbine modes. The flexibility offered by these facilities is highly valuable. Indeed, it improves the economic efficiency of existing resources such as wind farms or thermal power plants [7], [8], and provides ancillary services to ensure the grid stability (such as frequency control or congestion management).

The inherent potential of PHES units leads to the development of new technological solutions such as Underground PHES (UPHES) for which the lower basin is located underground. A significant advantage of UPHES is the limitation of expenses from civil engineering works thanks to the recycling of end-of-life mines or quarries. These stations have very limited impact on landscape, vegetation and wildlife, and are not limited by topography so that more sites can be exploited [9]. In the current competitive framework governing the electricity sector, UPHES units are exploited with an objective of return on investment. Consequently the profit of such stations must be maximized. This task is challenging since the UPHES operation is governed by two main nonlinear effects that cannot be easily modeled with traditional analytical models [10].

Firstly, groundwater exchanges between the reservoirs and their hydro-geological porous surroundings may occur. This situation typically arises for UPHES when the waterproofing work is not feasible or uneconomical [11].

Secondly, UPHES units are generally subject to important variations of the net hydraulic head (i.e., height difference between water levels in the reservoirs). These variations are referred to as the head effects [12], and are typically quantified through laboratory measurements on a scaled model of the hydraulic machines [13]. This characterization of head effects is important since the head value defines both the safe UPHES operating range as well as the efficiency of both pump and turbine processes. In this way, the safe operating limits in pump and turbine modes continuously vary over time with regard to head variations. In general, the performance curves of UPHES stations are difficult to model since they present a non-convex and non-concave behavior.

Directly integrating these effects into model-based optimization (which maximizes the UPHES profit in the different market floors) implies high computational burden or strong assumptions that may jeopardize the feasibility of the obtained solution. To address these issues, a simulation-based BO strategy as been developed. The simulator, which is a black-box from the user perspective (developed independently without access to the source code), returns the daily UPHES profit accounting for all techno-economic constraints.

This simulator is denoted as f in the following and, according to a decision vector $\boldsymbol{x} \in \mathbb{R}^{12}$, it returns the expected profit $y = f(\boldsymbol{x}) \in \mathbb{R}$. The 12-dimensional decision vector includes 8 decision variables to participate to the different time slots of the energy market, and 4 to the reserve market (i.e., provision of ancillary services). The number of decision variables is set in accordance with standard recommendation of electrical engineering. It is subject to modification in order to get more flexibility in future studies. The objective is then to find the decisions that maximize the daily expected profit:

$$\boldsymbol{x}_{opt} = \mathrm{argmax}_{\boldsymbol{x} \in \Omega} f(\boldsymbol{x}),$$

where Ω is the domain (or design space). The objective function f also involves the constraints and deals with them by adding a penalty term. The full description of physical and economical constraints can be found in Toubeau *et al.* [10].

These UPHES decisions must comply with hydraulic and electro-mechanical constraints over the whole daily horizon. This results into a challenging optimization problem (embedded in the simulator), which is discontinuous (from the cavitation effects of the pump-turbine machine that incur unsafe operating zones), nonlinear (from the complex performance curves of the unit), mixed-integer (to differentiate the pump-turbine-idle operation modes), which is subject to uncertainties (e.g., on water inflows and market conditions). The formulation used in this paper can be found in [10].

This kind of management problem in electrical engineering is typically solved using Mixed-Integer Linear Programming (MILP) [14], dynamic programming [15] or nonlinear programming [16]. Current techniques also involve meta-heuristics such as Genetic Algorithm (GA) [17] or Particle Swarm Optimization (PSO) [18].

Since the simulator is time-consuming (such that we cannot perform a large number of simulations during the available time), a judicious choice is to resort to PBO to find a good decision vector. Indeed, a decision must in practice be taken within tens of minutes at most. In this context of very limited time budget and numerical simulations lasting between 6 to 10 seconds, parallel computing offers an efficient tool to improve the search within a fixed wall time. In particular, as it usually provides good candidate solutions with a small amount of evaluations and short timing, PBO appears to be a natural choice.

Usually, BO assumes that time-cost associated to the model fitting and the AP are negligible compared to evaluation time. However, in the UPHES optimization problem, the simulator is considered costly because the optimization budget is defined as a time period. In this context, model fitting and AP cannot be neglected.

III. Parallel Bayesian Optimization

The general idea of BO is to fit a probabilistic predictive model of the black-box objective function f - the simulator - in order to guide the optimization process. Indeed, f being time-consuming, we cannot afford to query a large number of simulations. Therefore, the surrogate model is used to evaluate the utility of a candidate point before it is evaluated with the costly simulator. This utility measure is often called AF, but also Infill Criteria (IC) or figure of merit. It uses the predicted value of the surrogate as well as the prediction variance provided by probabilistic models such as Gaussian Processes (GP). One famous example of AF is the Expected Improvement (EI) used in Efficient Global Optimization (EGO) [19]. BO operates in a loop composed of (i) fitting a metamodel, (ii) searching for the most valuable candidate(s) to simulate, (iii) simulation of the candidate(s). The three steps are referred to as a cycle.

BO involves two key elements: the definition of a surrogate model $\mathcal{M}$ that provides a prediction $\hat{y}_{cand}$ for any candidate point $\boldsymbol{x}_{cand}$ as well as a measure of uncertainty $\sigma(\boldsymbol{x}_{cand})$, and an AP that proposes a (batch of) valuable point(s) for evaluation. Four different APs are investigated in this paper and presented in the following.

A. Gaussian Process Regression

Usually, a GP surrogate model is used. The latter assumes a linear relation between inputs $\boldsymbol{x}$ and outputs y, such that $y = \boldsymbol{\omega}^T \boldsymbol{x} + \epsilon$. The observation/output is often considered noisy, hence the ϵ error term, which is assumed gaussian - $\epsilon \sim \mathcal{N}(0, \sigma^2)$. As explained in [20], a Bayesian approach assumes a prior distribution over weights $\boldsymbol{\omega}$, which is also gaussian ($\boldsymbol{\omega} \sim \mathcal{N}(0, \Sigma)$) and constitutes the prior belief. Starting from that belief, it is possible to update the prior knowing the data $(\boldsymbol{X}, \boldsymbol{y})$, which is defined as the posterior distribution. Using the Bayes rule the posterior is expressed as

$$posterior \ = \frac{likelihood \ \times \ prior}{evidence}. \tag{2}$$

With previous notations, Equation 2 becomes

$$p\left(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{y}\right) = \frac{p\left(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\omega}\right) p\left(\boldsymbol{\omega}\right)}{p\left(\boldsymbol{y}|\boldsymbol{X}\right)} = \frac{p\left(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\omega}\right) p\left(\boldsymbol{\omega}\right)}{\int_{\boldsymbol{\omega}} p\left(\boldsymbol{y}|\boldsymbol{X}, \ \boldsymbol{\omega}\right) p(\boldsymbol{\omega}) d\boldsymbol{\omega}}. \tag{3}$$

The evidence $p(y|\boldsymbol{X})$ is also referred to as the *normalizing constant* and can be expressed as the marginal likelihood by marginalizing over the weights $\boldsymbol{\omega}$. Often, input data $\boldsymbol{X}$ is projected into a feature space using a set of basis functions $\Phi(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_k(\boldsymbol{x}))^T$ which leads to the following model: $y = \boldsymbol{\omega}^T \Phi(\boldsymbol{x}) + \epsilon$. Denoting $\boldsymbol{\Phi} = \Phi(\boldsymbol{X})$, we can simply replace $\boldsymbol{X}$ by $\boldsymbol{\Phi}$ in Equation 3. Knowing the distribution of the posterior $p\left(\boldsymbol{\omega}|\boldsymbol{\Phi}, y\right) \sim \mathcal{N}\left(\frac{1}{\sigma^2} A^{-1} \boldsymbol{\Phi} \boldsymbol{y}, A^{-1}\right)$ with $A = \sigma^{-2} \boldsymbol{\Phi} \boldsymbol{\Phi}^T + \Sigma^{-1}$, it is possible to make inference. The predictive distribution of a design point $\boldsymbol{x}^*$ writes

$$\begin{aligned} p\left(y^*|\boldsymbol{\Phi}, \boldsymbol{y}, \boldsymbol{x}^*\right) &= \int_{\boldsymbol{\omega}} p\left(y^*|\boldsymbol{\omega}\right) p\left(\boldsymbol{\omega}|\boldsymbol{\Phi}, \boldsymbol{y}\right) d\boldsymbol{\omega} \\ &\sim \mathcal{N}\left(\frac{1}{\sigma^2} \Phi(\boldsymbol{x}^*)^T A^{-1} \boldsymbol{\Phi} \boldsymbol{y}, \Phi(\boldsymbol{x}^*)^T A^{-1} \Phi(\boldsymbol{x}^*)\right) \end{aligned} \tag{4}$$

and is also gaussian. It can be shown that Equation 4 can be equivalently written as:

$$\begin{aligned} p\left(y^*|\boldsymbol{\Phi}, \boldsymbol{y}, \boldsymbol{x}^*\right) \sim \mathcal{N}\big(&\Phi(\boldsymbol{x}^*)^T \Sigma \boldsymbol{\Phi} (K + \sigma^2 I)^{-1} \boldsymbol{y}, \\ &K^{**} - K^{*\cdot} (K + \sigma^2 I)^{-1} K^{\cdot *}\big) \end{aligned} \tag{5}$$

where $K = \boldsymbol{\Phi}^T \Sigma \boldsymbol{\Phi}$ is known as the covariance matrix and $K^{(1)(2)} = \Phi(\boldsymbol{x}^{(1)})^T \Sigma \Phi(\boldsymbol{x}^{(2)}) = k\left(x^{(1)}, x^{(2)}\right)$. The k function is referred to as the covariance kernel and $K = \left(k(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)_{i,j \in \{1,\ldots,n\}}$ and is chosen as an hyper-parameter.

B. Acquisition Process for Optimization

The BO framework is illustrated in Alg. 1. The mentioned parallel algorithms (TuRBO, BSP-EGO and q-EGO) follow the same scheme but differ in the candidate selection phase which is represented by the optimization of the α function in Alg. 1. Line 4 of Alg. 1 states that the algorithm is searching for the best (batch of) candidate(s) in terms of α to be added. The candidate selection, referred to as the AP, is also an optimization problem sometimes called inner optimization.

Even though BO and GP seems a legitimate choice in the context of optimization with a relatively small budget, and strong time constraint; the choice of the AP is difficult and must be done in accordance with the time constraint. This is why four different APs presenting different behaviors are investigated in this study.

Algorithm 1 Bayesian Optimization

Initial DoE: $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$
while Budget available **do**
  $\mathcal{M} = \mathcal{GP}(\mathcal{D})$
  $\boldsymbol{x}_{new} = argmax_{\mathcal{D}}(\alpha(\boldsymbol{x}))$
  $y_{new} = f(\boldsymbol{x}_{new})$
  $(\boldsymbol{X}, \boldsymbol{y}) = (\boldsymbol{X}, \boldsymbol{y}) \cup (\boldsymbol{x}_{new}, y_{new})$
end while

The q-EGO algorithm is a common parallel BO approach that relies on a sequential heuristic to select a batch of candidates and uses a global model fitted on the entire data set. BSP-EGO also uses a global model, but uses spatial decomposition in order to perform independent parallel APs and reduce the acquisition time. TuRBO is based on trust regions in which surrogate models are locally fitted. A notable difference lies in the way each algorithm exploits parallelism: q-EGO only uses batch parallelization for the evaluation of the candidates, while BSP-EGO and TuRBO extend parallel computing to the AP.

1) $q-EGO$ - Single criterion: The q-EGO algorithm refers to the work of Ginsbourger *et al.* [4], [21] where they present heuristics to approximate multi-point criteria which are difficult to exploit when the required number of candidates exceeds $q = 2$. The idea is to replace the time-consuming simulation by a fast-to-obtain temporary value in order to partially update the surrogate model $\mathcal{M}$, which allows to propose a new distinct candidate. The operation can be repeated until q design points are selected. Then, it is possible to exactly evaluate them in parallel and replace the temporary values by the real costs. However this strategy has the major drawback of requiring q sequential updates of $\mathcal{M}$ per cycle. In order to alleviate the fitting cost, the budget allocated to the partial fitting of the surrogate model is reduced compared to a full update performed at the beginning of a cycle. Nevertheless, increasing q makes the data set size increase faster and the algorithm becomes prohibitively time-consuming without any further precaution.

2) q-EGO - Multi-criteria: In order to limit the number of surrogate fitting operations, we propose an alternative AP that uses multiple IC in a same cycle. This approach is a combination of q-EGO and a multi-infill approach such as in De Palma *et al.* [22]. Using multiple AFs allows to select different candidate points without updating the surrogate model, assuming that their optimization yields distinct candidate points. Moreover, it has been observed that resorting to different AFs can favorably impact the objective value, especially when the batch size is high. Indeed, repeatedly updating $\mathcal{M}$ using non-simulated data may degrade the relevance of proposed candidates. The process is described in Alg. 2. A counter is initialized to 0 at line 6 and increments each time the AP adds a candidate to $\boldsymbol{X}_{batch}$. We denote α_i the i-th AF, and $n_{cand}[i]$ the number of candidates chosen with α_i. For the same model, a candidate can be chosen according to each AF (11). The optimizations of α_i can be conducted in parallel. If more candidates are required, a partial model update is necessary (line 18). The loop continues until q candidate points are selected.

Algorithm 2 Multi-infill q-EGO acquisition process

1: $\mathcal{M}$: Surrogate model
2: n_{crit}: number of chosen AF
3: n_{cand}: vector of size n_{crit}
4: $n_{cand}[i]$: number of candidates for AF α_i
5: Initialize n_{cand} with minimum 1
6: $ct = 0$; initialize counter
7: $\boldsymbol{X}_{batch} = \{\},\ \boldsymbol{y}_{batch} = \{\}$
8: **while** ct $< q$ **do**
9:   **for** α_i in AF list **do**
10:     **if** $n_{cand}[i] \neq 0$ **then**
11:       $\boldsymbol{x}_{new} = argmax_{\mathcal{D}}(\alpha_i(\boldsymbol{x}), \mathcal{M})$
12:       $n_{cand}[i] \leftarrow n_{cand}[i] - 1$
13:       $\boldsymbol{X}_{batch} = \boldsymbol{X}_{batch} \cup \boldsymbol{x}_{new}$
14:       $\boldsymbol{y}_{batch} = \boldsymbol{y}_{batch} \cup y_{PV}$
15:       $ct \leftarrow ct + 1$
16:     **end if**
17:   **end for**
18:   $\mathcal{M} \leftarrow partial\_fit(\boldsymbol{X} \cup \boldsymbol{X}_{batch}, \boldsymbol{y} \cup \boldsymbol{y}_{batch})$
19: **end while**
20: Returns $\boldsymbol{X}_{batch}$ to exactly evaluate in parallel

3) BSP-EGO: BSP-EGO [5] uses a dynamic binary partition of the search space to optimize local AFs. A local AP based on a global model is conducted in each sub-domain. This considerably reduces the acquisition time compared to q-EGO since it can be conducted in parallel thanks to the spatial decomposition. This is interesting since the optimization must be completed in very restricted timing. The partition evolves in accordance with the performance of each sub-region: the one containing the best candidate in terms of the AF will be split further while the supposedly less valuable sub-regions are merged. At any time, the partition covers the entire search space and involves the same number of sub-regions. The number of sub-regions is chosen to be a multiple of n_{batch}. Having $n_{cand} > n_{batch}$ avoids sampling into clearly low potential sub-regions by evaluating only a subset of candidates. Besides, it is advised to maintain a multiple of n_{batch} sub-domains to balance the load between parallel workers. As the supposed best sub-region is split at each cycle, diversification is imposed at the beginning, while intensification is favored as the budget fades.

BSP-EGO is an algorithm developed by the authors within the objective of tackling time-consuming black-box objective function for which the execution time does not dominate the acquisition time. In this context, it is particularly important to be able to run the AP in parallel. A study over benchmark function is performed in [23] and revealed that BSP-EGO can compete with q-EGO while having a faster AP.

4) TuRBO: TuRBO [6] performs several local optimizations with independent probabilistic models in different trust regions. It aims at compensating the overemphasized exploration resulting from global AP. A number of hyper-rectangles centered at the best solutions found so far are used as trust regions and are maintained simultaneously. The side length for each dimension of the hyper-rectangle is scaled according to the length scale λ_i (from the GP model) while maintaining a total volume of L^d. Thompson Sampling is used to select a batch of candidates inside a trust region, and also to select the region in which a local BO optimization is performed (i.e in which trust region to intensify). In TuRBO, as well as in BSP-EGO, both AP and batch evaluations can be performed in parallel, contrary to q-EGO where only the batch evaluation is executed in parallel.

IV. UPHES Optimization - Performance evaluation

A. Problem instance

The experimental evaluation uses a real-world UPHES station located in Maizeret - Belgium as a test-case. Its configuration is shown in Fig. 1. The lower basin is a former underground open pit mine subject to groundwater exchanges. Furthermore, the surface of both reservoirs is relatively limited, which results in significant head effects. The specific features of the UPHES unit are taken into account in the simulator implemented in the Resource-Action-Operation (RAO) language [24] and Matlab. The UPHES nominal output ranges (for the nominal value of the hydraulic head) are respectively [6, 8] MW and [4, 8] MW in pump and turbine modes and the energy capacity is of 80 MWh.

B. Experimental protocol

TuRBO is implemented within the Python-based BOTorch framework [26], whereas BSP-EGO and q-EGO are implemented in C++ using the BayesOpt library [27]. In all three approaches Message Passing Interface (MPI) is used for parallelization. The UPHES simulator itself is implemented in Matlab and the domain-specific RAO language. As the black-box UPHES simulator executable requires a software license, experiments are performed on a shared university cluster, preventing us from conducting a meaningful speed-up analysis.

Due to energy market constraints, the optimization must be completed within tens of minutes. Exact timing is not relevant for this study since it is subject to case-dependant choices. However to remain consistent with the time-constraints, the global budget of each optimization run is chosen equivalent to 60 cycles. According to usual recommendations in BO, 20% of the budget is allocated to the initial sampling, which is randomly created. The optimization budget for this study is then fixed at 48 cycles, meaning that the budget in terms of simulations increases proportionally to the batch size (and degree of parallelism) q. Experiments are performed for $q = 2, 4, 8$. The total number of simulations is then $q \times 48$, and the budget allocation is summarized in Tab. I.

The metamodel selected for every optimization is a GP model with constant trend and Matern-$\frac{5}{2}$ kernel. Single criterion q-EGO is executed with the EI criterion, and since it is suggested in Rehbach *et al.* [28] that the variance of GP might not be reliable in high dimensions ($\boldsymbol{x} \in \mathbb{R}^{12}$), we also use the predicted value (PV) as an additional AF. According to previous experiments demonstrating that the simultaneous use of complementary AF can improve the performances, and consistently with the observation of [28], PV is used as supplementary AF when the number of candidates per AF exceeds 2. Finally, we follow the experimental setup presented in Tab. II.

Table I: Allocation of the budget according to available computing power (i.e. n_{batch})

n_{batch}	initial sample	simulation budget
2	24	96
4	48	192
8	96	384

Table II: Experimental setup

n_{batch}	TuRBO	q-EGO	MC_q-EGO	BSP-EGO
2	EI	EI	/	EI
4	EI	EI	EI/PV	EI
8	EI	EI	EI/PV	EI

C. Experimental Results

Results displayed in Fig. 2, 3 and 4 present the average profit value of the best found decision vector according to the number of cycles executed by the algorithms. Bar-plots are added to indicate the standard deviation around the average objective value. Final numerical values of the average profit and its standard deviation are also presented in Tab. III, IV and V. Optimization runs are repeated 10 times with 10 different initial sets, and each algorithm is run once with each initial set. Therefore, within each figure, every curve has the same starting point. However, for the sake of readability, windows are re-scaled by removing the 5 first cycles (otherwise the curves are hard to distinguish).

Table III: Average final profit and standard deviation for $n_{batch} = 2$

$n_{batch} = 2$	average profit	standard deviation
TuRBO	-627.33	1222.95
q-EGO	-434.06	751.88
BSP-EGO	-927.54	1033.66

Looking at Fig. 2 presenting the results of TuRBO, q-EGO and BSP-EGO for $n_{batch} = 2$, no clear distinction can be done between the three algorithms. They all manage to significantly improve the expected profit of the UPHES operator, however the budget is not sufficient to guarantee a positive profit. A slight advantage can be given to q-EGO in terms of final profit and consistency. Indeed, it possesses the best profit on average, and its standard deviation is smaller than its contestants. The

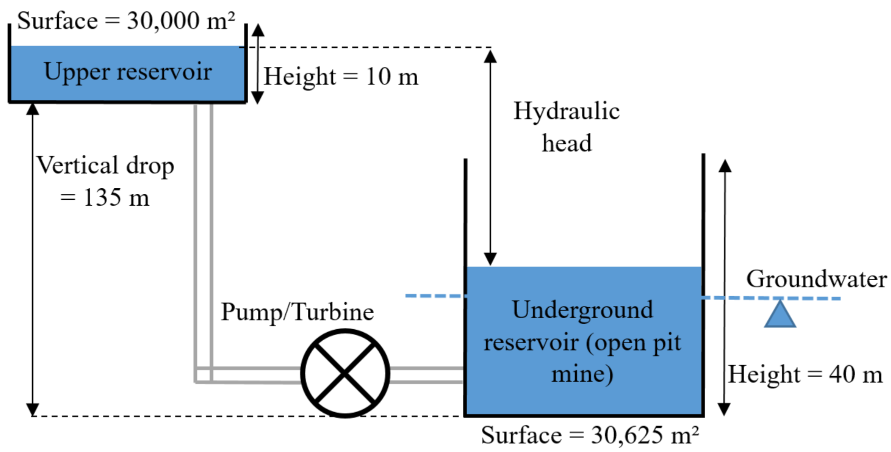

Figure 1: Topology of the modeled PSH unit on Maizeret site [25].

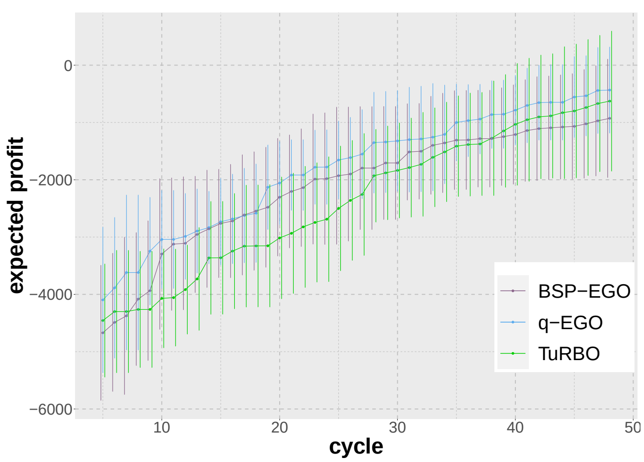

Figure 2: Evolution of the best known objective value according to the number of executed cycles with $n_{batch} = 2$

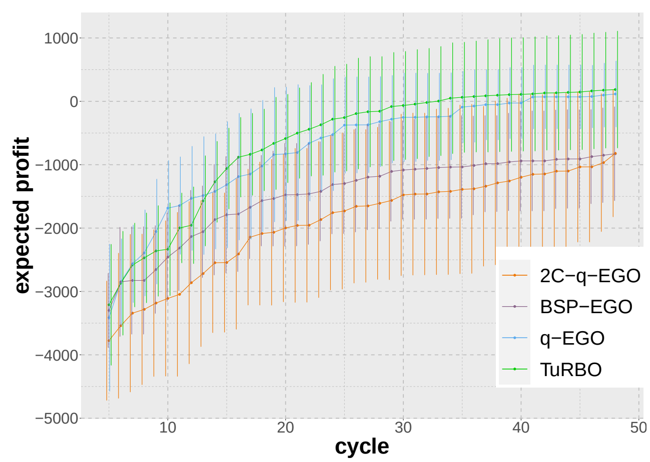

Figure 3: Evolution of the best known objective value according to the number of executed cycles with $n_{batch} = 4$

TuRBO algorithm is able to find very good profits, but not consistently as indicated by the observed extreme values. As for the BSP-EGO curve, even though not statistically significant, it seems to be on the verge of being outperformed by the two other approaches at the end of the budget. Average final objective values for the three algorithms are displayed in Tab. III along with the associated standard deviations. A one-way analysis of variance is conducted to compare the three average final profits and revealed no meaningful difference (p-value $>> 0.05$).

Table IV: Average final profit and standard deviation for $n_{batch} = 4$

$n_{batch} = 4$	average profit	standard deviation
TuRBO	185.81	922.91
***q*-EGO**	115.56	521.24
2C-*q*-EGO	-826.96	741.95
BSP-EGO	-820.71	1002.75

However the difference becomes more prominent when looking at Fig. 3. Indeed, for a larger batch size ($n_{batch} = 4$), the gap grows larger after 20 cycles, and BSP-EGO does not seem competitive for optimizing the UPHES profit. When looking closer at q-EGO and TuRBO, the average curves appears very similar over the cycles. The only noticeable difference, as already observed for $n_{batch} = 2$, is the much wider standard deviation of the TuRBO algorithm. It is almost twice bigger for TurBO as visible in Tab. IV for equivalent average profit. Fig. 3 also shows the results of the multi-criteria version of q-EGO, however according to the curve, the coupling of PV and EI does not perform as well as its single criterion EI counterpart. As for the previous results, a one-way analysis of variance is performed and reveals a significant difference between the four approaches with a confidence over 99%. A pairwise comparison lead with Student's t-tests between TuRBO or q-EGO and the two others is statistically significant (p-value < 0.05), which supports the visual observation of Tab. IV and Fig. 3.

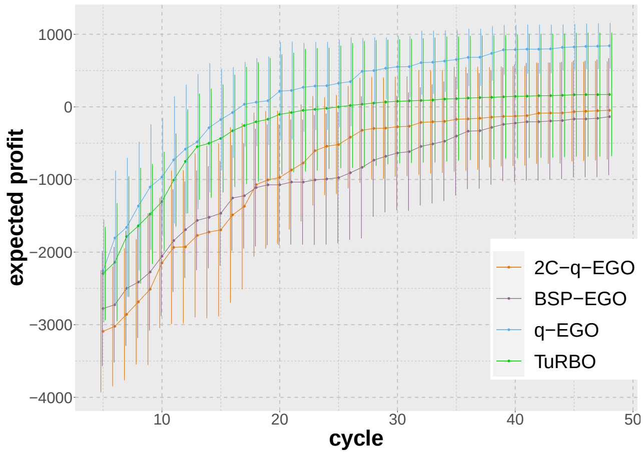

Figure 4: Evolution of the best known objective value according to the number of executed cycles with $n_{batch} = 8$

Table V: Average final profit and standard deviation for $n_{batch} = 8$

$n_{batch} = 8$	average profit	standard deviation
TuRBO	170.57	849.49
***q*-EGO**	841.51	313.14
2C-*q*-EGO	-47.64	675.35
BSP-EGO	-134.97	803.24

One can notice that the multi-criteria approach represented by the 2C-q-EGO algorithm is more appealing when larger batches of candidates are proposed. Indeed, in Fig. 4 ($n_{batch} = 8$), we can see that the predicted profits obtained with 2C-q-EGO at the end of the optimization are similar to BSP-EGO and TuRBO. TuRBO achieves faster improvement of the profit at the beginning of the search, but seems to remain stuck in local maxima for some optimization runs. Again, its standard deviation is higher than other methods, which supports the hypothesis of over-intensification of local optima. The one-way analysis of variance shows significant difference (p-value ≈ 0.01) between average final profits. Regarding Fig. 4, we identify a clear preference for q-EGO which almost always gives positive profits. The consistency of the results, described by the standard deviation in Tab. V, as well as the results themselves are significantly better than contestant approaches. Indeed, the pairwise comparison with Student's t-tests allows to reject the equality of means with confidence over 95% between q-EGO and the three other approaches.

Another level of interpretation can be provided regarding the parallel efficiency. Both versions of q-EGO seem to benefit from higher parallelization, and the multi-criteria algorithm seems to make sense when many candidates must be sampled. BSP-EGO also appears to benefit from a wider batch size, achieving significantly better profits in average with higher batch sizes. However no clear difference is visible regarding the TuRBO curves or average profits between $n_{batch} = 4$ and $n_{batch} = 8$.

V. Discussion

Even though the quantification of execution time differences is difficult due to different implementations of the surrogate models and a variable workload of the used computing cluster, some noticeable differences must be stated. First, TuRBO relies on sub-models learnt on a subset of the data and on TS for the candidate selection which makes it the most time efficient algorithm of the study. TuRBO should be practical to larger batch sizes within acceptable time. On the other side, q-EGO with its pseudo-sequential AP is the most time-consuming algorithm of this set, and would not be suitable for larger parallelization within such a restricted timing. Although the multi-criteria q-EGO (denoted 2C-q-EGO) decreases the AP time-cost, the cost of learning a global model from a large data set becomes prohibitive for the UPHES optimization. BSP-EGO succeeds in decreasing the AP cost thanks to its parallel operation. However, the global model fitting cost cannot be reduced and increases rapidly as the data set gets bigger. BSP-EGO has a much higher time-cost than TuRBO, but lower than both q-EGO algorithms.

TuRBO presents the best efficiency in terms of computational time, while providing good expected profits. However, it is outperformed by q-EGO in terms of quality of the final profit when $n_{batch} = 8$. The global metamodel might be a factor explaining its better performance compared to TuRBO. It is also assessed by the fact that BSP-EGO fills the gap with TuRBO when reaching $n_{batch} = 8$.

Multi-criteria q-EGO has been investigated with the objective of reducing the time cost of the AP. The global execution time is indeed reduced. However the introduction of a secondary AF delays the convergence towards good profits.

A second AF has also been integrated into BSP-EGO's AP, without leading to a significant difference with the single-criterion approach. Hence, it is not included in the results section. The two tested multi-criteria approaches are then not appropriate for small batch size ($n_{batch} \leq 8$) on the UPHES management problem.

In this work, the additional AF is used to lower the acquisition cost. However it could be used with the objective of adding diversification into the candidate selection and keep the partial update within the q-points selection loop to make the selection more relevant. Operating this way will not reduce execution time, but might improve the search and find better solutions at the end.

VI. Conclusions and Future Research Directions

In this paper, we have investigated three parallel BO algorithms: q-EGO, BSP-EGO and TuRBO. We have experimented and compared these algorithms on the UPHES management problem. The main observations from the reported results are summarized in the following.

Even though TuRBO is a recent BO algorithm addressing higher dimensional spaces (≈ 20), we can see on the UPHES management problem that a more conventional approach such as a fine-tuned version of q-EGO still outperforms TuRBO. Whereas the good performance of q-EGO on such application is highlighted by the reported results, it has to be noted that is less scalable than TuRBO due its sequential global AP.

A solution is proposed to tackle this problem, by adding a complementary AF in the AP, so that the number of model updates is reduced. However, the single-criterion EI-based q-EGO algorithm remains a more robust choice for all considered batch sizes. Indeed, it offers consistent performance in terms of objective values, while being always (in our testbed) among the best performing. Regarding the results of Section IV and the statistical analysis, q-EGO is the best option.

Even though BO considerably reduces the number of simulations needed to find a good solution, the optimization itself is time-consuming. Therefore it is crucial to find a balance between acquisition time and simulation time. BSP-EGO and TuRBO offer more possibilities to lower the acquisition time especially because their AP can be efficiently parallelized. However, a global model seems to be an advantage in the UPHES optimization task and relying on lower cost global models should be investigated within the q-EGO framework.

Trying to achieve better results in a very limited timing remains challenging for the UPHES management problem or any application with similar time constraints. Even though q-EGO is a suitable choice, approaches using subsets of data such as TuRBO can become attractive to better exploit parallel computing resources. It is planned to run the algorithms with higher batch sizes and q-EGO to confirm this hypothesis. The aspect of data set decomposition is applicable within the BSP-EGO framework but remains to be investigated. We also plan to investigate the impact of space filling methods for initial sampling and inside the optimization process in order to reduce the variance between optimization runs in terms of final cost.

References

[1] J.-F. Toubeau, J. Bottieau, Z. De Grève, F. Vallée, and K. Bruninx, "Data-driven scheduling of energy storage in day-ahead energy and reserve markets with probabilistic guarantees on real-time delivery," *IEEE Transactions on Power Systems*, vol. 36, no. 4, pp. 2815–2828, 2021.

[2] M. Gobert, J. Gmys, J.-F. Toubeau, F. Vallée, N. Melab, and D. Tuyttens, "Surrogate-assisted optimization for multi-stage optimal scheduling of virtual power plants," in *2019 International Conference on High Performance Computing Simulation (HPCS)*, 2019, pp. 113–120.

[3] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[4] D. Ginsbourger, R. Le Riche, and L. Carraro, "A multi-points criterion for deterministic parallel global optimization based on kriging," 03 2008.

[5] M. Gobert, J. Gmys, N. Melab, and D. Tuyttens, "Adaptive Space Partitioning for Parallel Bayesian Optimization," in *HPCS 2020 - The 18th International Conference on High Performance Computing Simulation*, Barcelona / Virtual, Spain, Mar. 2021. [Online]. Available: https://hal.inria.fr/hal-03121209

[6] D. Eriksson, M. Pearce, J. R. Gardner, R. Turner, and M. Poloczek, "Scalable global optimization via local bayesian optimization," 2020.

[7] L. V. L. Abreu, M. E. Khodayar, M. Shahidehpour, and L. Wu, "Risk-constrained coordination of cascaded hydro units with variable wind power generation," *IEEE Transactions on Sustainable Energy*, vol. 3, no. 3, pp. 359–368, 2012.

[8] J.-F. Toubeau, Z. De Grève, and F. Vallée, "Medium-term multimarket optimization for virtual power plants: A stochastic-based decision environment," *IEEE Transactions on Power Systems*, vol. 33, no. 2, pp. 1399–1410, 2018.

[9] R. Montero, T. Wortberg, J. Binias, and A. Niemann, "Integrated assessment of underground pumped-storage facilities using existing coal mine infrastructure," 07 2016, pp. 953–960.

[10] J.-F. Toubeau, Z. De Grève, P. Goderniaux, F. Vallée, and K. Bruninx, "Chance-constrained scheduling of underground pumped hydro energy storage in presence of model uncertainties," *IEEE Transactions on Sustainable Energy*, vol. 11, no. 3, pp. 1516–1527, 2020.

[11] E. Pujades, P. Orban, S. Bodeux, P. Archambeau, S. Erpicum, and A. Dassargues, "Underground pumped storage hydropower plants using open pit mines: How do groundwater exchanges influence the efficiency?" *Applied Energy*, vol. 190, pp. 135–146, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261916318608

[12] R. Ponrajah, J. Witherspoon, and F. Galiana, "Systems to optimize conversion efficiencies at ontario hydro's hydroelectric plants," *Power Systems, IEEE Transactions on*, vol. 13, pp. 1044 – 1050, 09 1998.

[13] Y. Pannatier, "Optimisation des stratégies de réglage d'une installation de pompage-turbinage à vitesse variable," 01 2010.

[14] C. Cheng, J. Wang, and X. Wu, "Hydro unit commitment with a head-sensitive reservoir and multiple vibration zones using milp," *IEEE Transactions on Power Systems*, vol. 31, no. 6, pp. 4842–4852, 2016.

[15] A. Arce, T. Ohishi, and S. Soares, "Optimal dispatch of generating units of the itaipu hydroelectric plant," *IEEE Transactions on Power Systems*, vol. 17, no. 1, pp. 154–158, 2002.

[16] J. P. S. Catalao, S. J. P. S. Mariano, V. M. F. Mendes, and L. A. F. M. Ferreira, "Scheduling of head-sensitive cascaded hydro systems: A nonlinear approach," *IEEE Transactions on Power Systems*, vol. 24, no. 1, pp. 337–346, 2009.

[17] P.-H. Chen and H.-C. Chang, "Genetic aided scheduling of hydraulically coupled plants in hydro-thermal coordination," *IEEE Transactions on Power Systems*, vol. 11, no. 2, pp. 975–981, 1996.

[18] B. Yu, X. Yuan, and J. Wang, "Short-term hydro-thermal scheduling using particle swarm optimization method," *Energy Conversion and Management*, vol. 48, no. 7, pp. 1902–1908, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0196890407000489

[19] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, Dec 1998. [Online]. Available: https://doi.org/10.1023/A:1008306431147

[20] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[21] D. Ginsbourger, R. L. Riche, and L. Carraro, "Kriging is well-suited to parallelize optimization," 2010.

[22] A. D. Palma, C. Mendler-Dünner, T. Parnell, A. Anghel, and H. Pozidis, "Sampling acquisition functions for batch bayesian optimization," 2019.

[23] M. Gobert, J. Gmys, N. Melab, and D. Tuyttens, "Space Partitioning with multiple models for Parallel Bayesian Optimization," in *OLA 2021 - Optimization and Learning Algorithm*, Sicilia / Virtual, Italy, Jun. 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03324642

[24] A. Artiba, V. Emelyanov, and S. Iassinovski, "Introduction to intelligent simulation: The rao language," *The Journal of the Operational Research Society*, vol. 51, 10 2000.

[25] J.-F. Toubeau, S. Iassinovski, E. Jean, J.-Y. Parfait, J. Bottieau, Z. De Greve, and F. Vallee, "A nonlinear hybrid approach for the scheduling of merchant underground pumped hydro energy storage," *IET Generation, Transmission Distribution*, vol. 13, 09 2019.

[26] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems 33*, 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/f5b1b89d98b7286673128a5fb112cb9a-Abstract.html

[27] R. Martinez-Cantin, "Bayesopt: A bayesian optimization library for nonlinear optimization, experimental design and bandits," *Journal of Machine Learning Research*, vol. 15, pp. 3735–3739, 11 2014.

[28] F. Rehbach, M. Zaefferer, B. Naujoks, and T. Bartz-Beielstein, "Expected improvement versus predicted value in surrogate-based optimization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, ser. GECCO '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 868–876. [Online]. Available: https://doi.org/10.1145/3377930.3389816