Applied Soft Computing Journal 151 (2024) 111134

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Investigating surrogate-based hybrid acquisition processes. Application to Covid-19 contact mitigation

G. Briffoteaux [a,b,*], N. Melab [b], M. Mezmaz [a], D. Tuyttens [a]

[a] Mathematics and Operational Research Department, University of Mons, Belgium
[b] Université de Lille, CNRS CRIStAL, Inria Lille, France

ARTICLE INFO

Keywords:
Surrogate-assisted optimization
Bayesian optimization
Expensive optimization
Parallel computing
Simulation-based optimization
Black-box optimization

ABSTRACT

Surrogate models are built to produce computationally efficient versions of time-complex simulation-based objective functions so as to address expensive optimization. In surrogate-assisted evolutionary computations, the surrogate model evaluates and/or filters candidate solutions produced by evolutionary operators. In surrogate-driven optimization, the surrogate is used to define the objective function of an auxiliary optimization problem whose resolution generates new candidates. In this paper, hybridization of these two types of acquisition processes is investigated with a focus on robustness with respect to the computational budget and parallel scalability. A new hybrid method based on the successive use of acquisition processes during the search outperforms competing approaches regarding these two aspects on the Covid-19 contact mitigation problem. To further improve the generalization to larger ranges of search landscapes, another new hybrid method based on the dispersion metric is proposed. The integration of landscape analysis tools in surrogate-based optimization seems promising according to the numerical results reported on the CEC2015 test suite.

1. Introduction

To solve black-box expensive optimization problems where the objective function is computationally costly to evaluate, Parallel Surrogate-Based Optimization Algorithms (P-SBOAs) [1] are built by leveraging parallel computing and machine learning. Machine learning [2] is employed to create surrogate models that provide computationally cheap predictions of the expensive objective function. Parallel computing allows to perform multiple expensive evaluations simultaneously. Two categories of P-SBOAs arise: Parallel Surrogate-Assisted Evolutionary Algorithms (P-SAEAs) [3] and Parallel Surrogate-Driven Algorithms (P-SDAs), also called Bayesian Optimization [4]. Both families of algorithms differ by their Acquisition Process (AP), the mechanism in charge of suggesting new promising candidate solutions. The AP from P-SDAs consists of optimizing an infill criterion that relies on the surrogate model and defines the promisingness of new candidate solutions [5]. The AP from P-SAEAs consists of utilizing the evolutionary operators to produce new solutions that are filtered out or predicted by the surrogate model to only retain the most promising ones and to save computational budget [6].

Computational expensiveness

In a previous study, we observed that P-SAEAs are generally recommended in the context of moderately expensive problems and that P-SDAs are usually preferred to deal with very expensive problems [7]. Very expensive problems are characterized by a computational budget limited to few hundred of expensive evaluations [8] while several ten thousand evaluations are affordable in the context of cheap problems [9]. Moderately expensive problems come in between with budgets amounting to few thousand expensive evaluations. The bounds to define each categories of problem's expensiveness are not clearly identified and may depend on the surrogate type and the search landscape. In case of moderately expensive problems, the archive of exactly evaluated solutions may grow significantly enough for the surrogate training to become non-negligible [10]. In this situation, it is not convenient to express the computational budget only as a limited number of objective function evaluations as it is commonly done in the field of surrogate-based optimization [8,11,12]. In this paper, we investigate the two following questions: What are the bounds (in terms of computational budget and objective function computational expensiveness) allowing to distinguish between moderately and very expensive problems? How can the surrogate model training costs be included into the computational budget?

COVID-19 contact mitigation problem

The P-SBOAs can be leveraged to address a large range of real-world problems from aerospace design [13] to public health [14]. At

* Corresponding author at: Mathematics and Operational Research Department, University of Mons, Belgium.
E-mail addresses: guillaume.briffoteaux@umons.ac.be (G. Briffoteaux), nouredine.melab@univ-lille.fr (N. Melab), mohand.mezmaz@umons.ac.be (M. Mezmaz), daniel.tuyttens@umons.ac.be (D. Tuyttens).

https://doi.org/10.1016/j.asoc.2023.111134
Received 6 June 2023; Received in revised form 28 November 2023; Accepted 4 December 2023
Available online 12 December 2023

the onset of the COVID-19 pandemic, it was of primary importance to question and study the strategies for contact mitigation to alleviate the detrimental effects of the disease on the populations [15]. For this purpose, numerical tools were designed to simulate the propagation and impact of the SARS-COV-2 which therefore made it possible to formulate black-box simulation-based optimization problems to identify the best possible decisions [16]. In this study, the real-world problem taken into consideration consists of finding the best per-age contact reduction plan to reduce mortality due to COVID-19 in Spain while attaining simulated herd immunity.

Hybrid P-SBOAs

A hybrid AP proposed in [17] suggests to both optimize an infill criterion (similarly to P-SDAs) and invoke reproduction operators (similarly to evolutionary algorithms). This parallel hybrid approach, called Surrogate Model Based Optimization Evolutionary Algorithm (SMBOEA), demonstrates a superiority compared to state-of-the-art P-SDAs for a number of computing cores $n_{cores} < 10$. However, it shows a serious limitation regarding parallel scalability as it performs similarly to a surrogate-free parallel evolutionary algorithm for $n_{cores} \geqslant 10$. Relying on numerous cores may not be beneficial to P-SBOAs as it is difficult to locate numerous promising solutions during one iteration because of the challenge of balancing exploration and exploitation [4]. Indeed, evaluating uninteresting candidates with the expensive objective function would waste the computational budget. How to hybridize P-SDAs and P-SAEAs to achieve robustness with respect to the computational budget? How to benefit from a large number of computing cores? These two questions are also addressed in this study.

The contributions reported in this article can be summarized as follows:

- Concerning the categories of optimization problem's expensiveness. We approximate the threshold, in terms of computational budget, that allows to distinguish between very and moderately expensive problems by running a computationally intensive grid search over the design space of P-SAEAs and P-SDAs. In order to take the surrogate training into account in the computational budget, a fixed wall-clock time in a capped number of computing units is set to run the optimization exercise.
- The computationally expensive COVID-19 contact mitigation problem is solved by P-SBOAs. A promising solution is pinpointed and a landscape analysis is conducted to shine a light on the landscape features hidden by the black-box objective function.
- We design hybrid P-SBOAs that retain the best of both P-SDAs and P-SAEAs to achieve robustness with respect to the computational budget. In particular, the HSAP strategy (Hybrid Successive Acquisition Processes) yields the best resolution of the COVID-19-related problem and demonstrates a reliable parallel scalability by efficiently leveraging up to 144 computing cores.
- As a first step towards landscape analysis-driven hybrid algorithms, we introduce DDHAP (Dispersion-Driven Hybrid Acquisition Process). The landscape analysis dispersion metric is analyzed to drive the selection of APs to be used during the search. This method demonstrates auspicious performances on a broad range of optimization problems featuring various landscape shapes.

Section 2 provides a background to the field of surrogate-based optimization by introducing surrogate models, APs and Promisingness Criteria (PC). The COVID-19 contact reduction problem and artificial benchmark problems are presented in Section 3. In Section 4, an intensive numerical grid search is conducted on the design space of P-SDAs and P-SAEAs. Section 5 focuses on solving of the COVID-19-related problem *via* hybrid methods such as HSAP. The generalization of hybrid methods to a larger range of problems is tackled in Section 6 by the specification of DDHAP. Finally, conclusions and directions for future works are outlined in Section 7.

2. Background on surrogate-based optimization

In this paper, we consider the minimization problem of locating $\boldsymbol{x}^*$ such that

$$\boldsymbol{x}^* = \arg\min_{x\in\Omega} f(x) \tag{1}$$

where the objective function $f : \Omega \subset \mathbb{R}^D \to \mathbb{R}$ is a computationally expensive function. In order to deal with this type of problems, a surrogate model $\hat{f}$ is built to predict f in a computationally cheap way. In this paper, we term the evaluation of f *expensive evaluation* while the evaluation of $\hat{f}$ is called *prediction*. The value returned by f at a given point $\boldsymbol{x}$ is called an "expensive objective value" while the one returned by $\hat{f}$ is a "predicted objective value". A point $\boldsymbol{x}$ lying in the search space Ω is referred to as "solution", "candidate" or "candidate solution" indistinctly.

2.1. Surrogate models

The surrogate model is based on a machine learning algorithm for interpolation or regression in order to imitate the expensive objective function. Five surrogate models are considered in this study because of their popularity in the field of surrogate-based optimization. They are briefly described hereafter.

Gaussian Processes are very appreciated surrogate models as they provide a reliable uncertainty information around their prediction [18–20]. The general idea of a Gaussian Process with Radial Basis Function kernel (GP_RBF) is to model the influence of one point $(\boldsymbol{x}, f(\boldsymbol{x}))$ on the prediction at another point $\boldsymbol{x}'$ by the kernel function defined by:

$$k_{RBF}(\boldsymbol{x}, \boldsymbol{x}') = \sigma \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2s^2}\right) \tag{2}$$

where σ and s are hyper-parameters. With this kernel, the influence of $(\boldsymbol{x}, f(\boldsymbol{x}))$ on the prediction at $\boldsymbol{x}'$ increases as the distance between $\boldsymbol{x}$ and $\boldsymbol{x}'$ decreases. By considering the training targets as random variables:

$$\begin{pmatrix} Y(\boldsymbol{x}^{(1)}) \\ \vdots \\ Y(\boldsymbol{x}^{(n)}) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix}, \ \Sigma\right) \tag{3}$$

and by applying the Bayes theorem, we can derive the prediction $\hat{f}(\boldsymbol{x}')$ and the standard deviation $\hat{s}(\boldsymbol{x}')$ around $\hat{f}(\boldsymbol{x}')$. The operation of training a GP is cubic in the number of training samples. More thorough details about GPs are given in [2,10].

The interpolation Ordinary Kriging model (iKRG) is a particular case of GP [10]. It is assumed that an observation $y = f(\boldsymbol{x})$ is the realization of a random variable $Y(\boldsymbol{x}) \sim \mathcal{N}(\mu, \sigma^2)$. The kernel function is given by:

$$k_{iKRG}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\sum_{i=1}^{D} \eta_i |x_i - x_i'|^{p_i}\right) \tag{4}$$

When $\boldsymbol{x}$ and $\boldsymbol{x}'$ move away from each other, their difference tends to infinity and their correlation tends to 0. Building the Ordinary Kriging model consists in estimating μ, σ^2 and the hyper-parameters $\boldsymbol{\eta}$ and $\boldsymbol{p}$ *via* likelihood maximization [5,21]. The training of iKRG is more computationally expensive than that of GP_RBF because of the larger number of hyper-parameters ($\boldsymbol{\eta}$ and $\boldsymbol{p}$).

The regression Ordinary Kriging model (rKRG) is obtained by adding a regularization term $\lambda \cdot I$ to the covariance matrix Σ:

$$\Sigma = K + \lambda \cdot I \tag{5}$$

where $K_{i,j} = k_{iKRG}(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$. This new term $\lambda \cdot I$ is treated as a hyper-parameter.

The Bayesian Linear Regressor with Artificial Neural Network as basis functions (ANN_BLR) [22] allows to reduce the computational cost entailed to build the surrogate while still providing uncertainty information around the prediction. It is defined as a linear combination of m_b non-linear functions ϕ_j of the input vector $\boldsymbol{x}$:

$$\hat{f}(\boldsymbol{x};\boldsymbol{w}) = \sum_{j=1}^{m_b} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) \tag{6}$$

where $\boldsymbol{w}$ are the parameters to be learned. In this particular case of linear model, the basis functions ϕ_j are given by the last hidden layer of an Artificial Neural Network [23]. The Bayesian training allows to derive uncertainty information around the prediction. The computational cost of building ANN_BLR is linear in the number of training data but cubic in the number of basis functions.

The Bayesian Neural Network approximated by Monte-Carlo Dropout (BNN_MCD) [24] further lowers the computational load of surrogate building while yielding a strong predictive capacity. The main principle behind BNN_MCD is to sample n_{sub} sub-networks $\hat{f}_i$ from a global Artificial Neural Network and to use the n_{sub} predictions to compute an average prediction and a standard deviation:

$$\hat{f}(\boldsymbol{x}') = \frac{1}{n_{sub}} \sum_{i=1}^{n_{sub}} \hat{f}_i(\boldsymbol{x}') \qquad \hat{s}(\boldsymbol{x}') = \sqrt{\frac{1}{n_{sub}} \sum_{i=1}^{n_{sub}} (\hat{f}_i(\boldsymbol{x}') - \hat{f}(\boldsymbol{x}'))^2} \tag{7}$$

The sub-networks are sampled by randomly deactivating neurons in the global network, thus approximating a Bayesian training. The operation of training a BNN_MCD is linear in the number of training samples, however, the uncertainty information is less reliable than the one provided by GPs and Kriging models.

2.2. *Acquisition processes*

The two categories of P-SBOAs, namely P-SAEAs and P-SDAs, differ by the coupling between the surrogate model and the optimizer [7].

In P-SAEAs, the surrogate is attached to the evolutionary algorithm by means of an Evolution Control that defines the promisingness of new candidate solutions [3]. The evolutionary algorithm carries out the search by evolving a population of candidates through the stages of selection, reproduction and replacement. The surrogate is introduced at one or multiple stages, also possibly at the initialization of the population, to replace the expensive objective function.

In this study, we use a Genetic Algorithm (GA) as the algorithm leading the search and three APs are considered depending on the role played by the surrogate. Fig. 1 depicts the Parallel Surrogate-Assisted GA (P-SAGA) framework. The algorithm begins by initializing a population and an archive with parallel expensive evaluations of solutions sampled through Latin Hypercube Sampling (steps 1 and 2 in Fig. 1). The surrogate is initialized based on this archive. The population is then evolved by an iterative process stopping when the computational budget is exhausted. During one iteration, solutions are selected from the population and recombined *via* reproduction (step 3) to generate a set of new candidates (step 4). According to the Evolution Control (step 5), the n_{disc} less promising new candidates are discarded and the q most promising ones go through parallel expensive evaluations (step 6). The archive is enriched with the newly expensive evaluations and the surrogate is updated (step 7). The n_{pred} remaining new candidates are predicted by the updated surrogate (step 8) and participate to replacement along with the q expensive evaluations (step 9) and the solutions from the population.

In this framework, the surrogate plays the role of an evaluator and/or a filter. When $q, n_{disc}, n_{pred} \neq 0$, it plays both roles as it filters out solutions considered as unpromising and it predicts moderately promising solutions. This AP is called Surrogate as an Evaluator and a Filter (SaaEF) [25]. When $n_{disc} = 0$, the surrogate is only employed as an evaluator (SaaE) [14,26] while $n_{pred} = 0$ implies the unique role of a filter (SaaF) [27]. In SaaE and SaaEF, a high degree of confidence is granted to the surrogate model as predicted solutions are allowed into the population. While the risk of misleading the search appears for inaccurate surrogates, the benefit lies in the saving of computational budget. The idea behind SaaF is to give more opportunity to the reproduction operators to come up with promising solutions.

In P-SDAs, whose generic diagram is given in Fig. 2, the surrogate is used to define a metric of promisingness called the Infill Criterion that is optimized to locate new promising candidates. At each iteration, multiple candidates are acquired and evaluated in parallel with the expensive function. The surrogate is updated thanks to the new expensive evaluations and the iterative procedure stops when the computational budget is depleted [28].

The q-point Efficient Global Optimization algorithm (qEGO) is a popular P-SDA proposed under two variant APs, namely, *Constant Liar* (CL) and *Kriging Believer* [29]. The acquisition of q points *via* CL is described in Algorithm 1. In CL, $q > 1$ auxiliary optimization problems are solved sequentially at each iteration in order to propose q new candidate solutions for parallel expensive evaluations (lines 4–9 in Algorithm 1). One iteration also involves q sequential updates of the surrogate model, among which $q-1$ updates are based on an artificially-constructed training set to prevent redundancy in the batch of new candidates. Relying on the mean objective value in the archive to enrich the artificially-constructed training set (line 3 and 7) provides a trade-off between exploration and exploitation. Indeed, using the minimum objective value would attract the next auxiliary optimization exercise in the area around the last acquired point. Reversely, employing the maximum value would repel the subsequent sampling from this area.

Kriging Believer follows the same scheme as CL with the only exception of using the prediction of the last acquired point $\hat{f}(\boldsymbol{x}^{(i)})$ instead of L to enrich the artificially-constructed archive in line 7 of Algorithm 1.

Algorithm 1 *Constant Liar* AP for qEGO

```
Input
    archive: set of solutions already evaluated by the expensive function
    $IC$: infill criterion
    surrogate: surrogate model
    $q$: number of candidates to sample
 1: $\mathcal{B}_{sim} \leftarrow \emptyset$                          ▷ batch of new candidates
 2: tmp_archive ← copy(archive)
 3: $L \leftarrow \text{mean}_{y \in archive}(y)$
 4: for $i = 1 : q$ do
 5:     $\boldsymbol{x}^{(i)} \leftarrow$ optimize($IC$, surrogate)
 6:     $\mathcal{B}_{sim} \leftarrow \mathcal{B}_{sim} \cup \boldsymbol{x}^{(i)}$
 7:     tmp_archive ← tmp_archive $\cup \{(\boldsymbol{x}^{(i)}, L)\}$
 8:     update_surrogate(surrogate, tmp_archive)
 9: end for
10: return $\mathcal{B}_{sim}$
```

2.3. *Promisingness criteria*

The Evolution Control, in the framework of P-SAEAs, or the Infill Criterion, in the case of P-SDAs, define what is a promising solution. As these two terminologies point out the same concept, we choose the global denomination "Promisingness Criterion" (PC) in this article. Let $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ be two candidate solutions and $>_p$ the comparison operator defining the PC:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow \boldsymbol{x}^{(1)} \text{ is more promising than } \boldsymbol{x}^{(2)}$$

Let $\hat{f}$ and $\hat{s}^2$ be the predicted objective value (mean) and the uncertainty (variance) as delivered by the surrogate respectively. Let d_2 be the Euclidean distance, in search space, to the archive of solutions already evaluated by the expensive function. For a minimization problem, the PC can then be defined by:

- Predicted Objective Value (*pov*) [6]:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow \hat{f}(\boldsymbol{x}^{(1)}) < \hat{f}(\boldsymbol{x}^{(2)})$$

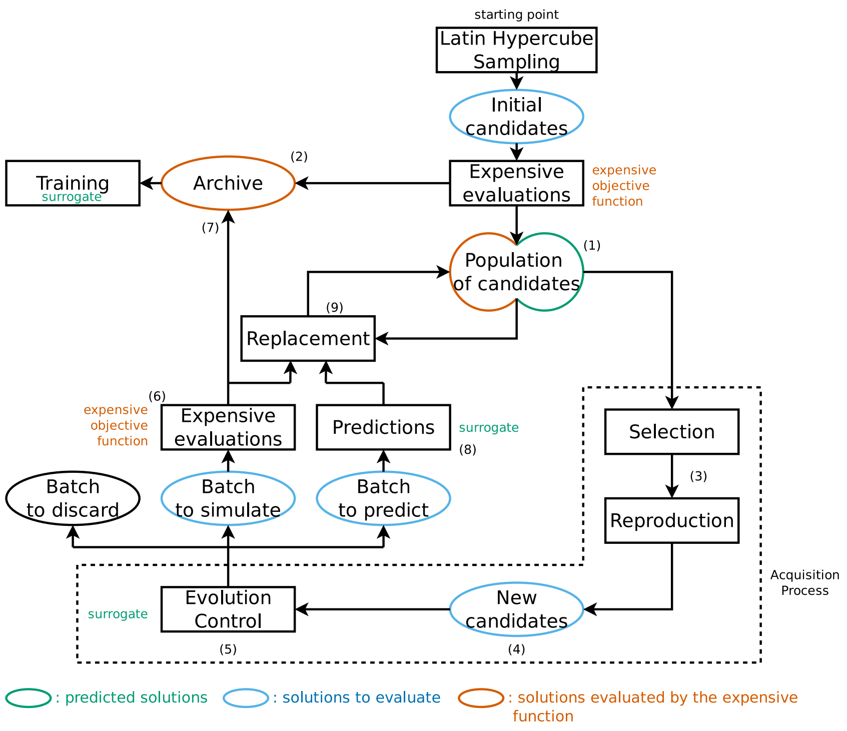

Fig. 1. P-SAGA framework. The ellipses represent the sets of candidate solutions along with their objective values or not, while the rectangles stand for the operators. Steps 3–9 are repeated until the computational budget is exhausted.

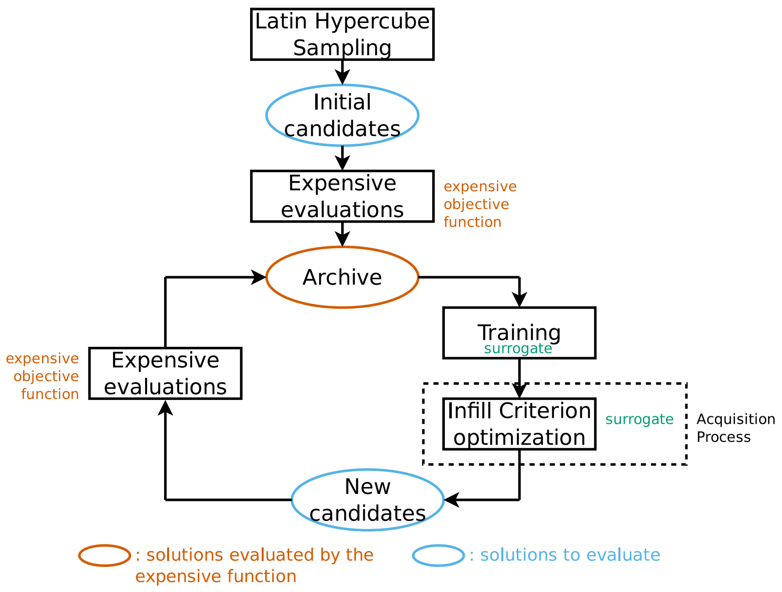

Fig. 2. P-SDA framework. The ellipses represent the sets of candidate solutions along with their objective values or not, while the rectangles stand for the operators. Iterations are stopped when the computational budget is exhausted.

- Distance to the archive (*dist*) [14]:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow d_2(\boldsymbol{x}^{(1)}) > d_2(\boldsymbol{x}^{(2)})$$

- Standard Deviation on predictions (*stdev*) [14]:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow \hat{s}(\boldsymbol{x}^{(1)}) > \hat{s}(\boldsymbol{x}^{(2)})$$

- Expected Improvement (*ei*) [28]:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow EI(\boldsymbol{x}^{(1)}) > EI(\boldsymbol{x}^{(2)})$$

$$EI(\boldsymbol{x}) = \begin{cases} (y_{min} - \hat{f}(\boldsymbol{x})).\Phi_{\mathcal{N}}\left(\frac{y_{min}-\hat{f}(\boldsymbol{x})}{\hat{s}(\boldsymbol{x})}\right) + \hat{s}(\boldsymbol{x}).\phi_{\mathcal{N}}\left(\frac{y_{min}-\hat{f}(\boldsymbol{x})}{\hat{s}(\boldsymbol{x})}\right) & \text{for } \hat{s}(\boldsymbol{x}) > 0 \\ 0 & \text{for } \hat{s}(\boldsymbol{x}) = 0 \end{cases}$$

where y_{min} is the best expensive objective value currently known and $\phi_{\mathcal{N}}$ is the probability density function of $\mathcal{N}(0,1)$.

- Lower Confidence Bound (*lcb*) [12,30]:

$$\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow \hat{f}(\boldsymbol{x}^{(1)}) - \lambda_{lcb}.\hat{s}(\boldsymbol{x}^{(1)}) < \hat{f}(\boldsymbol{x}^{(2)}) - \lambda_{lcb}.\hat{s}(\boldsymbol{x}^{(2)})$$

While *pov* favors exploitation by relying on the regions featuring small predicted objective values, *dist* and *stdev* promote exploration by favoring the distant or uncertainly predicted regions of the search space. Expected Improvement represents the expectation of the area enclosed by the surrogate predicting distribution below the best expensive objective value y_{min} found hitherto. The first part of the sum is the cumulative probability of improvement multiplied by the amount of improvement. The second part of the sum is the probability of improvement multiplied by the standard deviation on the prediction. Over the search space, *ei* is high in regions of improvement thus enhancing exploitation, and high in regions of high uncertainty thus promoting exploration. It is also high in regions that represent both a moderate degree of improvement and uncertainty. *ei* prevents resampling by having zero value for candidates already evaluated with the expensive function ($\hat{s}(\boldsymbol{x}) = 0$). The Lower Confidence Bound also allows for a trade-off between exploration and exploitation by setting the parameter $\lambda_{lcb} \in \mathbb{R}^{+}$. A small value of λ_{lcb} promotes exploitation while a larger value incorporates $\hat{s}$ to include exploration.

In addition to the aforementioned scalar criteria, there exists different ways of generating ensembles of criteria.

- Pareto-based criteria with crowding distance (*par-fs-cd*, *par-fd-cd*) [31]:

$$\begin{aligned}&\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \text{ if}\\&NDR(\boldsymbol{x}^{(1)}) < NDR(\boldsymbol{x}^{(2)}) \text{ or}\\&[NDR(\boldsymbol{x}^{(1)}) = NDR(\boldsymbol{x}^{(2)}) \text{ and } cd(\boldsymbol{x}^{(1)}) > cd(\boldsymbol{x}^{(2)})]\\&\text{for the bi-objective problems } (min\ \hat{f}, max\ \hat{s}) \text{ or } (min\ \hat{f}, max\ d_2)\end{aligned} \tag{8}$$

  where NDR is the non-dominated rank and $cd()$ is the crowding distance [32]. If both non-dominated ranks are equal and both crowding distances are equal, the most promising candidate is the one producing the lowest predicted objective value.
- Pareto-based criteria from Tian et al. [18] (*par-tian-fs*, *par-tian-fd*):

$$\begin{aligned}&\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \text{ if}\\&[NDR(\boldsymbol{x}^{(1)}) = 1 \text{ and } NDR(\boldsymbol{x}^{(2)}) \neq 1] \text{ or}\\&[NDR(\boldsymbol{x}^{(1)}) = t \text{ and } NDR(\boldsymbol{x}^{(2)}) \notin \{1, t\}] \text{ or}\\&[NDR(\boldsymbol{x}^{(1)}) < NDR(\boldsymbol{x}^{(2)}) \text{ and } NDR(\boldsymbol{x}^{(1)}), NDR(\boldsymbol{x}^{(2)}) \notin \{1, t\}]\\&\text{for the bi-objective problems } (min\ \hat{f}, max\ \hat{s}) \text{ or } (min\ \hat{f}, max\ d_2)\end{aligned} \tag{9}$$

  where t is the number of non-dominated fronts.
- Dynamic exclusive bi-criterion [7]: The search is divided into two equal periods according to the computational budget.
  - *dyn-df-excl*: first *dist*, then *pov* (from exploration to exploitation)
  - *dyn-fd-excl*: first *pov*, then *dist* (from exploitation to exploration)
- Dynamic inclusive criteria [31] (*dyn-df-incl*): Two criteria, *dist* and *pov* participate concurrently according to participation rates $p_1^{(i)}$ and $p_2^{(i)}$ respectively. The search is divided into five equal periods according to the computational budget. The participation rates are set at each period i of the search according to:

$$(p_1^{(i)}, p_2^{(i)}) = \left(1 - \frac{i}{4}, \frac{i}{4}\right) \quad i = 0, \dots, 4$$

  which corresponds to a switch from exploration to exploitation during the search.
- Adaptive criterion from Wang et al. [19] (*ada-wang*):

$$\begin{aligned}&\boldsymbol{x}^{(1)} >_p \boldsymbol{x}^{(2)} \Leftrightarrow f_{ada-wang}(\boldsymbol{x}^{(1)}) < f_{ada-wang}(\boldsymbol{x}^{(2)})\\&f_{ada-wang}(\boldsymbol{x}) = (1-\alpha)\frac{\hat{f}(\boldsymbol{x})}{\max_{c \in \mathcal{B}} \hat{f}(\boldsymbol{c})} + \alpha \frac{\hat{s}(\boldsymbol{x})}{\max_{c \in \mathcal{B}} \hat{s}(\boldsymbol{c})}\\&\alpha = -\frac{1}{2}\cos\left(\frac{b_s}{b}\pi\right) + \frac{1}{2}\end{aligned} \tag{10}$$

  where $\mathcal{B}$ is a batch of new candidates, b_s is the amount of computational budget already spent and b is the total budget. At the beginning of the search $\alpha \approx 0$ thus promoting exploitation by minimization of the predicted objective value. As the search proceeds, $\alpha \to 1$, thus further reinforcing exploitation by minimizing $\hat{s}$.
- Voting Committees (*com-dpf*, *com-spf*) [7]: The most promising candidates are those receiving the larger number of votes considering a committee of three criteria C_1, C_2, C_3. The choice between two candidates with the same number of votes is made randomly.
  - *com-dpf*: C_1: *dist*, C_2: *par-fd-cd*, C_3: *pov*
  - *com-spf*: C_1: *stdev*, C_2: *par-fs-cd*, C_3: *pov*

Table 1
Summary of numerical experiments with the associated subsections and problem suites.

Subsection	Numerical experiments	Problems suite
4.1	Off-line comparison of surrogates	Schwefel–Rastrigin -Rosenbrock & COVID-19
4.2	Grid-search on P-SAGA and qEGO design space	Schwefel–Rastrigin -Rosenbrock & COVID-19
5.4	Hybrid methods on the COVID-19-related problem	COVID-19
6.1, 6.2	Generalizable hybrid methods	CEC2015

3. Benchmark and real-world problems

Multiple optimization problems are considered in the numerical experiments reported in the next sections of this paper. This section aims at describing these optimization problems and Table 1 summarizes the numerical experiments and the associated subsections where the different problem suites are employed.

3.1. Artificial benchmark problems

3.1.1. Schwefel–Rastrigin–Rosenbrock

This test suite is composed of the three artificial functions Schwefel, Rastrigin and Rosenbrock with 16 decision variables. The analytical formulas are given in [7] while the graphs of the 2-D variants are given in Fig. A.14, Fig. A.15 and Fig. A.16 in Appendix A for the sake of illustration. These graphs exhibit different features known to hamper the search process as multi-modality (Schwefel and Rastrigin), weak global structure (Schwefel) and connected valley (Rosenbrock). The weak global structure refers to a rugged landscape lacking an underlying general structure. In other terms, smoothing such a landscape would not provide any indication about the location of the optima. Reversely, the strong global structure as provided by the Rastrigin function is governed by a general shape that may be grasped by an adequately tuned regression model. This test suite is used in this article to calibrate the algorithms as the low number of problems allows to compare a large number of possible algorithms configurations.

3.1.2. CEC2015 test suite

The CEC2015 test suite was proposed for an international competition on bound constrained single-objective computationally expensive numerical optimization [33]. It is composed of 15 artificial objective functions with 10 and 30 decision variables, spanning a large range of landscape characteristics such as uni-modality, multi-modality, non-separability and weak-global structure. Hybrid and composite functions embedding the Rastrigin, Rosenbrock and Schwefel functions are notably included. In this study, we rely on the CEC2015 test suite to test the generalization of algorithms.

For both the Schwefel–Rastrigin–Rosenbrock and the CEC2015 test suite, the terminology "expensive objective function" is kept because an artificial wall clock time is added to every objective function evaluation to simulate the context of expensive evaluation.

3.2. COVID-19 contact reduction problem

At the beginning of the COVID-19 crisis, when no vaccines were available, governments of the affected countries adopted different strategies to contain the spread of the virus. While some countries imposed lockdown and physical distancing, others, bet on reaching herd immunity by natural transmission. This approach has not proven to be effective during the first two years of the epidemic [34]. However, at the time, studying the possible consequences of this strategy was of importance. For potential new outbreaks appearing in the future, herd immunity might be acquired by natural transmission which therefore would make this study an asset.

The problem consists of optimizing the contact reduction strategy to minimize the number of COVID-19-related deaths in Spain while reaching herd immunity. The Spanish population is divided into 16 age-categories and the decision variables represent the contact mitigation factors to apply to each category. For a decision vector $\boldsymbol{x} \in [0,1]^{16}$, $f_1(\boldsymbol{x})$ represents the simulated number of deaths after the considered period and $f_2(\boldsymbol{x}) \in \{0,1\}$ is a simulated boolean variable indicating whether herd immunity has been reached. The optimization problem consists in finding $\boldsymbol{x}^*$ such that:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}\in[0,1]^{16} \text{ s.t. } f_2(\boldsymbol{x})=1}{\arg\min} f_1(\boldsymbol{x}) \tag{11}$$

According to [35], handling constrained problems with evolutionary algorithms can be realized by different means. For our problem, it is not known how to generate feasible candidates so designing repairing operators or specific reproduction operators is impossible. Rejecting infeasible individuals would prevent to keep knowledge about the infeasible region location, besides, this technique works only if the search space is convex, that is probably not the case. Adding the amount of infeasibility as an additional objective would increase the complexity of the problem because the new objective would be boolean. Finally, we opt for the penalization of the infeasible candidates to handle the constraint of the COVID-19 contact reduction problem. The penalty value is set to the approximate Spanish population size (46,000,000) as it is the only *a priori* known upper bound for f_1. Even if this value could seem unrealistic, the optimization algorithm is expected to locate the basins of interest of the penalized function, which corresponds to the feasible region of the original problem. Therefore, the problem is re-formulated as an unconstrained optimization problem by applying a penalty to the objective f_1 when herd immunity is not reached. The re-formulated problem thus consists in finding $\boldsymbol{x}^*$ such that:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}\in[0,1]^{16}}{\arg\min} \tilde{f}(\boldsymbol{x}) \tag{12}$$

where:

$$\tilde{f}(\boldsymbol{x}) = \begin{cases} f_1(\boldsymbol{x}) \text{ if } f_2(\boldsymbol{x}) = 1 \\ f_1(\boldsymbol{x}) + 46,000,000 \text{ if } f_2(\boldsymbol{x}) = 0 \end{cases} \tag{13}$$

The impact of the contact reduction strategy is simulated thanks to the AuTuMN simulator available at https://github.com/monash-emu/AuTuMN/ [36]. Both quantities f_1 and f_2 are obtained *via* resolution of differential equations governing the flow of individuals in a compartmental model where the population is divided according to the disease state (Susceptible, Exposed, Infectious, Recovered) [37]. The graph of f_1 is expected to be multi-modal with flat regions according to the prior knowledge issued by the developers of AuTuMN. The simulation takes place in three phases. First, the past dynamic of the epidemic is analyzed by calibrating uncertain parameters according to past information. Age-dependent metrics such as fatality rates, hospital proportions uncertainty and symptomatic proportions uncertainty are calibrated while considering the daily numbers of COVID-19 confirmed cases as calibration targets. Second, the contact reduction strategy is applied during a period of 12 months. After the 12-month period, mobility restrictions are lifted and population herd immunity is recognized if incidence still decreases after two weeks while assuming persistent immunity for recovered individuals [38]. Different ways of modeling herd immunity could be thought of such as measuring the gradient of incidence to propose another objective in a bi-objective problem. These interesting perspectives involve additional complexities that we suggest to investigate as future work. The degrees of contact between individuals are integrated into the model through the contact matrix C provided by [39] where the populations are divided into 16 age-categories. $C_{i,j}$ is the average number of contacts per day that an individual of age-group j makes with individuals of age-group i. The decision variables of the COVID-19 contact reduction problem represent the mitigation factors. They are applied to matrix C such that $C_{i,j}$ is replaced by $x_i \cdot x_j \cdot C_{i,j}$. A decision variable $x_i = 0$ impedes any contact to individuals from age-category i while setting $x_i = 1$ lets the contact rates unchanged compared to the pre-COVID-19 era. There could exist some correlations among the mitigation factors for close age-categories but these are not known *a priori* because a lot of parameters, including those calibrated at the first phase and those from matrix C, are age-dependent. More details about the simulation phase and notably the underlying differential equations are given in [38].

4. Investigating the design of P-SBOAs

Three main design choices need to be made to build actual implementations of P-SBOAs: the surrogate model, the Acquisition Process and the Promisingness Criterion. In this section, we analyze empirically these possible design choices and their combination with respect to the search landscape characteristics provided by the Schwefel–Rastrigin–Rosenbrock test suite and the COVID-19 contact mitigation problem.

4.1. Offline comparison of surrogates

In this preliminary step to the design of P-SBOAs, we propose to compare the five surrogates presented in Section 2.1, GP_RBF, iKRG, rKRG, ANN_BLR and BNN_MCD with respect to global prediction accuracy and training time.

The BNN_MCD was calibrated by a grid search reported in Appendix B. The values retained for the BNN_MCD hyper-parameters are exposed in Table 4. The remaining surrogates are tuned according to the literature and the default parameters of the selected libraries. The GP_RBF is implemented thanks to the GPyTorch library [40]. The interpolation and regression Kriging models are set up *via* the pyKriging package [41]. The ANN_BLR is tuned and implemented according to [22].

Two training sets of 72 and 256 samples, and a validation set of 1024 samples are generated by Latin Hypercube Sampling for the three artificial problems. It is worth mentioning that, because of the small size of the training sets and the hard-to-approximate landscapes (16-D, multi-modal, weak global structure...), we do not expect the surrogates to provide reliable approximations over the whole search space. In other terms, we do not expect good validation mean squared errors (MSE). Nevertheless, this experiment should produce broad indications about a surrogate's training time and approximation quality. A surrogate fitting perfectly the global search landscape with few training samples would indicate an easy-to-optimize problem that would not require iterative surrogate-update-based algorithms as those investigated in this paper. Through the AP, we rather expect a surrogate to reproduce as accurately as possible the comparison operation between two possible candidate solutions. In case a surrogate seems to fit very well globally with few training samples, it is advisable to optimize the surrogate predictions only and evaluate the optimization output with the expensive objective function as in [42]. To best train a machine learning model to fit globally, active learning techniques can be used [43], but this falls out of the scope of this paper.

To compare the models, the training is repeated ten times for each of the five models and the training sets, and the averaged training time

Table 2
Offline comparison of surrogates. Training time (TT) and validation mean squared error (vMSE) averaged over 10 runs for each surrogate and each **benchmark problem**. Ranks according to TT and vMSE are denoted in parentheses.

Surrogates	72 training samples		256 training samples	
	TT	vMSE	TT	vMSE
		Schwefel		
BNN_MCD	6.56(2)	**6.1e+05(1)**	**7.09(1)**	**5.9e+05(1)**
ANN_BLR	**3.99(1)**	1.1e+06(2)	1.3e+01(2)	8.8e+05(2)
GP_RBF	4.0e+01(4)	3.7e+07(3)	8.9e+01(3)	3.7e+07(3)
rKRG	2.2e+01(3)	6.3e+14(4)	2.6e+02(4)	9.4e+14(4)
iKRG	1.2e+02(5)	6.3e+14(4)	6.3e+14(4)	9.4e+14(4)
		Rastrigin		
BNN_MCD	6.6(2)	1.6e+03(2)	**8.3(1)**	1.592e+03(2)
ANN_BLR	**4.0(1)**	2.8e+03(3)	1.3e+01(2)	2.5e+03(3)
GP_RBF	4.1e+01(4)	**1.59e+03**(1)	1.0e+02(3)	**1.590e+03(1)**
rKRG	1.5e+01(3)	3.3e+09(4)	2.3e+02(4)	4.35e+09(4)
iKRG	8.2e+01(5)	3.3e+09(4)	1.5e+03(5)	4.36e+09(5)
		Rosenbrock		
BNN_MCD	7.38(2)	**6.9e+11(1)**	**7.21(1)**	6.8e+11(2)
ANN_BLR	**4.03(1)**	1.1e+12(2)	1.4e+01(2)	**5.5e+11(1)**
GP_RBF	4.1e+01(4)	4.3e+12(3)	9.2e+01(3)	4.3e+12(3)
rKRG	1.1e+01(3)	3.5e+25(4)	3.7e+02(4)	6.4e+25(4)
iKRG	7.9e+01(5)	3.6e+25(5)	1.8e+03(5)	7.0e+25(5)

Table 3
Offline comparison of surrogates. Normalization effect on GP_RBF training for a training set of size 72. Training time (TT) and validation mean squared error (vMSE) averaged over 10 runs for each **benchmark problem**.

Normalization	TT	vMSE	TT	vMSE	TT	vMSE
	Schwefel		Rastrigin		Rosenbrock	
None	4.0e+01	3.7e+07	4.1e+01	1.6e+03	4.1e+01	4.3e+12
[0, 1]	4.1e−01	6.2e+05	2.8e−01	1.7e+03	2.8e−01	5.4e+11

and MSEs computed on the validation set are presented in Table 2. According to the reported results, the BNN_MCD is ranked 1st or 2nd according to both training time and validation MSE in all the cases. It seems better suited than the other models to approximate the Schwefel function. The GP_RBF is the best predictor of the Rastrigin function but shows a higher training time than ANN_BLR and BNN_MCD. Indeed the ANN_BLR is the fastest to train on a training set of 72 samples but increasing the training set size significantly impacts its training time when compared to BNN_MCD that demonstrates no significant training time increase from 72 to 256 training samples. The Kriging models perform poorly regarding both training time and validation MSE compared to the other models. While Kriging training time is known to be substantial because of the large number of hyper-parameters, the poor prediction accuracy may be surprising. Based on this observation, the Kriging models are discarded in the rest of this study.

To alleviate the high training time implied by the GP_RBF, the training data is normalized and the new results are reported in Table 3. Normalizing the training data relieves the training task as the search for the hyper-parameters is stopped prematurely due to the early stopping mechanism implemented in GPyTorch. The prediction accuracy provided by the GP_RBF trained on normalized data seems not to be badly affected as shown in Table 3. On the contrary, GP_RBF predictions are even improved on the Rosenbrock and the Schwefel functions.

4.2. Grid search on P-SAGA and qEGO design spaces

The grid search over the design space of P-SBOAs assumes the following three dimensions: the surrogate, the AP and the PC. In this study, the Parallel Surrogate-Assisted Genetic Algorithm (P-SAGA) [26, 27] and the q-point Efficient Global Optimization (qEGO) [29] are considered as the representatives of P-SAEAs and P-SDAs respectively.

4.2.1. Experimental protocol

The surrogate-free parallel GA is applied on the benchmark and represents a baseline to test the impact of surrogate-based approaches. Its calibration is displayed in Table 4 and the tuning of its population size and cross-over probability is reported in Appendix C.

The GP_RBF, ANN_BLR and BNN_MCD are the possible choices of surrogate models in the grid search. Two variants of GP_RBF are employed, GP_RBF_RTS is trained on the last 72 expensive evaluations and GP_RBF_CTS is trained on the whole archive of known expensive evaluations. ANN_BLR is trained on the last 256 expensive evaluations while BNN_MCD is trained on the whole archive. Three variants of BNN_MCD differing by the number of sub-networks for prediction $n_{sub} \in \{5, 20, 100\}$ were also considered. All the BNN_MCD instances used in the numerical experiments reported in this paper consider $n_{sub} = 5$ as it provided the best results among the three tested values.

The selectable APs are the three variants of P-SAGA and the two variants of qEGO presented in Section 2.2 (SaaE, SaaF, SaaEF, CL and *Kriging Believer*). Because the Kriging models are not considered as surrogate, we take the liberty to rename *Kriging Believer* into *Surrogate Believer* (SB). The parameters' values for SaaEF, SaaE and SaaF are exposed in Table 4 while the calibration of SaaEF is described in Appendix D. The main difference between SaaE and SaaEF is the reduction of n_{chld} that results in less opportunity for the reproduction operators to come up with promising candidates. In qEGO, the optimization of the PC is realized by a surrogate-free GA whose parameters are set as in Table 4 except for the population size ($n_{pop} = 50$) and the number of generations ($n_{gen} = 100$) that are tuned by a grid search reported in Appendix E.

The 15 criteria described in Section 2.3 are employed as PC except *dyn-df-incl* that is not considered in qEGO as it would be equivalent to *dyn-df-excl*.

The algorithm configurations retained for the grid search are summarized in Table 5. To limit the computational task of the grid search, inauspicious combinations were removed as soon as first results were made available. It is for example the case for (BNN_MCD, SaaE, Rastrigin) or (GP_RBF_CTS, SB, COVID-19). Each algorithm instance is run 10 independent times on every targeted problem for a computational budget of 30 min on 18 computing cores while assuming an evaluation time of 15 s.

Table 4
Calibration of the algorithms.

Symbol	Name	Value	Calibration method
	Calibration of BNN_MCD		
n_{sub}	number of sub-networks	5	grid search
n_{hl}	number of fully-connected hidden layers	1	grid search
m_u	number of units per layer	1024	grid search
λ_{decay}	weight decay coefficient	10^{-1}	grid search
l	normal standard deviation for weights initialization	10^{-2}	grid search
p_{drop}	dropout probability	0.1	grid search
$h()$	activation function	Relu	[23]
ξ	Adam initial learning rate	0.001	[23]
	Calibration of GA		
n_{pop}	population size	72	grid search
p_c	cross-over probability	0.9	grid search
η_c	cross-over distribution	10	[44]
p_m	mutation probability	$\frac{1}{d}$	[45]
η_m	mutation distribution	50	[44]
n_t	tournament size	2	[32]
	Calibration of SaaEF		
n_{chld}	children/iteration	288	grid search
q	exp. eval./iteration	$72 = 0.25 * n_{chld}$	[26,46]
n_{pred}	predictions/iteration	$72 = 0.25 * n_{chld}$	[26,46]
n_{disc}	discardings/iteration	216	$n_{chld} - q - n_{pred}$
	Calibration of SaaF		
n_{chld}	children/iteration	288	same as SaaEF
q	exp. eval./iteration	72	same as SaaEF
n_{pred}	predictions/iteration	0	
n_{disc}	discardings/iteration	216	$n_{chld} - q - n_{pred}$
	Calibration of SaaE		
n_{chld}	children/iteration	144	$q + n_{pred}$
q	exp. eval./iteration	72	same as SaaEF
n_{pred}	predictions/iteration	72	[26,46]
n_{disc}	discardings/iteration	0	

Table 5
Grid search on P-SAGA and qEGO design space. Algorithm configurations. The * symbol refers either to the whole set of surrogates {GP_RBF_RTS, GP_RBF_CTS, ANN_BLR, BNN_MCD}, to the whole set of 15 PC, or to the whole set of problems {Schwefel, Rastrigin, Rosenbrock and COVID-19}.

Surrogates	AP	PC	Problems
∅	GA	∅	*
* \ {GP_RBF_CTS}	SaaEF	*	*
BNN_MCD	SaaE	*	Schwefel, COVID-19
GP_RBF_RTS	SaaE	*	Rastrigin, Rosenbrock
BNN_MCD	SaaF	*	Schwefel, COVID-19
GP_RBF_RTS	SaaF	*	Rastrigin, Rosenbrock
*	CL	* \ {*dyn-df-incl*},	*
* \ {GP_RBF_CTS}	SB	* \ {*dyn-df-incl*},	*\ COVID-19

4.2.2. Analysis of experimental results

Table 6 exposes the top-5 ranking at the end of the search according to the objective value averaged over the ten repetitions for the Schwefel, Rastrigin, Rosenbrock and COVID-19 problems. The convergence profiles are exposed in Fig. 3, Fig. 4, Fig. 5 and Fig. 6 respectively. They represent the best identified expensive objective value, averaged over the ten repetitions, with respect to the number of expensive evaluations. Only the most interesting curves are displayed for readability purpose: those that demonstrate the minimum averaged best objective value for any number of expensive evaluations. The length of the curves informs about the computational efficiency of the approaches with longer curves indicating more efficient approaches. In this respect, qEGO is much more computationally expensive than P-SAGA due to the q sequential auxiliary optimizations and surrogate trainings involved at each iteration. The computational budget of 18 computing cores during 30 min characterizes a moderately expensive problem for an expensive evaluation of 15 s on one computing unit. Indeed, the upper bound for the number of expensive evaluations amounts to 2160. In this context, both the convergence profiles and the top-5 rankings presented in Table 6 indicate a clear preference for P-SAGAs. Nevertheless, if the budget is defined as a limited number of around 100 expensive evaluations, which defines a very expensive problem, the P-SDAs are clearly the best performing methods as shown by Figs. 3–6. In most cases, it is preferable to use P-SDAs for less than around 500 expensive evaluations and to use P-SAGAs for more than 500 expensive evaluations. However, this threshold depends on the problem at hand and the algorithm configuration. Indeed, for the Rosenbrock problem this threshold seems to be larger than 500 as shown in Fig. 5. It is also greater than 500 for (CL, GP_RBF, *com-spf*) on the COVID-19 problem in Fig. 6 while the value of 500 seems to be accurate for (CL, GP_RBF_CTS, *ei*) on the same graph.

Among the P-SAGAs, there is no unique AP outperforming all the others in all the cases as demonstrated by Table 6. Using the surrogate as an evaluator may be a good idea if an acceptable prediction accuracy is reached as it seems to be the case for the GP in the Rastrigin problem according to both Tables 2 and 6. For a low prediction accuracy, only using the surrogate as a filter is advised as erroneously discarded candidates may be recovered easily during next generations thanks to the

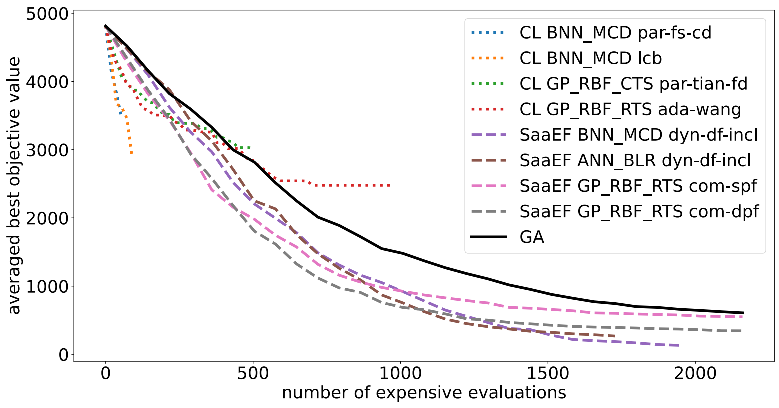

Fig. 3. Grid search on P-SAGA and qEGO design space. Schwefel problem. Convergence profile in terms of best expensive objective values averaged over the 10 runs of the experiment. Dotted lines represent qEGO-like approaches, dashed lines represent P-SAGA approaches and the plain line represents the GA.

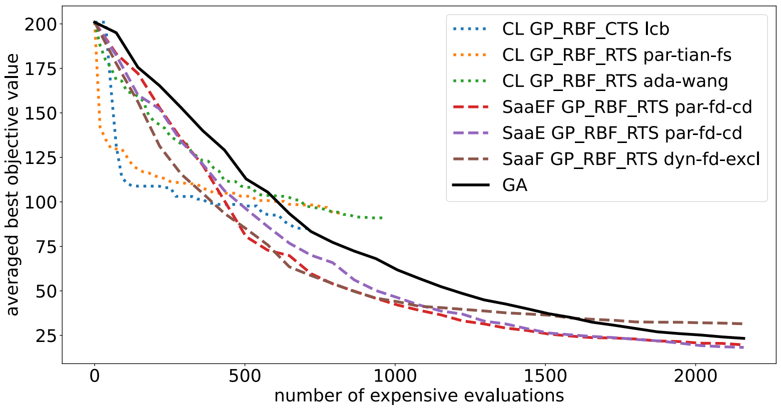

Fig. 4. Grid search on P-SAGA and qEGO design space. Rastrigin problem. Convergence profile in terms of best expensive objective values averaged over the 10 runs of the experiment. Dotted lines represent qEGO-like approaches, dashed lines represent P-SAGA approaches and the plain line represents the GA.

evolutionary operators. Among the qEGO variants, SB is outperformed by CL on the artificial problems. This is the reason why it has not been applied to the COVID-19-related problem.

By comparing the PC, the dynamic and Pareto-based criteria seem to be adequate choices in P-SAGA. In particular, *dyn-df-incl* and *dyn-df-excl* appear recurrently in the top-5 exposed in Table 6. They first promote exploration based on the distance to the archive thus allowing to gain a global view over the search space. As the search proceeds, they enhance exploitation of the most promising regions. In P-SAGAs, maximizing the distance to the archive is a better way of exploring than relying on the standard deviation around the prediction as the results show. Conversely, in qEGO, the standard deviation is more reliable and should be combined with the intensification metric of predicted objective value all along the search as shown by the performance of the scalarized, Pareto-based and adaptive criteria in Figs. 3–6. In P-SDA, it is crucial to both quickly focus on promising regions and refine the surrogate in regions of high uncertainty.

5. Hybrid algorithms for the COVID-19 contact reduction problem

5.1. Motivations

The observations from the experiments reported in Section 4.2.2 indicate the suitability of qEGO for very computationally expensive problems while P-SAGAs are more adequate in a moderately expensive context. From this conclusion, it is proposed to design hybrid algorithms to provide robustness with respect to the computational budget and to improve the optimization quality in the moderately expensive setting. On the one hand, the threshold defining the domain of suitability of P-SAGAs and qEGO – in terms of number of expensive evaluations – is not clearly defined and may depend on the algorithm configuration and the problem being tackled as seen in Figs. 3–6. On the other hand, optimization in moderately expensive settings might be improved by boosting P-SAGAs thanks to a qEGO-like AP. In this section, the hybridization of P-SAGAs and qEGO is investigated for the solving of the COVID-19 contact reduction problem.

On the COVID-19 contact mitigation application, the best method identified in a moderately expensive context is (BNN_MCD, SaaF, *dyn-df-incl*) according to Table 6. In a very computationally expensive setting, (GP_RBF, CL, *com-spf*) is preferred as Fig. 6 shows. In the next subsection, we first suggest to run both APs concurrently at each iteration to benefit from the acquisition performances of both reproduction operators and optimization of a PC. We also propose, in Section 5.3, to execute both APs successively, starting with qEGO and continuing with P-SAGA if the computational budget allows it. Alternative algorithms combinations could be thought of such as integrating one method into another similarly to a memetic algorithm where a local search acts as a mutation operator in a GA [47]. In P-SDA, the internal auxiliary optimization problem could be solved by a P-SAEA. However, this auxiliary problem is not computationally expensive and it is therefore better addressed by a surrogate-free method. In the case of separable problems, different optimizers can be invoked on different sub-sets of decision variables such as in co-evolutionary algorithms [48]. Nevertheless, the COVID-19-related problem is not expected to be separable. Mixing more than two APs is envisioned in the Surrogate Model Based Optimization Evolutionary Algorithm (SMBOEA) [17] that is considered as a competing hybrid approach in the numerical experiments reported in the following.

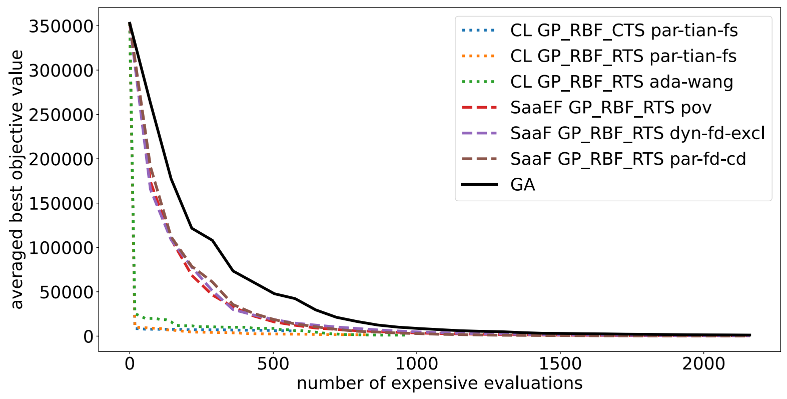

Fig. 5. Grid search on P-SAGA and qEGO design space. Rosenbrock problem. Convergence profile in terms of best expensive objective values averaged over the 10 runs of the experiment. Dotted lines represent qEGO-like approaches, dashed lines represent P-SAGA approaches and the plain line represents the GA.

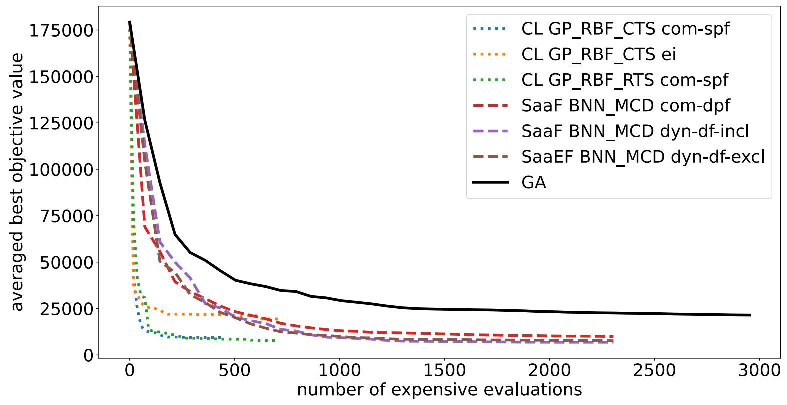

Fig. 6. Grid search on P-SAGA and qEGO design space. COVID-19 contact reduction problem. Convergence profile in terms of best expensive objective values averaged over the 10 runs of the experiment. Dotted lines represent qEGO-like approaches, dashed lines represent P-SAGA approaches and the plain line represents the GA.

5.2. Hybrid concurrent acquisition process

The first new hybrid method is named HCAP for "Hybrid Concurrent Acquisition Process" and is presented in Algorithm 2. The two aforementioned APs, (BNN_MCD, SaaF, *dyn-df-incl*) and (GP_RBF, CL, *com-spf*) and their respective parameters calibrated in the previous section, are executed concurrently at each iteration to propose new candidates. One iteration consists of firstly generating $q_1 = 9$ new promising candidates (line 6 in Algorithm 2) *via* qEGO. Thence, parent solutions are selected from the population and reproduced to create a batch of new solutions (lines 7 and 8) from which $q_2 = 63$ are kept (line 9) according to SaaF. A total of $q_1 + q_2 = 72$ new candidates are evaluated with the expensive objective function in parallel at each iteration (lines 10 and 11). Thereafter, the surrogates are updated (lines 13 and 14) and a new population is formed by elitist replacement (line 15). The value for q_1 is maintained at a small level as qEGO is limited in this respect [29]. The value for q_2 is chosen such that the number of expensive evaluation $q_1 + q_2 = 72$ such as SaaF.

5.3. Hybrid successive acquisition processes

This method runs successively the two APs as described in Algorithm 3. It is referred to as HSAP for "Hybrid Successive Acquisition Processes". The first stage consists of running six iterations of (GP_RBF, CL, *com-spf*) with $q_1 = 18$ expensive evaluations per iterations thus corresponding to a total of 108 expensive evaluations (lines 2 to 11). Afterwards, (BNN_MCD, SaaF, *dyn-df-incl*) is run until the budget is completely consumed (lines 12 to 24). The population is initialized by taking special care of balancing between exploration and exploitation. To foster exploitation, the ten best candidates identified so far are included in the initial population (line 12). To boost exploration, a K-Means algorithm [49,50] partitions the set of decision vectors from the archive into 62 groups and one randomly-selected solution per cluster is added to the initial population (line 13).

5.4. Numerical experiments

In this subsection, we present the experimental protocol set up to analyze the proposed methods in terms of optimization quality and parallel scalability. An *a posteriori* landscape analysis is also performed on the COVID-19-related problem.

5.4.1. Experimental protocol

The computational budget is set to 30 min on $n_{cores} = 18$ computing cores and ten independent runs per algorithm are performed on the COVID-19 contact mitigation problem. The seven competing algorithms are the parallel surrogate-free GA, SaaF (BNN_MCD, SaaF, *dyn-df-incl*), two versions of CL (GP_RBF_CTS, CL, *com-spf*) and (GP_RBF_RTS, CL, *com-spf*), HCAP, HSAP and SMBOEA from [17].

In SMBOEA, an iteration consists in running three APs in parallel. The first AP, executed on one computing core, maximizes the Expected Improvement [28] to produce a new candidate. The second AP, also running on one computing core, minimizes $\hat{f}$ to output one new solution. The third AP generates $q = n_{cores} - 2$ new candidates *via* reproduction of q parents extracted from the current population. The n_{cores} new candidates are evaluated in parallel by the expensive objective function. Afterwards, the archive, the surrogate and the population are updated and the procedure is repeated until the computational budget is wasted.

Table 6
Grid search on P-SAGA and qEGO design space. Top-5 strategies for each framework according to the final expensive objective value averaged over 10 runs. Ordering according to ascending average final expensive objective values from top to bottom.

Surrogate AP PC	Avg(stdev)	Surrogate AP PC	Avg(stdev)
P-SAGA		qEGO	
	Schwefel		
BNN_MCD SaaEF *dyn-df-incl*	**131.96(26.29)**	GP_RBF_CTS CL *dyn-df-excl*	2435.37(279.95)
BNN_MCD SaaF *dyn-df-incl*	153.93(82.76)	GP_RBF_RTS CL *ada-wang*	2478.06(262.74)
BNN_MCD SaaEF *par-fd-cd*	167.55(90.79)	BNN_MCD CL *lcb*	2723.47(317.26)
BNN_MCD SaaEF *dyn-df-excl*	168.64(92.18)	GP_RBF_RTS CL *par-tian-fs*	2730.8(497.62)
BNN_MCD SaaF *dyn-df-excl*	184.87(93.12)	GP_RBF_RTS CL *ei*	2913.49(398.53)
	GA 607.91(267.01)		
	Rastrigin		
GP_RBF_RTS SaaE *par-fd-cd*	**18.22(2.55)**	GP_RBF_CTS CL *lcb*	84.66(18.72)
GP_RBF_RTS SaaE *dyn-df-excl*	18.7(4.51)	GP_RBF_RTS CL *ada-wang*	90.59(20.04)
GP_RBF_RTS SaaF *com-dpf*	19.26(4.2)	GP_RBF_CTS CL *pov*	90.82(10.32)
GP_RBF_RTS SaaF *com-dpf*	19.34(4.98)	GP_RBF_RTS CL *par-tian-fs*	92.92(7.89)
GP_RBF_RTS SaaEF *par-fd-cd*	19.66(6.91)	GP_RBF_CTS CL *ada-wang*	94.64(9.54)
	GA 23.30(5.53)		
	Rosenbrock		
GP_RBF_RTS SaaF *par-fd-cd*	**137.82**(76.89)	GP_RBF_RTS SB *ada-wang*	472.02(197.97)
GP_RBF_RTS SaaF *com-dpf*	156.64(**63.09**)	GP_RBF_RTS CL *ada-wang*	1109.17(588.57)
GP_RBF_RTS SaaF *ei*	232.37(201.10)	GP_RBF_CTS CL *dyn-df-excl*	1146.64(501.8)
GP_RBF_RTS SaaE *par-fd-cd*	249.21(114.15)	GP_RBF_RTS CL *par-tian-fs*	1515.04(589.03)
GP_RBF_RTS SaaF *dyn-df-incl*	263.31(137.81)	GP_RBF_RTS CL *par-tian-fd*	1629.54(711.67)
	GA 1191.14(1248.11)		
	COVID-19		
BNN_MCD SaaF *dyn-df-incl*	**6854**(2067)	GP_RBF_RTS CL *com-spf*	7824(1554)
BNN_MCD SaaF *dyn-df-excl*	7115(1536)	GP_RBF_CTS CL *com-spf*	8897(1834)
BNN_MCD SaaEF *dyn-df-excl*	7679(3373)	GP_RBF_RTS CL *par-fs-cd*	10,117(**1519**)
BNN_MCD SaaEF *dist*	7838(4133)	GP_RBF_CTS CL *par-fs-cd*	10,997(2319)
BNN_MCD SaaF *dist*	7852(3029)	GP_RBF_RTS CL *com-dpf*	11,067(2300)
	GA 21483(7345)		

In SMBOEA, contrary to HCAP and HSAP, no filtering occurs in the AP based on the reproduction operators. Relying on an PC at this step gives more opportunity to the reproduction operators to generate good candidates. The objective pointed out in [17] for future works on SMBOEA is to improve the performance of the method when n_{cores} increases. Indeed, in the experiments reported in [17], SMBOEA performs similarly to the parallel surrogate-free GA for $n_{cores} \geqslant 15$. In HCAP and HSAP, the use of two surrogates from different types aims at enhancing diversification in the batch of new samples and improving the overall performance of the hybrid methods. In SMBOEA, the three APs are performed in parallel while the two APs from HCAP are performed sequentially thus giving a slight advantage to SMBOEA regarding idleness of computing cores.

The only adaptation made to SMBOEA to tackle the COVID-19 contact reduction problem is to replace the Kriging model by GP_RBF_CTS because of the results of the offline comparison of surrogates of Section 4.1. The remaining algorithms are calibrated as presented in Table 4, Algorithm 2 and Algorithm 3. The pySBO platform is used as the software framework for implementation and experimentation [51]. The experiments are conducted on 18 computing cores from an Intel Xeon Gold 5220 CPU. The parallel machine is part of the Grid5000, an infrastructure dedicated to parallel and distributed computing [52].

5.4.2. Analysis of optimization quality

Fig. 7 shows the distribution of the ten best objective values obtained at the end of the search (one value at the end of each repetition) for each strategy. It can be observed that the new hybrid method HSAP significantly outperforms all its competitors on the COVID-19 contact reduction problem in terms of average, median and variance of the results. The concurrent combination of APs proposed by HCAP is also a reliable strategy as it outperforms all the non-hybrid methods and SMBOEA. It can be noticed that SMBOEA demonstrates a similar averaged final objective value than the parallel surrogate-free GA but with a larger variation.

The convergence profiles are displayed in Fig. 8 with a zoom on the best performing methods around 300 expensive evaluations. Expectedly, HSAP and CL exhibit a similar very steep improvement for less than 108 expensive evaluations. After the AP switch in HSAP, the improvement is slowed down but a continuous progress is noted until around 600 expensive evaluations where the convergence is almost reached. The zoomed picture highlights the benefit from using HSAP over CL from 300 expensive evaluations onward. Firstly, HSAP allows one to perform more expensive evaluations than CL as indicates the length of the curves. Secondly, the use of the surrogate to inform the reproduction operators enables a continuous improvement as soon as CL reaches steady state. An attractive enhancement of HSAP would be to automatically detect the flatness in the convergence curve and trigger the AP switch. Such a mechanism is not trivial to design, particularly because user-defined parameters must be avoided. HCAP outperforms SMBOEA and SaaF in Fig. 8 while SaaF overtakes SMBOEA after 260 expensive evaluations. The bad performances of the parallel surrogate-free GA illustrate again the profit brought by surrogate models for both moderately and very expensive problems.

The length of the curves in Fig. 8 yields indications about the computational cost of the methods. Among the hybrid methods, SMBOEA is the more computationally costly as the surrogate is trained on the entire archive and PC optimizations are run at each iteration. By reducing the training set size as in HCAP, more expensive evaluations are enabled and by reducing the computational effort dedicated to PC optimization as in HSAP, the number of expensive evaluations gets closer to the one of SaaF. A possible way to relieve the computational cost of HCAP would be to execute both APs in parallel as it is the case in SMBOEA.

5.4.3. Analysis of parallel scalability

It is now proposed to study the behavior of the seven algorithms for different computational budgets obtained by varying the number of available computing units ($n_{cores} \in \{3, 9, 18, 72, 144\}$) while the allocated time is still fixed to 30 min.

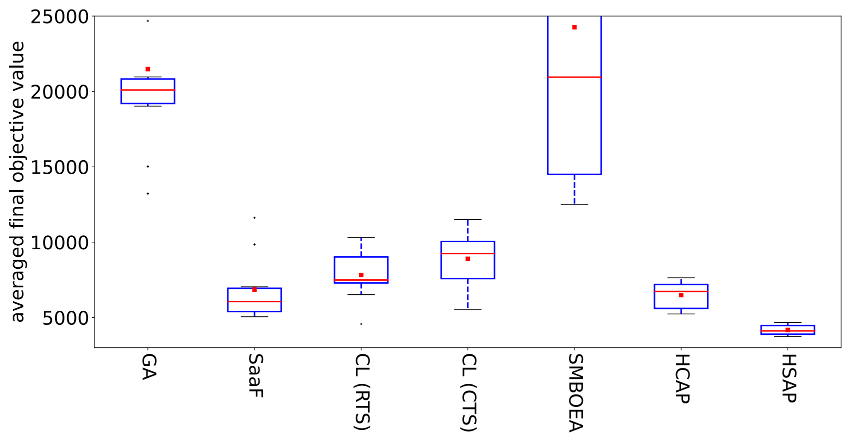

Fig. 7. Hybridization for the COVID-19 contact reduction problem. Analysis of optimization quality. Distribution of the best final expensive objective values from the 10 runs of the experiment. Averaged values are depicted by red squares.

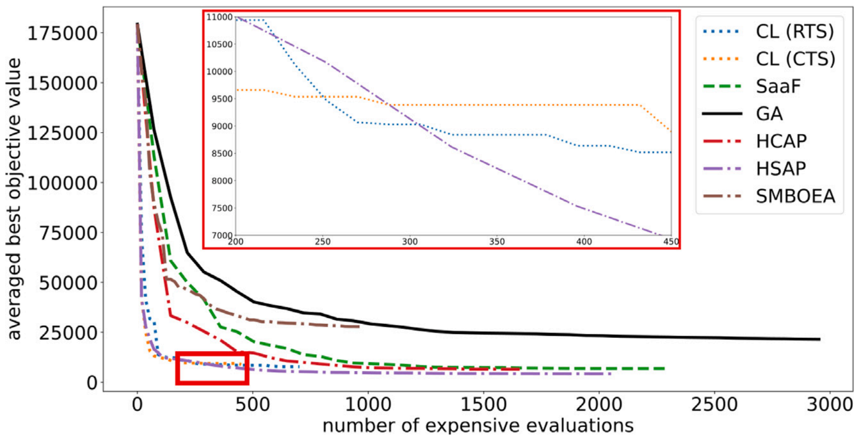

Fig. 8. Hybridization for the COVID-19 contact reduction problem. Analysis of optimization quality. Convergence profile in terms of best expensive objective values averaged over the 10 repetitions of the experiment. Dotted lines represent qEGO-like approaches, the dashed line represent the P-SAGA approach, the plain line represents the GA and the dash-dotted lines represent the hybrid methods.

For the CL-like APs, the number of expensive evaluations per iteration is set to the number of available computing cores $q = n_{cores}$. The challenge of acquiring multiple and diverse promising new candidates arises when the number of computing cores is large. When n_{cores} is small, more computational efforts are engaged into full surrogate trainings thus limiting the affordable number of expensive evaluations, but bringing the advantage of a more accurate surrogate.

For APs relying on reproduction operators, $q = n_{cores}$ is also applied. In SaaF, the population size $n_{pop} = 72$ and the number of children $n_{chld} = 288$ are kept unaltered and the number of discardings per iteration is set to $n_{disc} = 288 - n_{cores}$. In the parallel surrogate-free GA, the number of children is set to the number of computing cores $n_{chld} = n_{cores}$ while the population size is not modified compared to Table 4.

In HCAP, to preserve $q_1 < q_2$ and to guarantee $n_{cores} = q = q_1+q_2$, the following triplets are considered: $(n_{cores}, q_1, q_2) = (3, 1, 2), (9, 1, 8), (18, 2, 16), (72, 9, 63), (144, 18, 126)$.

Table 7 presents the average best expensive objective value found by the algorithms on the COVID-19 problem. The overall best average solution for this problem is provided by HSAP with $n_{cores} = 144$.

For $n_{cores} \geqslant 9$, HSAP is the best choice among the competing approaches while (GP_RBF_RTS, CL, *com-spf*) provides the best average for $n_{cores} = 3$. These observations are recovered when analyzing the box-plots of the final expensive objective values per n_{cores} value exposed in Fig. 9.

In both Table 7 and Fig. 9, we observe very bad results for the CL methods for $n_{cores} = 144$. During one iteration, the surrogate model is repeatedly updated with more and more hallucinated samples (evaluations of the solutions replaced by the mean of the objective values in the archive). The acquisition process is likely misguided by this hallucinated surrogate model and both acquisitions and expensive evaluations are misspent.

The box-plots of the final expensive values per algorithm are displayed in Fig. 10. The outstanding parallel scalability of HSAP and HCAP is reflected by the enhancement of the quality of the resolutions when n_{cores} increases. The parallel surrogate-free GA, SaaF and SMBOEA also demonstrate a satisfying parallel scalability but with less important improvement when n_{cores} increases compared to HSAP and HCAP. The difficulty of CL to benefit from large number of computing units is highlighted in Fig. 10 where the optimization quality decreases from $n_{cores} = 72$ to $n_{cores} = 144$. Adding too much new solutions per iteration leads to propose unpromising candidates which wastes the budget by investing too much in surrogate training as one surrogate training is realized to obtain one new candidate.

Table 8 presents the average number of expensive evaluations per search. The number of affordable expensive evaluations increases when the number of computing cores increases for all the approaches except (GP_RBF_CTS, CL, *com-spf*). Because the surrogate is trained on the complete archive, the last AP of CL with GP_RBF_CTS is not totally accomplished when $n_{cores} = 144$, therefore explaining the decrease in expensive evaluations compared to $n_{cores} \in \{9, 18, 72\}$. The computational expensiveness of the AP in CL is directly related to q so these strategies present the lowest increase as indicated by Table 8.

Table 7
Hybridization for the COVID-19 contact reduction problem. Analysis of parallel scalability. Best expensive objective values averaged over the 10 runs of the experiment for different numbers of available computing cores.

Method \ n_{cores}	144	72	18	9	3
HSAP	**3,791**	3,865	4,617	7,118	8,950
HCAP	5,487	6,100	7,767	11,275	17,658
SaaF	6,463	7,728	6,779	9,951	21,641
CL (RTS)	13,894	9,443	8,416	8,234	7,624
CL (CTS)	12,820	8,981	8,822	9,579	9,954
GA	7,766	9,604	23,875	25,153	29,355
SMBOEA	18,011	17,892	22,865	29,479	31,613

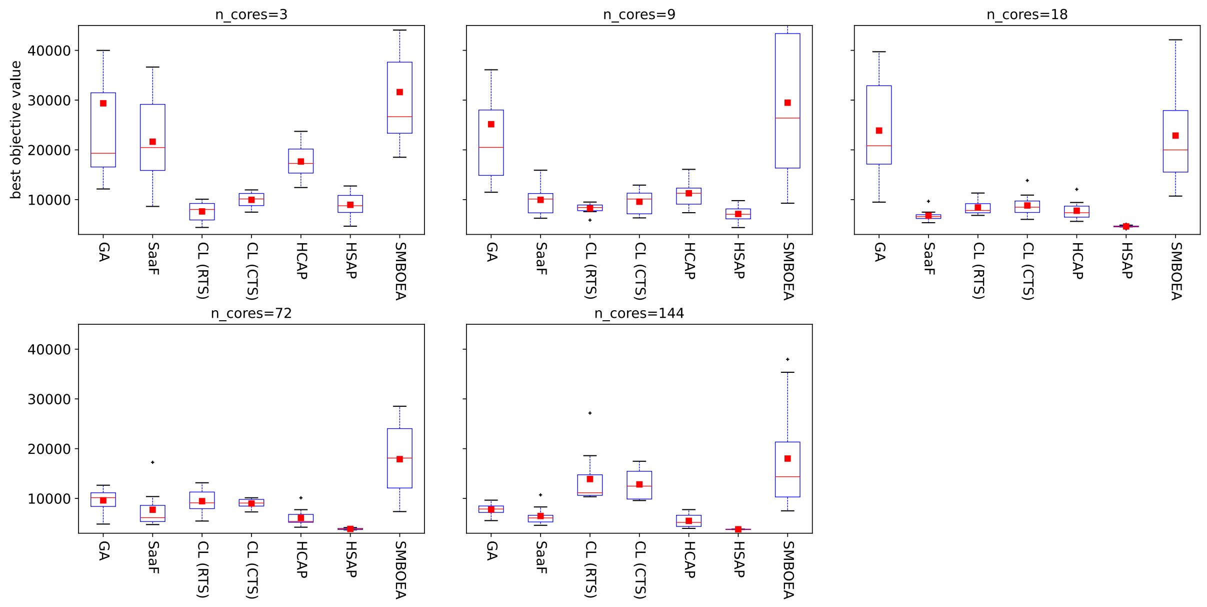

Fig. 9. Hybridization for the COVID-19 contact reduction problem. Analysis of parallel scalability. Distribution of the best objective values from the 10 repetitions of the experiment. Averaged values are depicted by red squares. The scale of the y-axis is homogenized among the subplots. Two outliers do not appear due to the choice in y-axis scale: around 85000 for GA at $n_{cores} = 3$ and around 65000 for GA at $n_{cores} = 9$.

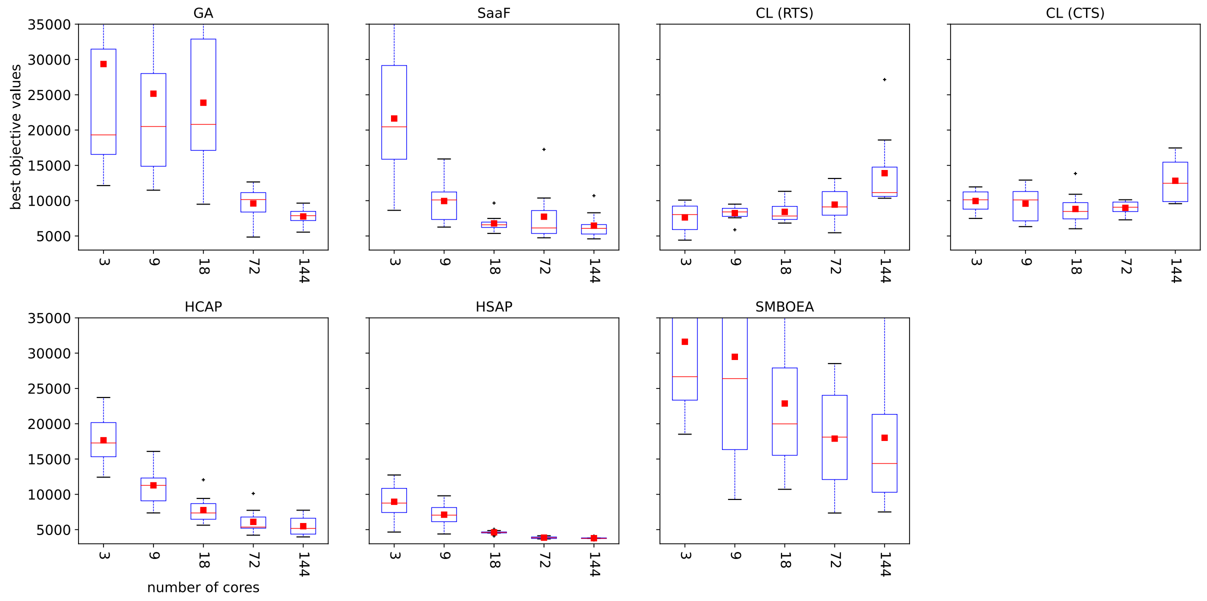

Fig. 10. Hybridization for the COVID-19 contact reduction problem. Analysis of parallel scalability. Distribution of the best objective values from the 10 repetitions of the experiment. Averaged values are depicted by red squares. The scale of the y-axis is homogenized among the subplots. Three outliers do not appear due to the choice in y-axis scale: around 85000 for GA at $n_{cores} = 3$, around 65000 for GA at $n_{cores} = 9$ and around 62000 for SMBOEA at $n_{cores} = 3$.

Table 8
Hybridization for the COVID-19 contact reduction problem. Analysis of parallel scalability. Average number of expensive evaluations per search over the 10 runs of the experiment for different numbers of available computing cores.

Method \ n_{cores}	144	72	18	9	3
GA	22,377	11,347	2,953	1,739	751
SaaF	8,654	5,522	1,818	1,134	483
SaaEF	8,481	4,975	1,792	1,121	488
HSAP	7,819	2,628	1,260	720	354
SMBOEA	3,960	3,355	1,324	1,079	650
HCAP	3,657	2,894	1,407	976	426
CL (RTS)	792	770	673	639	491
CL (CTS)	504	576	527	524	450

Algorithm 2 Framework of HCAP.

```
Input
    $f$: expensive objective function
    $budget$: computational budget for the search
    GP_RBF: surrogate model for AP1
    $com\text{-}spf$: promisingness criterion for AP1
    $q_1 = 9$: number of expensive evaluations per iteration for AP1
    $n_{pop1} = 50$: population size for AP1
    $n_{gen} = 100$: number of generations for AP1
    BNN_MCD: surrogate model for AP2
    $dyn\text{-}df\text{-}incl$: promisingness criterion for AP2
    $n_{pop2} = 72$: population size for AP2
    $n_{chld} = 288$: number of new candidates per iteration for AP2
    $q_2 = 63$: number of expensive evaluations per iteration for AP2
    $n_{disc} = 225$: number of discarding per iteration for AP2
1: $archive \leftarrow$ initial_sampling+expensive_evaluations($f$, $n_{pop2}$)
2: GP_RBF $\leftarrow$ training($archive$)
3: BNN_MCD $\leftarrow$ training($archive$)
4: $\mathcal{P} \leftarrow archive$                          ▷ initial population
5: while $budget \neq 0$ do
6:     $\mathcal{B}_1 \leftarrow$ Constant_Liar_AP($archive$, $com\text{-}spf$, GP_RBF, $q_1$, $n_{pop1}$, $n_{gen}$)
7:     $\mathcal{P}_p \leftarrow$ selection($\mathcal{P}$, $n_{chld}$)
8:     $\mathcal{P}_c \leftarrow$ reproduction($\mathcal{P}_p$, $n_{chld}$)
9:     $\mathcal{B}_2 \leftarrow$ filtering($\mathcal{P}_c$, $dyn\text{-}df\text{-}incl$, BNN_MCD, $q_2$, $n_{disc}$)
10:    $\mathcal{B} \leftarrow \mathcal{B}_1 \cup \mathcal{B}_2$
11:    expensive_evaluations($f$, $\mathcal{B}$)
12:    $archive \leftarrow archive \cup \mathcal{B}$
13:    GP_RBF $\leftarrow$ training($archive$, 72)
14:    BNN_MCD $\leftarrow$ training($archive$, all)
15:    $\mathcal{P} \leftarrow$ elitist_replacement($\mathcal{P}$, $\mathcal{B}$, $n_{pop2}$)
16:    $budget \leftarrow$ get_remaining_budget($budget$, elapsed_time)
17: end while
18: $(\boldsymbol{x}_{min}, y_{min}) \leftarrow$ get_best_cost($archive$)
19: return $\boldsymbol{x}_{min}$, $y_{min}$
```

For $n_{cores} = 3$, SMBOEA enables the highest number of expensive evaluations (650) among the surrogate-based approaches because both the generation and the evaluation of the candidates are performed in parallel. This is not true anymore when n_{cores} grows as the production of the $n_{cores} - 2$ new decision vectors by reproduction is executed by a unique computing core. The number of expensive evaluations is high for SaaF and HSAP when $n_{cores} = 144$ as the participation of the AP based on PC optimization is null or restricted.

In the context of a very expensive problem – where the budget is only limited by few hundred expensive evaluations – maximizing the benefits of a large number of computing cores is challenging as depicted by Fig. 11. For an AP built on reproduction operators, running multiple generations appears more adequate. For an AP set up on PC optimization, the difficulty to generate large new batches of promising solutions

Algorithm 3 Framework of HSAP.

```
Input
    $f$: expensive objective function
    $budget$: computational budget for the search
    GP_RBF: surrogate model for AP1
    $com\text{-}spf$: promisingness criterion for AP1
    $q_1 = 18$: number of expensive evaluations per iteration for AP1
    $n_{pop1} = 50$: population size for AP1
    $n_{gen} = 100$: number of generations for AP1
    BNN_MCD: surrogate model for AP2
    $dyn\text{-}df\text{-}incl$: promisingness criterion for AP2
    $n_{pop2} = 72$: population size for AP2
    $n_{chld} = 288$: number of new candidates per iterations for AP2
    $q_2 = 72$: number of expensive evaluations per iteration for AP2
    $n_{disc} = 216$: number of discarding per iteration for AP2
1: $archive \leftarrow$ initial_sampling+expensive_evaluations($f$, $n_{pop2}$)
2: GP_RBF $\leftarrow$ training($archive$, all)
3: $counter \leftarrow 0$
4: while $counter < 6$ AND $budget$ available do
5:     $\mathcal{B} \leftarrow$ Constant_Liar_AP($archive$, $com\text{-}spf$, GP_RBF, $q_1$, $n_{pop1}$, $n_{gen}$)
6:     expensive_evaluations($f$, $\mathcal{B}$)
7:     $archive \leftarrow archive \cup \mathcal{B}$
8:     GP_RBF $\leftarrow$ training($archive$)
9:     $budget \leftarrow$ get_remaining_budget($budget$, elapsed_time)
10:    $counter \leftarrow counter + 1$
11: end while
12: $\mathcal{P} \leftarrow$ get_best($archive$, 10)                 ▷ initial population
13: $\mathcal{P} \leftarrow \mathcal{P} \cup$ K-Means_sampling($archive$, 62)
14: BNN_MCD $\leftarrow$ training($archive$)
15: while $budget$ available do
16:    $\mathcal{P}_p \leftarrow$ selection($\mathcal{P}$, $n_{chld}$)
17:    $\mathcal{P}_c \leftarrow$ reproduction($\mathcal{P}_p$, $n_{chld}$)
18:    $\mathcal{B} \leftarrow$ filtering($\mathcal{P}_c$, $dyn\text{-}df\text{-}incl$, BNN_MCD, $q_2$, $n_{disc}$)
19:    expensive_evaluations($f$, $\mathcal{B}$)
20:    $archive \leftarrow archive \cup \mathcal{B}$
21:    BNN_MCD $\leftarrow$ training($archive$, all)
22:    $\mathcal{P} \leftarrow$ elitist_replacement($\mathcal{P}$, $\mathcal{B}$, $n_{pop2}$)
23:    $budget \leftarrow$ get_remaining_budget($budget$, elapsed_time)
24: end while
25: $(\boldsymbol{x}_{min}, y_{min}) \leftarrow$ get_best($archive$, 1)
26: return $\boldsymbol{x}_{min}$, $y_{min}$
```

and the associated reduction of number of iterations imply a waste of the computational budget. In this context, it is more convenient to employ qEGO or HSAP with a curbed number of computing cores.

5.4.4. Landscape analysis

Landscape analysis is an entire research area that aims at characterizing the shape of the graph $(\Omega, f(\Omega))$ produced by the objective

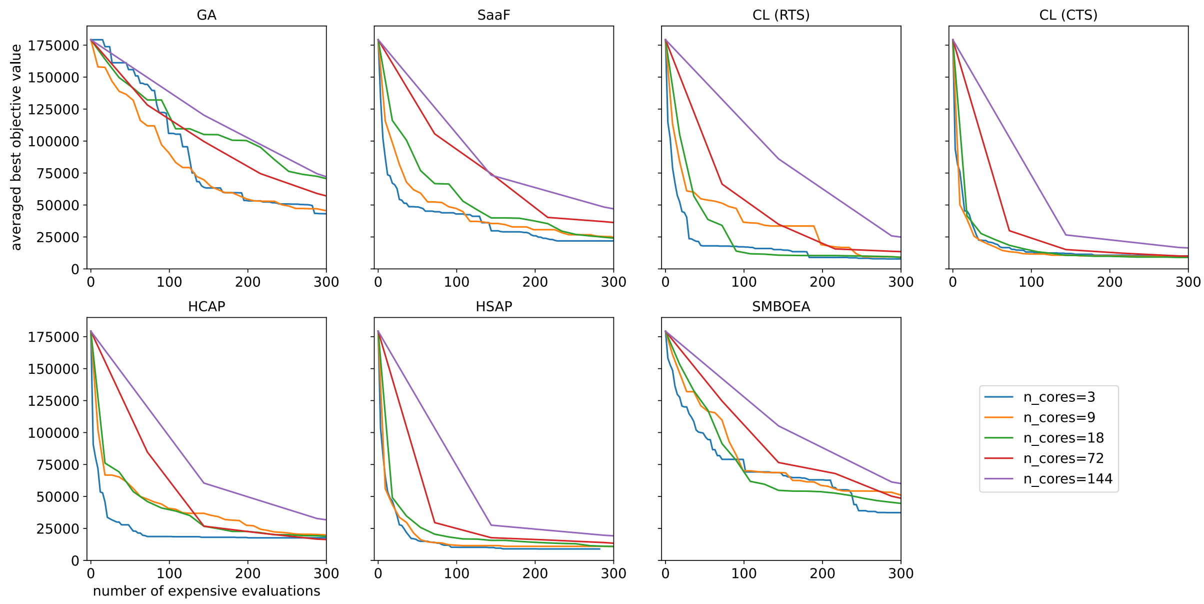

Fig. 11. Hybridization for the COVID-19 contact reduction problem. Analysis of parallel scalability. Convergence profile in terms of best expensive objective values averaged over the 10 runs of the experiment for different numbers of available computing cores.

Table 9
Hybridization for the COVID-19 contact reduction problem. Landscape analysis. Best decision vector to the COVID-19 contact reduction problem. x_i represents the contact mitigation factor for age-group i.

Age-group i	0–5	5–10	10–15	15–20	20–25	25–30	30–35	35-40
x_i	0.96	0.97	1.00	0.97	1.00	1.00	1.00	1.00
Age-group	40–45	45–50	50–55	55–60	60–65	65–70	70–75	75+
x	0.97	0.76	0.00	0.00	0.00	0.00	0.00	0.00

function f [53]. Indications such as the abundance of basins of attraction, the extent of the infeasible region, the presence of flat regions or ridges are valuable to the designers of optimization algorithms as well as to the experts of the targeted problem. In this sub-subsection, after revealing the best contact reduction strategy found so far, a landscape analysis is performed on the COVID-19 problem by analogy with the artificial functions, Schwefel, Rastrigin, Rosenbrock, whose landscape features are known.

The overall minimum objective value found so far amounts to 3,536 and has been produced by the new HSAP method with $n_{cores} = 144$. According to the associated decision vector exposed in Table 9, contact for people aged less than 50 years-old should only be slightly reduced. Conversely, drastic contact reductions should be applied to seniors of 50+ years-old. Indeed, almost no restriction (less than 5%) is suggested for people from 0 to 45 years-old. Extreme contact restrictions are suggested for the elders beyond 50 years-old surely as they present the highest risk of medical complications.

An *a posteriori* analysis of the COVID-19 contact reduction problem is now possible as the experiments conducted so far have produced a huge amount of evaluations of f. Among the 12,463,182 solutions evaluated by f, 4,452,189 are infeasible, representing 36% of the whole set. The extent of the infeasible search space is expected to be greater than 36% because the sampling is biased towards the feasible region. Indeed, from the 720 initial solutions obtained *via* Latin Hypercule Sampling, only one is feasible. This is actually the reason why landscape analysis was not performed prior to the optimization.

Relying on surrogate models to characterize a search landscape is a technique that has already been proposed in [54,55]. The empirical comparisons led in Section 4 suggested an analogy between the landscape of the Schwefel problem and the one of the COVID-19 application. Another tool for landscape analysis, namely the dispersion metric, is leveraged to gain more knowledge about the landscape of the COVID-19 problem. The dispersion metric [56] detects the multi-modality and the presence of global structure by measuring the average distance between the best solutions in the search space. The set of best candidates is defined by a proportion p_{DM} of the best solutions from the data set. It is stated in [55], that a database made of $50.d$ samples is sufficient to perform exploratory landscape analysis. Since $d = 16$ in this study, 800 samples are drawn for each problem. For Schwefel, 11,169,468 expensive evaluations are available, 11,161,044 ones for Rastrigin, 11,276,316 ones for Rosenbrock and 8,010,993 feasible solutions are accessible for the COVID-19 contact reduction problem. Each of these sets is divided into 800 clusters using the K-Means algorithm [50] implemented in Scikit-Learn [57] and the closest solution to each cluster's center is retained. The Flacco R package [55] is used to compute the dispersion metric for multiple values of p_{DM} and the outcomes are reported in Table 10.

Large values of the dispersion metric indicate high dispersion of the best solutions in the search space and consequently imply the presence of multiple basins of attraction. For small values of p_{DM}, high values of the dispersion metric indicate a weak global structure in the sense that the multiple basins of attraction are far from each other. According to Table 10, the landscape associated to the COVID-19 problem is similar to the one of the Schwefel problem in terms of multi-modality and global structure.

Other tools built on the concept of nearest neighbors have been elaborated in [58] to bring out weak global structures. Let us denote $\mathcal{S}$ the 800-samples set generated previously for a given problem and let us define the distance to the nearest neighbor by:

$$d_{nn}(\boldsymbol{x}, \mathcal{S}) = min(\{d_2(\boldsymbol{x}, \boldsymbol{y}) | \boldsymbol{y} \in \mathcal{S} \backslash \{\boldsymbol{x}\}\}) \tag{14}$$

where $d_2(.,.)$ is the Euclidean distance. Let us define the distance to the better nearest neighbors by:

$$d_{nb}(\boldsymbol{x}, \mathcal{S}) = min(\{d_2(\boldsymbol{x}, \boldsymbol{y}) | f(\boldsymbol{y}) < f(\boldsymbol{x}) \text{ and } \boldsymbol{y} \in \mathcal{S}\}) \tag{15}$$

Table 10
Hybridization for the COVID-19 contact reduction problem. Landscape analysis. Dispersion metric based on a subset of 800 samples for the benchmark and the COVID-19 contact reduction problem. The dispersion metric is computed as the average distance between the best $\lfloor 800.p_{DM} \rfloor$ solutions divided by the average distance between the 800 solutions. Higher values characterize a harder optimization problem with respect to multi-modality and global structure.

p_{DM}	Schwefel	COVID-19	Rastrigin	Rosenbrock
0.02	0.7253051	0.6087193	0.3987407	0.2755931
0.05	0.7811394	0.7201954	0.4048308	0.3098171
0.1	0.8310808	0.7868463	0.4437553	0.3624034
0.25	0.8872491	0.8637474	0.5376603	0.4896596

Table 11
Hybridization for the COVID-19 contact reduction problem. Landscape analysis. Nearest neighbors-related metrics, as defined in Eq. (18), based on a subset of 800 samples for the artificial and the COVID-19 contact reduction problems. Values closer to 1 indicate a more adequate global structure.

	COVID-19	Schwefel	Rastrigin	Rosenbrock
nbf1	0.863	0.967	1.000	0.998
nbf2	0.884	0.926	0.986	0.991

The set of the nearest neighbors distances is given by:

$$\mathcal{D}_{nn} = \{d_{nn}(\boldsymbol{x}, \mathcal{S}) | \boldsymbol{x} \in \mathcal{S}\} \tag{16}$$

and the set of the better nearest neighbors distances is given by:

$$\mathcal{D}_{nb} = \{d_{nb}(\boldsymbol{x}, \mathcal{S}) | \boldsymbol{x} \in \mathcal{S}\} \tag{17}$$

The two metrics used to compare the landscapes are:

$$nbf1 = \frac{sd(\mathcal{D}_{nn})}{sd(\mathcal{D}_{nb})} \qquad nbf2 = \frac{mean(\mathcal{D}_{nn})}{mean(\mathcal{D}_{nb})} \tag{18}$$

The first metric $nbf1$ is the ratio of the standard deviation of the two distance sets. For highly multi-modal problems or problems with a weak global structure, $sd(\mathcal{D}_{nb})$ is expected to be high so $nbf1 < 1$ while for problems with adequate global structure $sd(\mathcal{D}_{nn}) \approx sd(\mathcal{D}_{nb})$ is expected such that $ndf1 \approx 1$. The same reasoning applies for the second metric $nbf2$ when considering the ratio of the mean of the sets. Table 11 presents the nearest neighbors-related metrics computed for the artificial and the real-world problems. According to Table 11, the COVID-19 problem exhibits the less adequate topology followed by the Schwefel problem. The Rastrigin adequate global structure is detected by showing $nbf1 = 1$ for the associated samples set.

By the *a posteriori* landscape analysis conducted in this subsection, it can be deduced that the constraint is severe and the landscape is multi-modal with a weak global structure. Adding the fact that the simulation is moderately expensive, the COVID-19 contact reduction problem is undoubtedly tedious to solve. In such a critic case, the design and application of hybrid methods is thus relevant as they yield the best resolution of the problem.

6. Hybridization and robustness to search landscapes

In the previous section, the focus was on the design of hybrid acquisition processes to solve the COVID-19 contact reduction problem. In this section, we study the robustness of the algorithms with respect to the search landscape. To this end, the CEC2015 benchmark test suite is invoked.

6.1. Experiments on the CEC2015 test suite

The algorithms studied in the previous section are now compared on the broader set of problems defined by the CEC2015 test suite. A total of 30 objective functions are addressed for a computational budget of 30 min on 18 computing cores while assuming an expensive evaluation lasting 15 s on one core.

The six algorithms employed in the COVID-19-related problem, CL (with GP_RBF_CTS), SaaF, GA, HCAP, HSAP and SMBOEA are included in this experiment. Because some of these algorithms are specifically calibrated to the COVID-19 problem (*e.g.* the AP switch parameter of HSAP), three variants, CL-lcb, HCAP-par and HSAP-lcb are added to the pool of competing algorithms in an attempt to provide more generalizable performances. In CL-lcb, Lower Confidence Bound is selected as PC. In HCAP-par, the Pareto-based PC *par-tian-fs* is employed in the AP based on PC optimization. Indeed, both *lcb* and *par-tian-fs* provide good performances across more different problems than *com-spf* as shown in Section 4.2. In HSAP-lcb, *lcb* is also used as PC in the CL-like AP and the switch from one AP to the other is automatically triggered if no improvement of at least 2% is observed during three consecutive iterations.

The top-2 algorithms according to the average best expensive objective value discovered after 108 expensive evaluations and at the end of the budget are reported in Table 12. The new hybrid strategies HCAP, HCAP-par, HSAP and HSAP-lcb do not generalize very well. For some problems, such as the 30-D CEC2015-1 problem, HSAP-lcb behaves as expected by automatically triggering the switch from one AP to another and further improving the results as shown by the convergence plot of Fig. 12. However, for other problems such as the 10-D CEC2015-15 instance, the extreme roughness of the landscape cuts drastically the prediction accuracy of the surrogate model making GA and SMBOEA the best performing methods as exhibited in Fig. 13.

The dispersion metric presented in the previous section is leveraged to try explaining the different performances. The second column of Table 12 exposes the average differences between the dispersion metric at 25% and the dispersion metric at 5%, denoted d_{disp}, computed on the 20 sets of solutions used to initialize the algorithms. According to [56], a low value of d_{disp} reflects a more difficult landscape to optimize as the multiple basins of attraction are far from each other, therefore suggesting a weak global structure. The algorithms primarily relying on reproduction operators without surrogate model, typically GA and SMBOEA, demonstrate the best performances at the end of the search and for landscape with small d_{disp} according to Table 12. This is especially the case for the 30-D problems 2–5 and 9–12. On landscapes with stronger global structures, HSAP-lcb and SaaF are much performing by the end of the search, specifically on the 10-D problems 6–8 and 13–14 and on the 30-D problems 1 and 6–8. When considering smaller budgets, CL-lcb, CL, HSAP and HSAP-lcb are good options as they recurrently appear in the top-2 after 108 expensive evaluations in Table 12.

By the results, no significant difference of performance is identified between CL and CL-lcb. Table F.28 in Appendix F presents the number of problems for which one method is better than the other according to the average best expensive objective value computed over the 20 runs. The same observation applies for HSAP and HSAP-lcb (Table F.29) and for HCAP and HCAP-par (Table F.30).

Based on these new evidences, we propose a new hybrid method relying on the dispersion metric to drive the choice of AP at the beginning of the search.

Table 12
Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. Top-2 algorithms according to the average best expensive objective value over the 20 repetitions of the experiment. Ordering per dimension according to ascending average difference between the dispersion metrics at 25% and at 5% (denoted d_{disp}) from top to bottom. Smaller values of the average d_{disp} (darker backgrounds) suggest a search landscape with a weak global structure while larger values (lighter background) indicate a stronger global structure.

Problem index	Average d_{disp}	Top-2 end of search		Top-2 108 evaluations	
CEC2015 10-D					
15	-0.011	GA	SMBOEA	GA	SMBOEA
4	0.006	SMBOEA	GA	CL-lcb	HCAP
9	0.01	GA	SMBOEA	HCAP-par	HCAP
3	0.019	CL-lcb	GA	CL-lcb	SMBOEA
2	0.025	SMBOEA	SaaF	HSAP	CL-lcb
11	0.03	SMBOEA	GA	HSAP	CL
10	0.031	SaaF	GA	CL	HSAP
5	0.036	GA	SMBOEA	HSAP	SaaF
12	0.062	GA	SaaF	SMBOEA	SaaF
14	0.065	HSAP	HCAP	HSAP	CL
1	0.092	CL-lcb	GA	CL-lcb	HSAP-lcb
13	0.093	SaaF	HSAP-lcb	SMBOEA	HCAP
7	0.103	SaaF	HSAP-lcb	CL-lcb	HSAP-lcb
6	0.111	GA	SaaF	CL-lcb	HSAP-lcb
8	0.116	SaaF	GA	SMBOEA	HCAP
CEC2015 30-D					
2	-0.006	GA	SaaF	CL-lcb	SMBOEA
5	-0.005	GA	SMBOEA	SMBOEA	SaaF
3	0.002	GA	SMBOEA	CL-lcb	HSAP
4	0.008	GA	SMBOEA	SaaF	CL-lcb
10	0.008	GA	SMBOEA	CL	HSAP
9	0.01	GA	SMBOEA	SaaF	HCAP
14	0.013	HSAP	HCAP-par	CL	HSAP
12	0.023	GA	SMBOEA	CL	HSAP
11	0.029	GA	SMBOEA	CL	CL-lcb
13	0.033	SaaF	GA	SMBOEA	SaaF
8	0.034	SaaF	HCAP	HSAP	CL
15	0.034	GA	HSAP-lcb	CL	HSAP
6	0.039	HSAP-lcb	SaaF	CL-lcb	HSAP-lcb
1	0.051	HSAP-lcb	SaaF	HSAP-lcb	CL-lcb
7	0.06	HSAP-lcb	SaaF	HSAP-lcb	CL-lcb

6.2. Towards a Dispersion-Driven Hybrid Acquisition Process

The new hybrid approach is called Dispersion-Driven Hybrid Acquisition Process (DDHAP) and is presented in Algorithm 4. The algorithm starts by computing the difference d_{disp} between the dispersion metric at 25% and the dispersion metric at 5% computed on the initial archive of already evaluated solutions (line 3 in Algorithm 4). If d_{disp} is smaller than a specified threshold indicating a weak global structure, optimization begins with SMBOEA for a quarter of the affordable budget (lines 4–8). If d_{disp} is larger than the threshold, SMBOEA is not run. Then, CL is run (lines 10–17) with a stopping mechanism based on the lack of improvement during 3 iterations (lines 14–16). Finally, if the CL stopping mechanism is triggered and some budget is still available then SaaF is executed (lines 22–26). The surrogate in use in SaaF is either a GP if a strong global structure was detected or a BNN_MCD if a weak global structure was detected (lines 18–21).

The new DDHAP method is compared experimentally to the previously considered approaches, CL-lcb, SaaF, GA, HCAP-par, HSAP-lcb and SMBOEA. Moreover, a multi-start version of the qEI (MS-qEI) is added to the pool of competing algorithms. The qEI algorithm is implemented *via* the BoTorch library [59]. The qEI acquisition function [29] is sampled thanks to the reparametrization trick coupled with Monte-Carlo sampling [60,61]. The restart mechanism consists to run a new Latin Hypercube Sampling to reset the archive of evaluated solutions when a lack of improvement is detected. The restart is triggered when no improvement of at least 0.1% is observed during 5 consecutive iterations. These values are fixed after few trials on the CEC2015 test suite.

The 30 problems from the CEC2015 test suite are tackled for a computational budget of 30 min on 18 computing cores and an expensive evaluation of 15 s on one core. Twenty repetitions of the experiment are realized.

Algorithm 4 Framework of DDHAP.

```
Input
    $th$: threshold for dispersion-based choice of AP
    $\mathcal{A}$: initial archive of expensive evaluations
    $budget$: computational budget allocated to the search
 1: $available\_budget \leftarrow budget$
 2: $\boldsymbol{x}_{min}, y_{min} \leftarrow$ update_best($\mathcal{A}$)
 3: $d_{disp} \leftarrow$ compute_difference_dispersion($\mathcal{A}$)
 4: while $d_{disp} < th$ AND $available\_budget < 0.25\ budget$ do
 5:     $\mathcal{A} \leftarrow$ SMBOEA_iteration($\mathcal{A}$, GP, $pov$, $ei$)
 6:     $\boldsymbol{x}_{min}, y_{min} \leftarrow$ update_best($\mathcal{A}$)
 7:     $available\_budget \leftarrow$ update_budget()
 8: end while
 9: $switch \leftarrow$ False
10: while switch==False and $available\_budget > 0$ do
11:     $\mathcal{A} \leftarrow$ CL_iteration($\mathcal{A}$, GP, $lcb$)
12:     $\boldsymbol{x}_{min}, y_{min} \leftarrow$ update_best($\mathcal{A}$)
13:     $available\_budget \leftarrow$ update_budget()
14:     if no improvement of at least 2% during 3 iterations then
15:         $switch \leftarrow$ True
16:     end if
17: end while
18: $surrogate \leftarrow$ GP
19: if $d_{disp} < th$ then
20:     $surrogate \leftarrow$ BNN_MCD
21: end if
22: while $available\_budget > 0$ do
23:     $\mathcal{A} \leftarrow$ SaaF_iteration($\mathcal{A}$, $surrogate$, $par\text{-}fd\text{-}cd$)
24:     $\boldsymbol{x}_{min}, y_{min} \leftarrow$ update_best($\mathcal{A}$)
25:     $available\_budget \leftarrow$ update_budget()
26: end while
27: return $\boldsymbol{x}_{min}, y_{min}$
```

Table 13 shows the average ranks of the algorithms over the CEC2015 problems for different budgets expressed as a limited time on 18 computing cores, thus taking into account the surrogate training which reflects more realistically the context of moderately expensive problems. According to this table, DDHAP provides consistent results across the monitored budgets and consequently succeeds in providing more robustness with respect to the computational budget than GA and CL. Similar results are obtained with SMBOEA which yields slightly worst ranks than DDHAP in the 30-D case.

Table 14 exposes the average ranks of the methods over the CEC2015 problems for different budgets expressed as a limited number of expensive evaluations reflecting the case of a very expensive problem. While DDHAP shows consistent ranks, MS-qEI outperforms other strategies for a budget of less than 360 expensive evaluations. It can be observed that DDHAP performs more poorly for very tight budgets. This is certainly due to the wrong detection of the global structure at the beginning of the search because the dispersion metric is a rough indicator rather than a metric that completely describes the landscape. Moreover, the size of initial archive of expensive evaluations is restricted and only represents sparsely the search landscape. It may be interesting to study the applicability of other landscape analysis tools [54] to improve the decision made at this step. Replacing CL by MS-qEI within the DDHAP should also improve the new hybrid strategy as MS-qEI demonstrates better ranks than CL in both Tables 13 and 14. Analogically, employing a surrogate-free AP for larger budgets in DDHAP may provide performance enhancement as GA produces the best results for a budget of 30 min in Table 13.

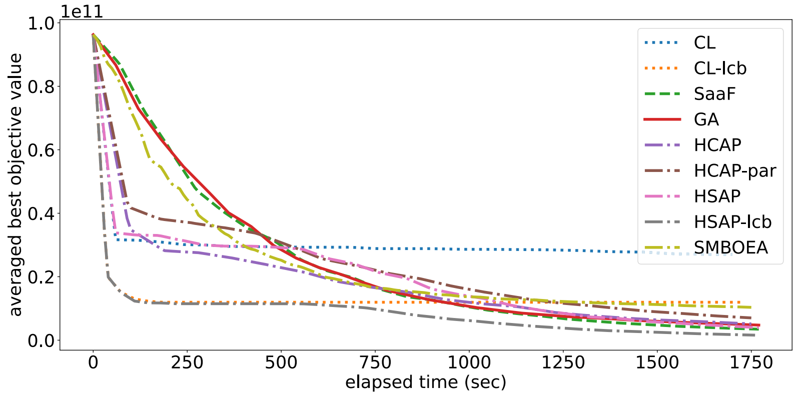

Fig. 12. Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. 30-D CEC2015-1 problem. Convergence profile in terms of best objective values averaged over the 20 runs of the experiment. Dotted lines represent qEGO-like approaches, the dashed line represent the P-SAGA approach, the plain line represents the GA and the dash-dotted lines represent the hybrid methods.

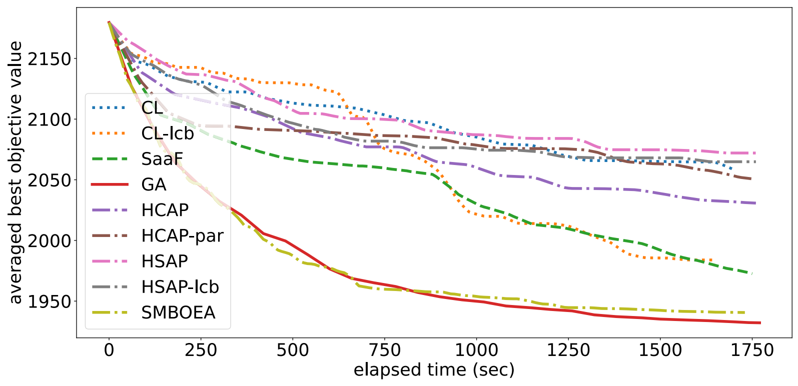

Fig. 13. Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. 10-D CEC2015-15 problem. Convergence profile in terms of best objective values averaged over the 20 runs of the experiment. Dotted lines represent qEGO-like approaches, the dashed line represent the P-SAGA approach, the plain line represents the GA and the dash-dotted lines represent the hybrid methods.

Table 13
Hybrid algorithms robustness to search landscapes. Dispersion-Driven Hybrid Acquisition Process. Friedman's average ranks of the competing algorithms in case of moderately expensive problem (budget expressed as a limited execution time). Smaller ranks (darker backgrounds) indicate better generalization performances across the CEC2015 test suite while larger ranks (lighter backgrounds) indicate lower generalization performances.

elapsed time (sec)	GA	SaaF	SMBOEA	CL-lcb	MS-qEI	HCAP	HSAP-lcb	DDHAP
				CEC2015 10-D				
60	5	5.2	3.4	4.8	2.2	5.53	5.8	4.06
120	4.06	4.93	3.13	5	3	5.66	6.2	4
240	3.6	4.86	3.13	5.4	3.66	5.73	6	3.6
480	3.13	5.13	2.8	5.53	4.26	5.93	6.2	3
960	2.73	4.33	3.53	5.6	4.66	6.33	5.93	2.86
1200	2.66	4	3.46	6	4.86	6.2	5.86	2.93
1800	2.86	2.86	4.13	6.46	5.26	5.53	5.8	3.06
				CEC2015 30-D				
60	5.26	5.53	3.8	4.26	3.53	5.66	5.06	2.86
120	4.73	5.8	4	3.8	2.46	5.66	5.46	4.06
240	4.46	5.6	3.66	4.93	2.46	5.73	5.46	3.66
480	3.80	5.40	3.33	6.33	3.13	6.20	5.13	2.66
960	2.33	4	3.53	7.4	4.66	6.26	4.73	3.06
1200	2.26	3.73	3.86	7.53	5.2	5.66	4.73	3
1800	2.33	3.2	4.2	7.93	5.86	4.93	4.59	2.93

Table 14
Hybrid algorithms robustness to search landscapes. Dispersion-Driven Hybrid Acquisition Process. Friedman's average ranks of the competing algorithms in case of very expensive problems (budget expressed as a limited number of expensive evaluations). Smaller ranks (darker backgrounds) indicate better generalization performances across the CEC2015 test suite while larger ranks (lighter backgrounds) indicate lower generalization performances.

n_{sim}	GA	SaaF	SMBOEA	CL-lcb	MS-qEI	HCAP	HSAP-lcb	DDHAP
				CEC2015 10-D				
72	6.66	5.13	4.06	4.13	1.86	4.4	5.6	4.13
108	6.2	5.26	3.93	4.06	2.33	4.59	5.93	3.66
144	5.8	4.86	3.66	4.13	2.66	4.46	6.2	4.2
216	5.13	4.86	3.46	4.73	3.4	4.93	5.86	3.6
288	4.66	4.93	3.06	5.06	3.53	5.53	5.93	3.26
360	4.33	5.33	3	4.93	3.6	5.66	5.93	3.2
432	3.93	5.46	2.93	4.8	3.93	5.73	6.2	3
504	3.66	5.26	2.8	5.06	4	5.93	6.26	3
				CEC2015 30-D				
72	7	5.93	4.93	2.2	2.53	4.8	4.59	4
108	6.66	5.66	4.66	2.86	2.26	4.93	4.8	4.13
144	6.4	5.73	4.59	3.46	2.2	4.59	4.59	4.4
216	6.06	5.8	4.06	4.13	2.06	5.26	5	3.6
288	5.53	5.4	4.4	4.66	2.4	5.26	5	3.33
360	5.53	5.4	4	4.73	2.4	5.46	5.2	3.26
432	5.2	5.46	3.86	5.06	2.73	5.66	5.26	2.73
504	5	5.53	3.6	5.66	3	5.6	5.26	2.33

The design and calibration of the new DDHAP is based on the previous results obtained on the CEC2015 test suite and exposed in Section 6.1. This fact raises the question of the applicability of DDHAP to other problems. Indeed, the CEC2015 test suite is only made of 30 optimization problems while the whole space of possible objective functions one can think of is infinite. On the one hand, we do not expect DDHAP to provide a good optimization quality for problems with a high number of decision variables. Firstly, because the numerical experiments considered in this study are limited to 30-D objective functions. Secondly, recent algorithms proposed in the field of surrogate-based optimization harness specific techniques to handle high dimensional decision vectors such as sub-space embedding [62] or additive models [63]. None of these techniques are considered in this study. On the other hand, the CEC2015 test suite covers multiple landscape features that are known to hamper the optimization procedure [33]. Hybridization and composition of such landscape features are also taken into account. Thanks to the CEC2015 test suite, we expect DDHAP to be applicable to a large range of optimization problem.

7. Conclusions

The acquisition process is in charge of identifying new promising solutions in parallel surrogate-based optimization algorithms. Multiple ways were devised to meet this purpose and it is of primary importance to study the suitability of each type of acquisition process with respect to the characteristic of the optimization problem at hand. This article specifically focuses on the Covid-19 contact reduction problem that consisted to determine the optimal per-age contact mitigation plan to reduce the impact of the pandemic. This task is challenging because of the computationally expensive evaluation of the black-box objective function and the binary simulation-based constraint. The joint combination of parallel computing, surrogate modeling and evolutionary computations is required to address such complexities. The surrogate model, the promisingness criterion and the acquisition process represent the three dimensions of the design space of parallel surrogate-based algorithms as considered in this study. We compared empirically different possible options for each dimension which allowed identifying a bound value to classify problems as very or moderately expensive. Consequently, we proposed a new hybrid successive acquisition process that yields the best resolution of the Covid-19 contact reduction problem and demonstrates a good parallel scalability when compared to competing approaches. The hybrid scheme relies on both evolutionary operators and auxiliary optimization of the promisingness criterion to issue new candidates for expensive evaluation. The switch from one acquisition process to another ensures robustness over budgets thus allowing to best address moderately and very expensive problems.

The generalization of hybridization to a large range of search landscapes is an open research question. In this article, we proposed to invoke the dispersion metric, a landscape analysis tools, to drive the selection of the acquisition processes during the optimization exercise. The auspicious results reported in this paper suggest to further investigate this topic in future works. Many strategies, such as exploratory landscape analysis, are available to characterize the landscape at hand and many other acquisition processes can be employed such as those of swarm optimization and probability density estimation. Another future direction of study on the COVID-19 contact reduction problem is to build a surrogate model for the simulation-based constraint. As this constraint is binary, a classifier could be used to help identifying the feasible region.

CRediT authorship contribution statement

G. Briffoteaux: Conceptualization, Methodology, Investigation, Software, Visualization, Writing – original draft, Writing – review & editing. **N. Melab:** Conceptualization, Supervision, Validation, Writing – original draft. **M. Mezmaz:** Conceptualization, Supervision, Validation. **D. Tuyttens:** Conceptualization, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Some algorithms are available online, others will be in the future. Some part of other algorithms are hidden for confidentiality.

Acknowledgments

We thank Dr Romain Ragonnet and the Department of Public Health and Preventive Medicine at Monash University in Melbourne, Australia, for helping us to set the AuTuMN simulator.

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

Appendix A. Schwefel–Rastrigin–Rosenbrock

See Figs. A.14–A.16.

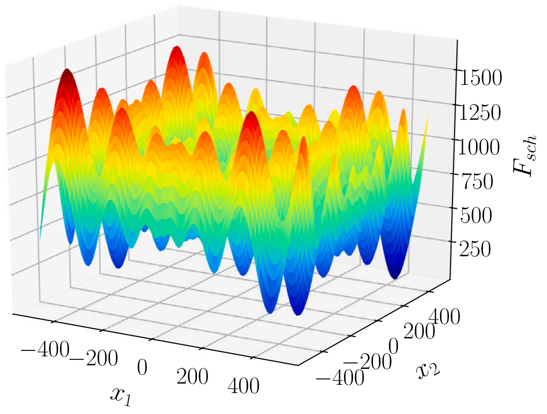

Fig. A.14. Search landscape provided by the 2-D Schwefel function.

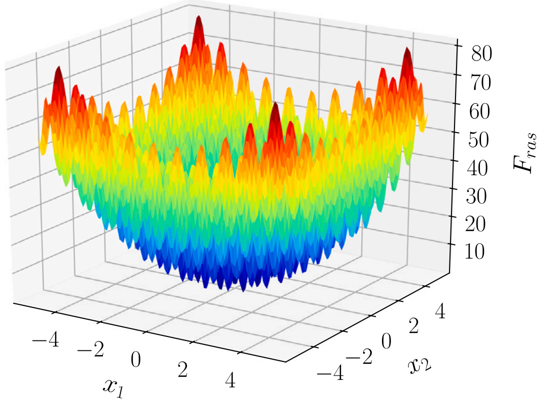

Fig. A.15. Search landscape provided by the 2-D Rastrigin function.

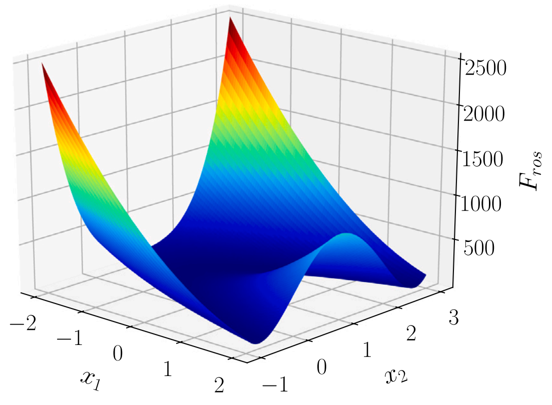

Fig. A.16. Search landscape provided by the 2-D Rosenbrock function.

Table B.15
Calibration of BNN_MCD. Possible values for the hyper-parameters.

Symbol	Name	Calibration method
n_{hl}	number of fully-connected hidden layers	grid search $\{1; 2; 5; 8; 10\}$
m_u	number of units per layer	grid search $\{256; 512; 1024; 2048; 4096\}$
λ_{decay}	weight decay coefficient	grid search $\{10^{-3}; 10^{-2}; 10^{-1}; 1\}$
l	Normal standard deviation for weights initialization	grid search $\{10^{-2}; 10^{-1}; 1; 10; 100\}$
p_{drop}	dropout probability	grid search $\{0.005; 0.05; 0.1; 0.3; 0.5\}$
$h()$	activation function	Relu [23]
ξ	Adam initial learning rate	0.001 [23]

Appendix B. Calibration of BNN_MCD

The hyper-parameters of BNN_MCD are listed in Table B.15 along with the possible values for the grid search. Each of the 2500 possible BNN_MCD configurations is tested on the Schwefel–Rastrigin–Rosenbrock test suite using training sets of 256 samples and validation sets of 1024 samples obtained through Latin Hypercube Sampling. The calibration is based on the bi-objective minimization of the validation mean squared error (VMSE) and minimization of the negative average validation log-likelihood (NAVLL), both calculated after a 50-epoch training. For a validation sample $(\boldsymbol{x_{val}}, y_{val})$, the validation log-likelihood is given by:

$$logsumexp\left(\frac{-\tau}{2}(y_{val} - \hat{f}_i(\boldsymbol{x_{val}}))^2\right) - \log(n_{sub}) - \frac{1}{2}\log(2\pi) + \frac{1}{2}\log(\tau)$$
$$\text{where } \tau = \frac{1 - p_{drop}}{2.n.l.\lambda_{decay}} \qquad \text{(B.1)}$$

where the number of sub-networks is arbitrarily fixed to $n_{sub} = 5$ for the moment. Averaging over the validation set yields the NAVLL. The NAVLL incorporates the model uncertainty and captures how well the model fits the data with larger values indicating better accuracy [24].

The BNN_MCD variants with two or three occurrences in the non-dominated fronts of the benchmarks are identified. Among them, the configuration displayed in the middle of the non-dominated front for both Schwefel and Rosenbrock is retained. In the middle of the non-dominated front, VMSE and NAVLL are balanced. Moreover, Schwefel and Rosenbrock are favored since the Covid-19-related problem may exhibit similar characteristics. The best non-dominated front according to simultaneous minimization of VMSE and NAVLL are provided in Table B.16, Tables B.17 and B.18 for Schwefel, Rastrigin and Rosenbrock respectively. The configurations with 2 or 3 occurrences in the non-dominated fronts are provided in Table B.19.

Appendix C. Calibration of GA

The parameters of the GA are listed in Table C.20 along with their values for the grid search. The Schwefel, Rastrigin and Rosenbrock functions are optimized 10 independent times with each of the 20 instances of GA obtained by varying the population size and crossover probability. The computational budget allocated to each search amounts to 30 min on 18 computing cores.

The grid search calibration outcomes are displayed as box-plot graphs in Fig. C.17, Figs. C.18 and C.19 for Schwefel, Rastrigin and Rosenbrock respectively. The corresponding main statistics are reported in Table C.21, Tables C.22 and C.23. The best configuration according to the average, median and minimum objective value found at the end of the search is $(n_{pop} = 72; p_c = 0.9)$ for the Schwefel and the Rastrigin problems and $(n_{pop} = 18; p_c = 0.9)$ for the Rosenbrock problem. Setting $n_{pop} = 8$ provides consistently bad results because it induces the idling of 10 computing cores.

Appendix D. Calibration of SaaEF

The parameters of SaaEF are listed in Table D.24 along with their values for the grid search. BNN_MCD and *par-fs-cd* are arbitrarily chosen as surrogate and PC respectively. The early stopping and cross-validation parameters of the BNN_MCD are also consider for tuning. The Schwefel–Rastrigin–Rosenbrock test suite is tackled 10 independent times by every possible SaaEF configuration for a budget amounting to 30 min on 18 computing cores.

The main statistics of the results are exposed in Table D.25, Tables D.26 and D.27 for the Schwefel, Rastrigin and Rosenbrock functions respectively. Increasing n_{chld} grants major opportunity for the reproduction operators to yield auspicious candidates. The number of new samples between consecutive surrogate updates is given by q for which higher values are preferred according to the calibration outcomes. In SaaEF, one surrogate update is carried out per iteration. Increasing the number of surrogate updates per iteration by means of partitioning the population of children into sub-batches has been attempted in the framework of this study but has not demonstrated any benefit.

The retained tuning is the one showing the best average and median objective value at the end of the search on the Schwefel problem and the fourth on the Rosenbrock problem. On Rastrigin, no configuration outperform the GA without surrogate.

Appendix E. Calibration of the PC optimizer in qEGO

A surrogate-free GA is used to optimize the PC in the AP of qEGO. A grid search calibration is performed on the population size and the number of generations considering the values $n_{pop} \in \{50; 100; 150; 200\}$ and $n_{gen} \in \{50; 100; 150; 200\}$. The (GP_RBF, CL, *ada-wang*) qEGO configuration is run ten independent times on the Schwefel and Rosenbrock problem for a computational budget amounting to 30 min in 18 computing units. Although this particular configuration creates a bias in the tuning procedure, the extreme computing load of a full calibration is bypassed.

The box-plots of the final expensive objective values reached at the end of the searches are shown in Figs. E.20 and E.21 for Schwefel and Rosenbrock respectively. The configuration $(n_{pop}, n_{gen}) = (50, 100)$ provides competitive results and is consequently retained for further experiments.

Appendix F. Comparing variants of CL, HSAP and HCAP on CEC2015

See Tables F.28–F.30.

Table B.16
Calibration of BNN_MCD. Best non-dominated front according to minimization of the validation mean squared error (VMSE) and the negative average validation log-likelihood (NAVLL) on the **Schwefel** problem. Ordering according to ascending VMSE from top to bottom. The retained configuration appears in bold.

n_{hl}	m_u	λ_{decay}	l	p_{drop}	VMSE	NAVLL
1	512	1e−3	1e−2	0.05	0.0281	6.0782
1	1024	1e−3	1e−2	0.1	0.0282	5.8554
10	2048	1e−3	1e−2	0.3	0.0284	5.2803
10	1024	1e−3	1e−2	0.3	0.0285	5.273
10	2048	1e−3	1e−2	0.5	0.0287	4.5629
1	1024	1e−3	1e−2	0.5	0.0289	4.5159
8	4096	1e−2	1e−2	0.005	0.0289	2.6749
8	4096	1e−2	1e−2	0.1	0.0289	2.5988
2	4096	1e−2	1e−2	0.1	0.0291	2.5987
1	1024	1e−2	1e−2	0.1	0.0291	2.591
10	4096	1e−2	1e−2	0.3	0.0292	2.4188
8	2048	1e−2	1e−2	0.3	0.0293	2.4183
1	2048	1e−2	1e−2	0.3	0.0293	2.4124
2	2048	1e−2	1e−2	0.5	0.0295	2.196
10	2048	1e−2	1e−2	0.5	0.0297	2.1956
1	2048	1e−1	1e−2	0.005	0.0299	1.2783
1	4096	1e−1	1e−2	0.005	0.0299	1.2782
1	4096	1e−1	1e−2	0.05	0.0301	1.2542
1	**1024**	**1e-1**	**1e-2**	**0.1**	0.0304	1.2257
2	2048	1e−1	1e−2	0.1	0.0305	1.2257
1	512	1e−1	1e−2	0.1	0.0308	1.2256
1	2048	1e−1	1e−2	0.3	0.0309	1.0946
1	512	1e−1	1e−2	0.3	0.0311	1.0946
1	2048	1e−1	1e−2	0.5	0.0312	0.9208
1	4096	1e−1	1e−2	0.5	0.0317	0.9208
2	4096	1e−1	1e−2	0.5	0.0321	0.9207
8	4096	1	1e−2	0.005	0.033	0.1026
10	4096	1	1e−2	0.05	0.0332	0.0793
5	4096	1	1e−2	0.1	0.0335	0.0521
10	4096	1	1e−2	0.3	0.034	−0.074
2	2048	1	1e−2	0.3	0.0348	−0.074
2	2048	1	1e−2	0.5	0.035	−0.2428
10	4096	1	1e−2	0.5	0.0351	−0.2428
1	256	1	1e−1	0.3	0.0925	−1.227
1	256	1	1e−1	0.5	0.1254	−1.3953
1	256	1	1	0.05	2667.34	−2.2039
1	256	1	1	0.3	3001.3754	−2.3104
1	256	1	1	0.5	3230.1306	−2.4348

Table B.17
Calibration of BNN_MCD. Best non-dominated front according to minimization of the validation mean squared error (VMSE) and the negative average validation likelihood (NAVLL) on the **Rastrigin** problem. Ordering according to ascending VMSE from top to bottom.

n_{hl}	m_u	λ_{decay}	l	p_{drop}	VMSE	NAVLL
8	2048	1e−3	1e−2	0.1	0.0323	6.3019
10	4096	1e−3	1e−2	0.3	0.0324	5.5486
10	2048	1e−3	1e−2	0.3	0.0328	5.546
10	4096	1e−3	1e−2	0.5	0.033	4.7496
10	4096	1e−2	1e−2	0.05	0.0332	2.6757
8	4096	1e−2	1e−2	0.05	0.0332	2.6757
10	4096	1e−2	1e−2	0.1	0.0334	2.6337
5	1024	1e−2	1e−2	0.1	0.0336	2.6328
2	4096	1e−2	1e−2	0.3	0.0337	2.4471
8	2048	1e−2	1e−2	0.3	0.0338	2.4454
10	2048	1e−2	1e−2	0.3	0.0339	2.4452
2	4096	1e−2	1e−2	0.5	0.0339	2.2152
1	4096	1e−1	1e−2	0.005	0.0345	1.2823
2	2048	1e−1	1e−2	0.05	0.0347	1.2577
2	4096	1e−1	1e−2	0.1	0.0348	1.229
1	4096	1e−1	1e−2	0.3	0.0356	1.0974
1	2048	1e−1	1e−2	0.3	0.0356	1.0972
5	4096	1e−1	1e−2	0.3	0.0364	1.0972
1	1024	1e−1	1e−2	0.5	0.0365	0.9229
1	2048	1e−1	1e−2	0.5	0.0372	0.9229
8	4096	1e−1	1e−2	0.5	0.0375	0.9228
10	4096	1e−1	1e−2	0.5	0.0378	0.9228
1	4096	1	1e−2	0.005	0.0384	0.1029
1	4096	1	1e−2	0.1	0.0385	0.0524
2	4096	1	1e−2	0.1	0.0399	0.0524

(*continued on next page*)

Table B.17 (*continued*).

n_{hl}	m_u	λ_{decay}	l	p_{drop}	VMSE	NAVLL
1	4096	1	1e−2	0.3	0.0403	−0.0736
5	4096	1	1e−2	0.3	0.0406	−0.0737
1	4096	1	1e−2	0.5	0.0411	−0.2426
2	4096	1	1e−2	0.5	0.0416	−0.2426
1	2048	1	1e−2	0.5	0.042	−0.2426
8	1024	1	1e−2	0.5	0.0461	−0.2426
1	256	1	1e−1	0.3	0.1234	−1.227
1	256	1	1e−1	0.5	0.129	−1.3953
1	512	1	1e−1	0.5	0.2099	−1.3953
1	1024	1	1e−1	0.5	0.4058	−1.3953
1	256	1	1	0.1	2634.2219	−2.2082
1	256	1	1	0.3	2967.1152	−2.2736
1	256	1	1	0.5	3345.4199	−2.3886

Table B.18
Calibration of BNN_MCD. Best non-dominated front according to minimization of the validation mean squared error (VMSE) and the negative average validation log-likelihood (NAVLL) on the **Rosenbrock** problem. Ordering according to ascending VMSE from top to bottom. The retained configuration appears in bold.

n_{hl}	m_u	λ_{decay}	l	p_{drop}	VMSE	NAVLL
1	2048	1e−3	1e−2	0.005	0.0235	5.6153
1	2048	1e−3	1e−2	0.05	0.0236	5.5204
1	4096	1e−3	1e−2	0.1	0.0243	5.2677
1	4096	1e−3	1e−2	0.3	0.0245	4.7687
1	2048	1e−3	1e−2	0.5	0.0252	4.2843
1	4096	1e−3	1e−2	0.5	0.0256	4.2509
2	4096	1e−3	1e−2	0.5	0.0309	4.2273
1	512	1e−2	1e−2	0.005	0.0311	2.6421
1	2048	1e−2	1e−2	0.1	0.0312	2.569
1	4096	1e−2	1e−2	0.1	0.0316	2.569
1	2048	1e−2	1e−2	0.3	0.0319	2.3986
1	4096	1e−2	1e−2	0.3	0.0319	2.3972
1	2048	1e−2	1e−2	0.5	0.0323	2.1841
1	4096	1e−2	1e−2	0.5	0.0325	2.1835
1	256	1e−2	1e−1	0.05	0.0385	1.2509
1	**1024**	**1e−1**	**1e−2**	**0.1**	0.0431	1.2362
1	1024	1e−1	1e−2	0.3	0.0436	1.1028
1	4096	1e−1	1e−2	0.3	0.0437	1.1026
1	512	1e−1	1e−2	0.3	0.0441	1.1025
1	1024	1e−1	1e−2	0.5	0.0443	0.9264
1	4096	1	1e−2	0.05	0.045	0.0805
2	4096	1	1e−2	0.05	0.0457	0.0805
2	4096	1	1e−2	0.1	0.0459	0.0532
1	4096	1	1e−2	0.3	0.0461	−0.0731
1	2048	1	1e−2	0.3	0.0461	−0.0731
1	1024	1	1e−2	0.3	0.0467	−0.0731
1	4096	1	1e−2	0.5	0.0468	−0.2421
1	512	1	1e−2	0.5	0.0477	−0.2421
8	2048	1	1e−2	0.5	0.0486	−0.2422
1	256	1e−1	1e−1	0.5	0.0659	−0.2423
1	512	1e−1	1e−1	0.5	0.0756	−0.2423
1	256	1	1e−1	0.1	0.1148	−1.1012
1	256	1	1e−1	0.5	0.132	−1.3952
1	512	1	1e−1	0.5	0.2207	−1.3953
1	2048	1	1e−1	0.5	0.7305	−1.3953
1	4096	1	1e−1	0.5	1.4349	−1.3953
2	4096	1	1e−1	0.5	424.7706	−1.3953
1	256	1	1	0.05	2658.1091	−2.1998
1	256	1	1	0.3	2677.07	−2.3144
1	256	1	1	0.5	3048.7062	−2.4446

Table B.19
Calibration of BNN_MCD. BNN_MCD configurations that appear in the non-dominated front of 2 or 3 **benchmark problems**. The retained configuration appears in bold.

n_{hl}	m_u	λ_{decay}	l	p_{drop}	Occurrences
1	256	1	1	0.5	3
1	256	1	1	0.3	3
1	256	1	1e−1	0.5	3
1	512	1e−1	1e−2	0.3	2
1	2048	1e−1	1e−2	0.3	2
2	4096	1	1e−2	0.1	2
1	1024	1e−1	1e−2	0.5	2
1	4096	1	1e−2	0.3	2
1	4096	1e−1	1e−2	0.005	2
10	2048	1e−3	1e−2	0.3	2
1	2048	1e−2	1e−2	0.3	2
1	2048	1e−1	1e−2	0.5	2
1	**1024**	**1e-1**	**1e-2**	**0.1**	2
1	256	1	1	0.05	2
1	256	1	1e−1	0.3	2
1	512	1	1e−1	0.5	2
8	2048	1e−2	1e−2	0.3	2
1	4096	1	1e−2	0.5	2
1	4096	1e−1	1e−2	0.3	2

Table C.20
Calibration of the surrogate-free parallel GA. Possible values for the parameters.

Symbol	Name	Calibration method
n_{pop}	population size	grid search $\{8; 18; 36; 72; 144\}$
p_c	cross-over probability	grid search $\{0.3; 0.5; 0.7; 0.9\}$
η_c	cross-over distribution index	10 [44]
p_m	mutation probability	$\frac{1}{d}$ [45]
η_m	mutation distribution index	50 [44]
n_t	tournament size	2 [32]

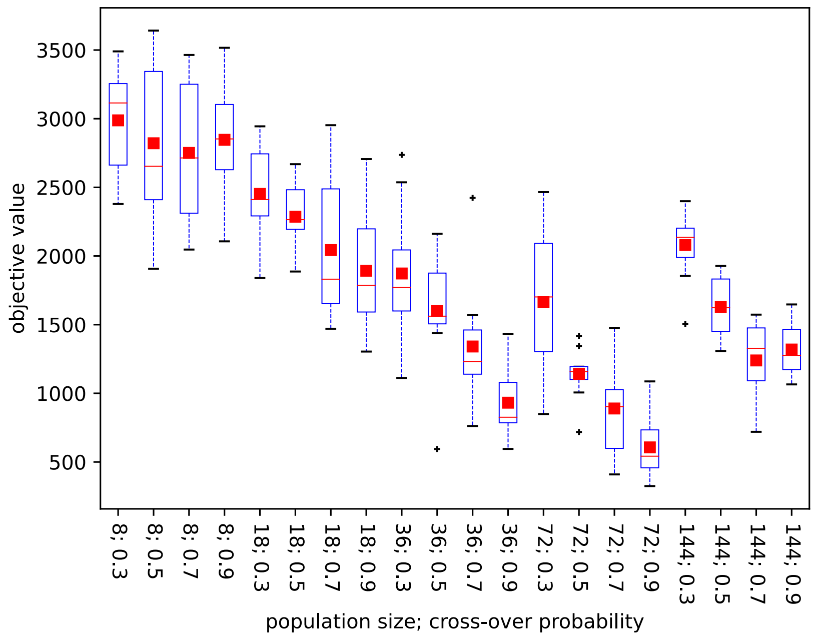

Fig. C.17. **Calibration of the surrogate-free parallel GA**. Distribution of the best objective values from the 10 repetitions of the experiments on the **Schwefel** problem. Average values are depicted by red squares.

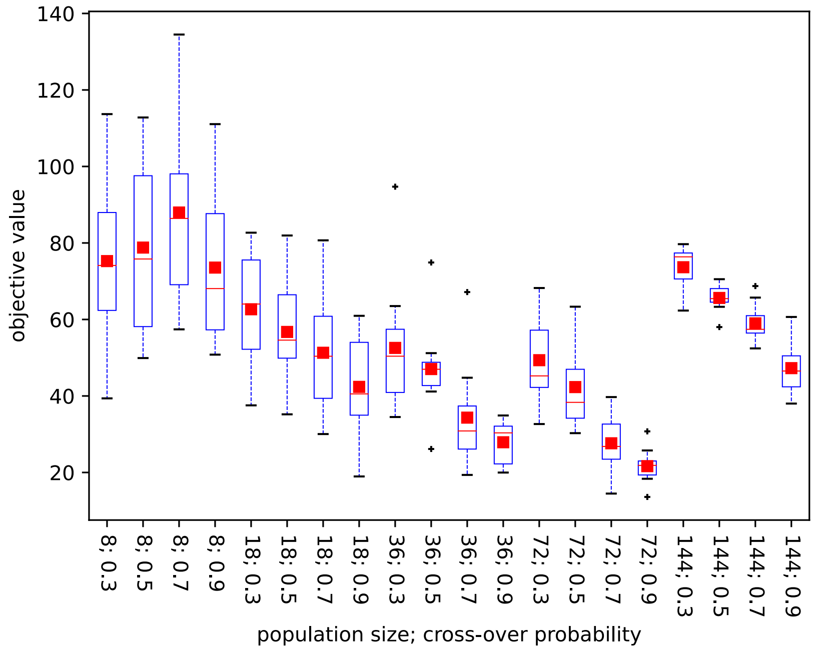

Fig. C.18. Calibration of the surrogate-free parallel GA. Distribution of the best objective values from the 10 repetitions of the experiments on the **Rastrigin** problem. Average values are depicted by red squares.

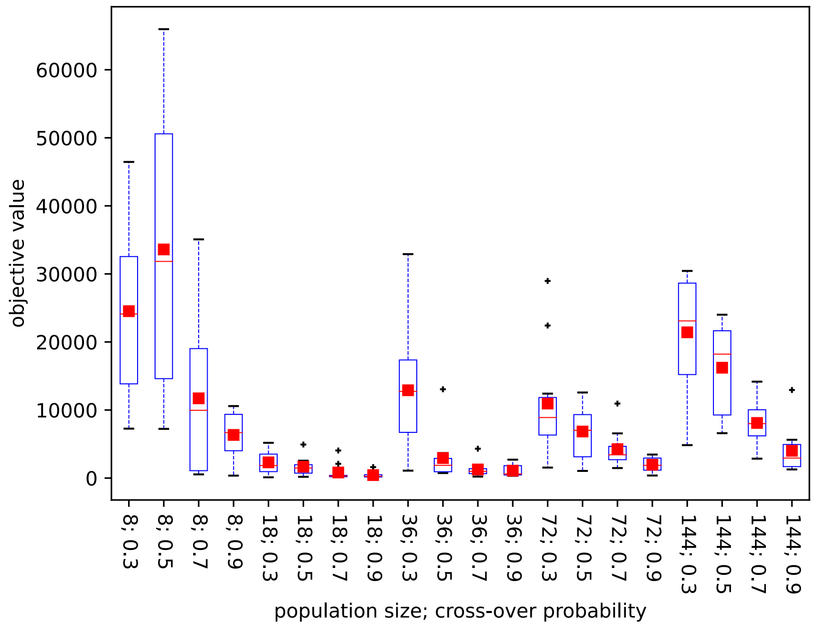

Fig. C.19. Calibration of the surrogate-free parallel GA. Distribution of the best objective values from the 10 repetitions of the experiments on the **Rosenbrock** problem. Average values are depicted by red squares.

Table C.21
Calibration of the surrogate-free parallel GA. Statistics of the distribution of the best objective values from the 10 repetitions of the experiments on the **Schwefel** problem. Ordering according to ascending average best objective value from top to bottom. The best value for each column appears in bold.

n_{pop}	p_c	Average	Median	Minimum	Variance
72	0.9	**606.16**	**542.0**	**324.5**	47,011.53
72	0.7	889.47	902.62	409.36	118,702.65
36	0.9	931.69	825.79	595.2	67,856.63
72	0.5	1141.93	1157.35	718.28	**32,334.32**
144	0.7	1239.44	1328.12	719.76	76,650.15
144	0.9	1318.88	1276.17	1064.83	35,878.64
36	0.7	1340.92	1231.3	761.9	175,525.18
36	0.5	1599.24	1561.78	594.54	160,909.86
144	0.5	1629.41	1623.04	1306.89	43,485.15
72	0.3	1663.67	1701.82	849.11	259,982.01
36	0.3	1872.68	1771.19	1111.97	208,935.32
18	0.9	1892.87	1787.23	1304.08	161,087.18
18	0.7	2043.18	1831.31	1470.45	273,082.05
144	0.3	2079.94	2136.23	1505.05	62,635.81
18	0.5	2286.32	2264.47	1887.0	55,069.82
18	0.3	2452.32	2411.11	1840.13	109,126.93
8	0.7	2750.68	2714.23	2047.0	278,623.58
8	0.5	2821.03	2653.74	1907.67	312,750.77
8	0.9	2846.3	2852.68	2106.74	173,903.74
8	0.3	2987.9	3114.64	2378.45	144,095.13

Table C.22
Calibration of the surrogate-free parallel GA. Statistics of the distribution of the best objective values from the 10 repetitions of the experiments on the **Rastrigin** problem. Ordering according to ascending average best objective value from top to bottom. The best value for each column appears in bold.

n_{pop}	p_c	Average	Median	Minimum	Variance
72	0.9	**21.65**	**21.85**	**13.6**	18.79
72	0.7	27.63	26.79	14.48	48.61
36	0.9	27.9	30.35	19.97	28.24
36	0.7	34.33	30.85	19.35	168.97
72	0.5	42.32	38.33	30.28	114.27
18	0.9	42.37	40.54	18.96	174.72
36	0.5	47.03	46.97	26.13	130.18
144	0.9	47.27	46.52	38.01	40.08
72	0.3	49.35	45.24	32.66	119.86
18	0.7	51.29	50.38	30.04	215.73
36	0.3	52.56	50.39	34.49	283.84
18	0.5	56.74	54.59	35.2	183.72
144	0.7	58.97	57.42	52.43	23.21
18	0.3	62.66	64.03	37.54	204.24
144	0.5	65.62	65.45	58.01	**10.98**
8	0.9	73.56	68.07	50.8	357.22
144	0.3	73.65	76.37	62.32	32.3
8	0.3	75.26	74.12	39.37	444.26
8	0.5	78.79	75.8	49.9	507.6
8	0.7	87.96	86.41	57.41	586.07

Table C.23
Calibration of the surrogate-free parallel GA. Statistics of the distribution of the best objective values from the 10 repetitions of the experiments on the **Rosenbrock** problem. Ordering according to ascending average best objective value from top to bottom. The best value for each column appears in bold.

n_{pop}	p_c	Average	Median	Minimum	Variance
18	0.9	**433.91**	**259.33**	**74.26**	**191,251.85**
18	0.7	820.67	295.8	110.05	1,466,702.09
36	0.9	1092.0	592.06	336.37	658,936.44
36	0.7	1261.31	984.69	216.17	1,191,134.25
18	0.5	1665.09	1477.32	184.67	1,661,523.63
72	0.9	1959.02	1872.94	373.53	1,082,062.41
18	0.3	2307.74	1843.04	110.43	2,586,536.83
36	0.5	2942.54	1875.14	736.89	12,259,372.39
144	0.9	4001.54	2935.33	1267.18	11,060,605.09
72	0.7	4252.02	3417.06	1469.76	6,778,061.45
8	0.9	6337.49	6684.22	358.06	11,193,763.83
72	0.5	6839.66	7021.64	1050.14	15,688,530.4
144	0.7	8099.69	8001.32	2850.41	11,235,759.19
72	0.3	10,946.07	8896.91	1538.18	66,445,555.65
8	0.7	11,717.1	9962.93	539.94	120,357,282.62
36	0.3	12,901.81	12,731.52	1088.82	78,500,881.02
144	0.5	16,214.5	18,213.14	6596.36	45,565,006.41

(*continued on next page*)

Table C.23 (*continued*).

n_{pop}	p_c	Average	Median	Minimum	Variance
144	0.3	21,415.34	23,089.57	4840.79	63,520,417.44
8	0.3	24,522.5	24,112.23	7262.69	155,115,614.92
8	0.5	33,595.52	31,836.06	7229.95	395,710,854.33

Table D.24
Calibration of SaaEF. Possible values for the parameters.

Symbol	Name	Calibration method
n_{chld}	children per iteration	grid search $\{144; 288\}$
q	exp. eval. per iteration	$72 = 0.25 * n_{chld}$ [26,46]
n_{pred}	predictions per iteration	$72 = 0.25 * n_{chld}$ [26,46]
n_{disc}	discardings per iteration	$144 = n_{chld} - q - n_{pred}$
(δ_{ES}, n_{ES})	BNN_MCD early stopping	grid search $\{(10^{-4}, 8), (10^{-8}, 32)\}$
	2-fold cross-validation	grid search {yes, no}

Table D.25
Calibration of SaaEF. Statistics of the distribution of the best expensive objective values from the 10 repetitions of the experiments on the **Schwefel** problem. Ordering according to ascending average best expensive objective value from top to bottom. The best value for each column appears in bold.

n_{chld}	(δ_{ES}, n_{ES}) cross. val.	Average	Median	Minimum	Variance	q
288	$(10^{-8}, 32)$ yes	**516.82**	**480.87**	**190.58**	66,999.17	72
–	No surrogate	607.91	503.25	253.29	71,295.35	–
144	$(10^{-8}, 32)$ yes	685.72	558.1	276.4	210,240.3	36
288	$(10^{-8}, 32)$ no	872.15	828.71	483.86	**58,040.28**	72
288	$(10^{-4}, 8)$ yes	893.4	820.97	502.66	80,785.52	72
288	$(10^{-4}, 8)$ no	1011.34	926.27	650.28	73,748.54	72
144	$(10^{-4}, 8)$ yes	1060.04	1004.16	649.05	65,682.39	36
144	$(10^{-8}, 32)$ no	1085.84	1103.41	686.69	66,753.01	36
144	$(10^{-4}, 8)$ no	1125.48	1085.87	721.91	89,206.08	36

Table D.26
Calibration of SaaEF. Statistics of the distribution of the best expensive objective values from the 10 repetitions of the experiments on the **Rastrigin** problem. Ordering according to ascending average best expensive objective value from top to bottom. The best value for each column appears in bold.

n_{chld}	(δ_{ES}, n_{ES}) cross. val.	Average	Median	Minimum	Variance	q
–	No surrogate	**23.30**	**22.66**	**12.99**	30.58	–
144	$(10^{-8}, 32)$ no	29.31	28.47	16.02	86.36	36
288	$(10^{-8}, 32)$ yes	29.4	27.0	21.39	44.37	72
288	$(10^{-4}, 8)$ yes	31.06	34.95	15.98	68.34	72
288	$(10^{-8}, 32)$ no	33.41	31.92	19.56	67.76	72
144	$(10^{-8}, 32)$ yes	34.76	36.64	23.65	34.01	36
144	$(10^{-4}, 8)$ no	35.38	34.6	19.21	166.75	36
144	$(10^{-4}, 8)$ yes	35.68	35.87	28.79	**17.78**	36
288	$(10^{-4}, 8)$ no	38.08	33.3	21.93	193.38	72

Table D.27
Calibration of SaaEF. Statistics of the distribution of the best expensive objective values from the 10 repetitions of the experiments on the **Rosenbrock** problem. Ordering according to ascending average best expensive objective value from top to bottom. The best value for each column appears in bold.

n_{chld}	(δ_{ES}, n_{ES}) cross. val.	Average	Median	Minimum	Variance	q
144	$(10^{-8}, 32)$ yes	**1096.8**	810.92	446.3	488,068.36	36
144	$(10^{-4}, 8)$ yes	1105.28	**755.04**	339.37	1,006,032.14	36
–	No surrogate	1191.14	757.03	246.71	1,557,771.06	–
288	$(10^{-8}, 32)$ yes	1359.59	1268.25	745.12	**336,182.97**	72
288	$(10^{-4}, 8)$ yes	1407.53	1052.41	**198.69**	869,984.37	72
288	$(10^{-4}, 8)$ no	1414.96	1293.39	280.94	732,488.17	72
288	$(10^{-8}, 32)$ no	1989.58	1973.39	528.56	1,482,968.13	72
144	$(10^{-4}, 8)$ no	2019.27	1034.21	392.79	4,621,452.92	36
144	$(10^{-8}, 32)$ no	4387.67	3873.49	1389.24	5,623,346.82	36

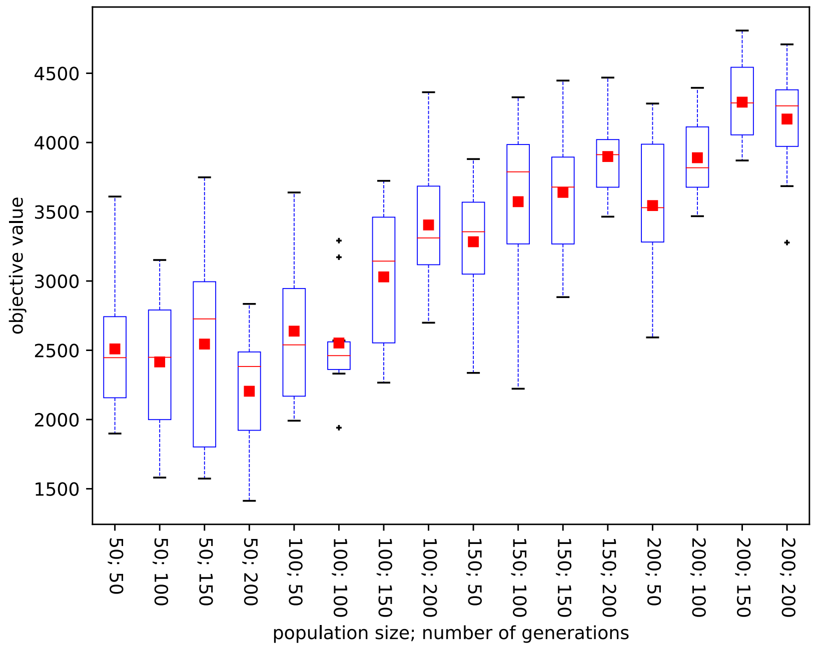

Fig. E.20. Calibration of the PC optimizer in qEGO. Distribution of the best expensive objective values from the 10 repetitions of the experiments on the **Schwefel** problem. Average values are depicted by red squares.

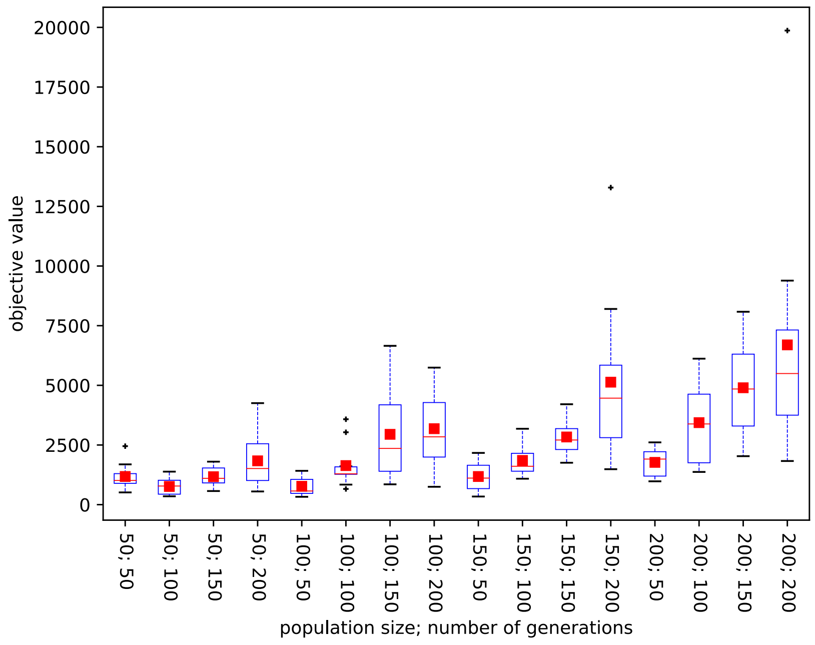

Fig. E.21. Calibration of the PC optimizer in qEGO. Distribution of the best expensive objective values from the 10 repetitions of the experiments on the **Rosenbrock** problem. Average values are depicted by red squares.

Table F.28
Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. Comparison of CL and CL-lcb. Number of problems for which one method is better than the other according to the average best expensive objective value computed over the 20 repetitions of the experiments.

Computational budget	Problem dimension	CL	CL-lcb
108 exp. eval.	10	7	8
	30	7	8
30 min. 18 cores	10	2	13
	30	6	9

Table F.29
Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. Comparison of HSAP and HSAP-lcb. Number of problems for which one method is better than the other according to the average best expensive objective value computed over the 20 repetitions of the experiments.

Computational budget	Problem dimension	HSAP	HSAP-lcb
108 exp. eval.	10	9	6
	30	10	5
30 min. 18 cores	10	4	11
	30	5	10

Table F.30
Hybrid algorithms robustness to search landscapes. Experiments on the CEC2015 test suite. Comparison of HCAP and HCAP-par. Number of problems for which one method is better than the other according to the average best expensive objective value computed over the 20 repetitions of the experiments.

Computational budget	Problem dimension	HSAP	HSAP-lcb
108 exp. eval.	10	9	6
	30	9	6
30 min. 18 cores	10	7	8
	30	8	7

References

[1] J. Stork, A.E. Eiben, T. Bartz-Beielstein, A new taxonomy of global optimization algorithms, Nat. Comput.: Int. J. 21 (2) (2022) 219–242.
[2] C.M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, Berlin, Heidelberg, 2006.
[3] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm Evol. Comput. 1 (2) (2011) 61–70, http://dx.doi.org/10.1016/j.swevo.2011.05.001.
[4] X. Wang, Y. Jin, S. Schmitt, M. Olhofer, Recent advances in Bayesian optimization, ACM Comput. Surv. 55 (13s) (2023).
[5] D. Jones, A taxonomy of global optimization methods based on response surfaces, J. Global Optim. 21 (2001) 345–383.
[6] Y. Jin, M. Olhofer, B. Sendhoff, On evolutionary optimization with approximate fitness functions, in: Proceedings of the 2Nd Annual Conference on Genetic and Evolutionary Computation, GECCO '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 786–793.
[7] G. Briffoteaux, M. Gobert, R. Ragonnet, J. Gmys, M. Mezmaz, N. Melab, D. Tuyttens, Parallel surrogate-assisted optimization: Batched Bayesian neural network-assisted GA versus q-EGO, Swarm Evol. Comput. 57 (2020) 100717.
[8] H. Wang, Y. Jin, J. Doherty, Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems, IEEE Trans. Cybern. 47 (9) (2017) 2664–2677.
[9] N. Masahiro, Y. Akimoto, I. Ono, CMA-ES with learning rate adaptation: Can CMA-ES with default population size solve multimodal and noisy problems? in: Proceedings of the Genetic and Evolutionary Computation Conference, ACM, 2023.
[10] C.E. Rasmussen, Gaussian processes for machine learning, MIT Press, 2006.
[11] R. Regis, C. Shoemaker, A stochastic radial basis function method for the global optimization of expensive functions, INFORMS J. Comput. 19 (2007) 497–509.
[12] M.T.M. Emmerich, K.C. Giannakoglou, B. Naujoks, Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels, IEEE Trans. Evol. Comput. 10 (4) (2006) 421–439.
[13] A. Hebbal, L. Brevault, M. Balesdent, E.G. Talbi, N. Melab, Bayesian Optimization using Deep Gaussian Processes, 2019, arXiv e-prints, arXiv:1905.03350.
[14] G. Briffoteaux, R. Ragonnet, M. Mezmaz, N. Melab, D. Tuyttens, Evolution control for parallel ANN-assisted simulation-based optimization application to tuberculosis transmission control, Future Gener. Comput. Syst. 113 (2020) 454–467.
[15] J.M. Trauer, M.J. Lydeamore, G.W. Dalton, D. Pilcher, M.T. Meehan, E.S. McBryde, A.C. Cheng, B. Sutton, R. Ragonnet, Understanding how Victoria, Australia gained control of its second COVID-19 wave, Nature Commun. 12 (2021).
[16] J.M. Caldwell, E. de Lara-Tuprio, T.R. Teng, M.R.J.E. Estuar, R.F.R. Sarmiento, M. Abayawardana, R.N.F. Leong, R.T. Gray, J.G. Wood, L.V. Le, E.S. McBryde, R. Ragonnet, J.M. Trauer, Understanding COVID-19 dynamics and the effects of interventions in the Philippines: A mathematical modelling study, Lancet Reg. Health - Western Pacific 14 (2021).
[17] F. Rehback, M. Zaefferer, J. Stork, T. Bartz-Beielstein, Comparison of parallel surrogate-assisted optimization approaches, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 1348–1355.
[18] J. Tian, Y. Tan, J. Zeng, C. Sun, Y. Jin, Multiobjective infill criterion driven Gaussian process-assisted particle swarm optimization of high-dimensional expensive problems, IEEE Trans. Evol. Comput. 23 (3) (2019) 459–472.
[19] X. Wang, Y. Jin, S. Schmitt, M. Olhofer, An adaptive Bayesian approach to surrogate-assisted evolutionary multi-objective optimization, Inform. Sci. 519 (2020) 317–331.
[20] X. Ruan, K. Li, B. Derbel, A. Liefooghe, Surrogate assisted evolutionary algorithm for medium scale multi-objective optimisation problems, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference, GECCO '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 560–568.
[21] A.I. J. Forrester, A. Sóbester, A.J. Keane, Constructing a surrogate, in: Engineering Design Via Surrogate Modelling, John Wiley and Sons, Ltd, 2008, pp. 33–76.
[22] J. Snoek, O. Rippel, K. Swersky, R. Kirosn, N. Satish, N. Sundaram, M.M. A Patwary, M. Prabhat, R. Adams, Scalable Bayesian optimization using deep neural networks, Statistics (2015).
[23] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.
[24] Y. Gal, Uncertainty in Deep Learning (Ph.D. thesis), University of Cambridge, 2016.
[25] G. Briffoteaux, R. Ragonnet, M. Mezmaz, N. Melab, D. Tuyttens, Evolution control ensemble models for surrogate-assisted evolutionary algorithms, in: High Performance Computing and Simulation 2020, Barcelona, Spain, 2021, URL https://hal.inria.fr/hal-03332521.
[26] Y. Jin, M. Olhofer, B. Sendhoff, Managing approximate models in evolutionary aerodynamic design optimization, in: Proceedings of the 2001 Congress on Evolutionary Computation, Vol. 1, 2001, pp. 592–599.
[27] L. Shi, K. Rasheed, A survey of fitness approximation methods applied in evolutionary algorithms, in: Computational Intelligence in Expensive Optimization Problems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 3–28.
[28] D.R. Jones, M. Schonlau, W.J. Welch, Efficient global optimization of expensive black-box functions, J. Global Optim. 13 (4) (1998) 455–492.
[29] D. Ginsbourger, R. Le Riche, L. Carraro, Kriging is well-suited to parallelize optimization, in: Computational Intelligence in Expensive Optimization Problems, in: Springer series in Evolutionary Learning and Optimization, springer, 2010, pp. 131–162.
[30] M. Emmerich, A. Giotis, M. Özdemir, T. Bäck, K. Giannakoglou, Metamodel—Assisted evolution strategies, in: Lecture Notes in Computer Science, 2002.
[31] G. Briffoteaux, Parallel Surrogate-Based Algorithms for Solving Expensive Optimization Problems (Ph.D. thesis), Université de Mons, Université de Lille, 2022.
[32] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[33] Q. Chen, B. Liu, Q.F. Zhang, J.J. Liang, P.N. Suganthan, B. Qu, Problem definitions and evaluation criteria for CEC 2015 special session on bound constrained single-objective computationally expensive numerical optimization, 2015.
[34] M. Claeson, S. Hanson, COVID-19 and the Swedish enigma, Lancet 397 (10271) (2021) 259–261.
[35] Z. Michalewicz, D. Dasgupta, R.G. Le Riche, M. Schoenauer, Evolutionary algorithms for constrained engineering problems, Comput. Ind. Eng. 30 (4) (1996) 851–870.
[36] J.M.C Trauer, R. Ragonnet, T.N. Doan, E.S. McBryde, Modular programming for tuberculosis control, the "AuTuMN" platform, BMC Infect. Dis. 17 (1) (2017) 546.
[37] J.M. Caldwell, et al., Modelling COVID-19 in the Philippines: Technical Description of the Model, Technical Report, Monash University, 2020.
[38] R. Ragonnet, et al., Optimising social mixing strategies achieving COVID-19 herd immunity while minimising mortality in six European countries, 2020, medRxiv.
[39] K. Prem, A.R. Cook, M. Jit, Projecting social contact matrices in 152 countries using contact surveys and demographic data, PLoS Comput. Biol. 13 (9) (2017) 1–21.
[40] J.R. Gardner, G. Pleiss, D. Bindel, K.Q. Weinberger, A.G. Wilson, GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration, in: Advances in Neural Information Processing Systems, 2018.
[41] C. Paulson, G. Ragkousis, pyKriging: A Python Kriging Toolkit, 2015, Zenodo.
[42] M.S. Innocente, S.M. Bastos Afonso, J. Sienz, H.M. Davies, Particle swarm algorithm with adaptive constraint handling and integrated surrogate model for the management of petroleum fields, Appl. Soft Comput. 34 (2015) 463–484.
[43] P. Kumar, A. Gupta, Active learning query strategies for classification, regression, and clustering: A survey, J. Comput. Sci. Tech. 35 (4) (2020) 913–945.
[44] K. Deb, P. Nain, An Evolutionary Multi-objective Adaptive Meta-modeling Procedure Using Artificial Neural Networks, in: Evolutionary Computation in Dynamic and Uncertain Environments, vol. 51, Springer, Heidelberg, 2007, pp. 297–322, http://dx.doi.org/10.1007/978-3-540-49774-5_13.
[45] E.G. Talbi, Metaheuristics: From design to implementation, in: Wiley Series on Parallel and Distributed Computing, Wiley, 2009, URL https://books.google.fr/books?id=SIsa6zi5XV8C.
[46] D. Buche, N.N. Schraudolph, P. Koumoutsakos, Accelerating evolutionary algorithms with Gaussian process fitness function models, IEEE Trans. Syst. Man Cybern. C (Applications and Reviews) 35 (2) (2005) 183–194.
[47] P. Moscato, On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts - Towards Memetic Algorithms, C3P Report 826:1989, Caltech Concurrent Computation Program, 1989.
[48] Z. Chen, J. Cao, F. Zhao, J. Zhang, A grouping cooperative differential evolution algorithm for solving partially separable complex optimization problems, Cogn. Comput. 15 (2023) 1–20.
[49] D. Sculley, Web-scale k-means clustering, in: Proceedings of the 19th International Conference on World Wide Web, WWW '10, Association for Computing Machinery, New York, NY, USA, 2010, pp. 1177–1178.
[50] D. Arthur, S. Vassilvitskii, K-means++: the advantages of careful seeding, in: Proceedings of the Symposium on Discrete Algorithms, 2007, pp. 1027–1035.

[51] G. Briffoteaux, pySBO: Python framework for surrogate-based optimization, 2021, https://pysbo.readthedocs.io/.

[52] F. Cappello, et al., Grid'5000: A large scale and highly reconfigurable grid experimental testbed, in: The 6th IEEE/ACM International Workshop on Grid Computing, 2005, 2005.

[53] K.M. Malan, A survey of advances in landscape analysis for optimisation, Algorithms 14 (2) (2021) 40.

[54] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, G. Rudolph, Exploratory landscape analysis, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, Association for Computing Machinery, New York, NY, USA, 2011, pp. 829–836.

[55] P. Kerschke, H. Trautmann, Comprehensive feature-based landscape analysis of continuous and constrained optimization problems using the R-package Flacco, in: Applications in Statistical Computing: From Music Data Analysis To Industrial Quality Improvement, Springer International Publishing, Cham, 2019, pp. 93–123.

[56] M. Lunacek, D. Whitley, The dispersion metric and the CMA evolution strategy, in: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO '06, Association for Computing Machinery, New York, NY, USA, 2006, pp. 477–484.

[57] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[58] P. Kerschke, M. Preuss, S. Wessing, H. Trautmann, Detecting funnel structures by means of exploratory landscape analysis, in: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, GECCO '15, Association for Computing Machinery, New York, NY, USA, 2015, pp. 265–272.

[59] M. Balandat, B. Karrer, D.R. Jiang, S. Daulton, B. Letham, A.G. Wilson, E. Bakshy, BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization, in: Advances in Neural Information Processing Systems, vol. 33, 2020.

[60] D.J. Rezende, S. Mohamed, D. Wierstra, Stochastic backpropagation and approximate inference in deep generative models, in: Proceedings of the 31st International Conference on International Conference on Machine Learning - Vol. 32, ICML '14, JMLR.org, 2014, pp. II–1278–II–1286.

[61] J.T. Wilson, R. Moriconi, F. Hutter, M.P. Deisenroth, The reparameterization trick for acquisition functions, 2017, arXiv abs/1712.00424.

[62] A. Nayebi, A. Munteanu, M. Poloczek, A framework for Bayesian optimization in embedded subspaces, in: Kamalika Chaudhuri, Ruslan Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 4752–4761.

[63] K. Kandasamy, J. Schneider, B. Póczos, High dimensional Bayesian optimisation and bandits via additive models, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Vol. 37, ICML '15, JMLR.org, 2015, pp. 295–304.