Surrogate-assisted cooperative optimization for computationally expensive black-box problems

José Carlos García García[1], Ricardo García-Ródenas[1] and Esteve Codina Sancho[2]

[1]*Departamento de Matemáticas, Escuela Superior de Informática, Universidad de Castilla-La Mancha. Spain*

[2]*Estadística i Investigació Operativa, Universitat Politècnica de Catalunya*

Abstract

Motivation. Most parallel surrogate-assisted optimization algorithms focus only on the mechanisms for generating multiple updating points in each cycle, but a less attention has been paid to the cooperation of several algorithms for producing them.

Methodology. Concerning parallel optimization of expensive black-box functions, a surrogate-assisted cooperative optimization framework with asynchronous communication is presented. A class of parallel surrogate-assisted methods is developed based on the idea of viewing the search infill criterion as bi-objective optimization problem. Each algorithm of this class is named *search agent* and it is the resulting combination of choosing a surrogate model, an exploitation measure, an exploration measure and a multi-objective optimization approach to solve it. Search agents are the basic algorithms on which collaboration mechanisms are established. Many search agents can be defined, and it has been focused on scalarization approaches for bi-objective problem such as $\varepsilon-$constrained method, revisiting Parallel *Constrained Optimization using Response Surfaces* (CORS-RBF) method and *Efficient Global Optimization with Pseudo Expected Improvement* (EGO-PEI) algorithm as instances of search agents that can be used within this cooperative optimization framework. The cooperation between search agents is developed in two ways. First, they share solutions and their objective function values to update their surrogate models. Second, they use the sampled points obtained from different search agents to guide their own search process. Some convergence results for this cooperative framework are given under weak conditions.

Computational highlights. Numerical experiments have been carried out on 17 analytic tests taken from Dixon-Szegö test and BBOB test suite. A numerical comparison between EGO-PEI, Parallel CORS-RBF and a cooperative scheme of them, named CPEI,

shows that CPEI improves the performance of the baseline algorithms. The numerical results evidence the reduction of wall-clock time with respect to the increase in the number of processors.

Keywords: Cooperative optimization; Parallel surrogate-based optimization; Black-box function; Efficient global optimization; Radial basis functions.

---

1. Introduction

In many engineering applications such as thermodynamic analysis, engine design, structural analysis or reservoir simulation, computer simulations are used as models of real systems. The search of the optimal simulation parameters often involves optimizing such simulation models. These applications are remarkable examples of noisy black-box optimization in which the analytical expression of the objective function and/or the constraints are unknown and the objective function is of the stochastic type. The challenges of handling noisy black-box functions are: i) computationally expensive, i.e., these simulations are time consuming and ii) no gradient-based methods can be used and thus, the analytically-based stopping criteria are not available. For example, in problems with black-box functions such as those in [11], each engine simulation takes around 48 hours, or in [27], the evaluation of reservoir simulation may take several days.

Contemporary simulation-based optimization methods include heuristic methods, stochastic approximation and surrogate-assisted methods. Metaheuristics are global optimization methods but require a large number of evaluations for convergence and are impractical for these problems. In order to deal with the high computational cost, surrogate-assisted techniques are commonly used in the literature, mainly because they use all the functional evaluations during the optimization process.

This paper focuses on surrogate-assisted optimization methods (also known as metamodels or *Response Surface Methods* (RSMs)). These methods use surrogate models that are updated in each iteration. The use of multiple points per major iteration allows us to take advantage of parallel computing capabilities and it offers great potential for reducing the wall-clock time to solve a global optimization problem.

The purpose of this research is to develop cooperation strategies to optimize expensive black-box functions. Previous efforts have used multiple independent surrogates or multi-objective optimization techniques to derive multi-point search infill criteria. In this work, we use a complementary approach based on the cooperation of parallel surrogate-assisted optimization methods. To achieve this goal, first, we introduce a formal definition of the

class of algorithms that can cooperate with each other. Each algorithm in the class is named *search agent*. The definition of a search agent requires the introduction of the so-called *exploitation* and *exploration* measures. A combination of both measures defines a bi-objective problem that together with the resolution method defines the search agent. The key issue is that the exploration measures are independent of the surrogate model and this fact allows: i) the search agents may generate q-points per cycle and ii) the coordination between these search agents (parallel surrogate-assisted algorithms).

The use of parallel computing capabilities is not only applicable to the evaluation of the expensive objective function but also in the update of the multiple surrogate models and in the resolution of the optimization problems associated with the search infill criteria.

The remainder of the paper is organized as follows. Section 2 describes the sate-of-the-art approaches. Section 3 introduces the proposed surrogate-assisted cooperative optimization framework, and its theoretical properties are described in Section 4. Section 5 illustrates the performance of these algorithms on a number of benchmark functions. Finally, we give our concluding remarks in Section 6.

2. Previous research

We consider the following optimization problem:

$$\underset{x\in\mathcal{D}}{\text{minimize}}\ f(x) \tag{1}$$

where the objective function is not known in a closed form, i.e. it is a *black-box function*. We will assume that the value of the objective function can be calculated, for example, by doing a simulation or experiment in a laboratory (for instance, chemical) to give the value of f or at least get an approximate value of f which we denote by $\widehat{f}$. This approximation can be due to a truncation of the execution or to the introduction of *noise* in the experiment, when it is not possible to control all the parameters involved.

We assume that the problem (1) exhibits the following features:

1. $\mathcal{D}$ is a compact set of $\mathbb{R}^{\nu}$;
2. $x \in \mathcal{D}$ is a vector of continuous variables;
3. f is continuous on $\mathcal{D}$;
4. there is a single-objective function;
5. noise may be present ($\widehat{f}$ perturbed from f);
6. $\widehat{f}$ is expensive to evaluate;

7. no analytical derivates of $\widehat{f}$ are available ($\widehat{f}$ is a black-box function).

Surrogate-assisted optimization techniques are a successful strategy for solving this kind of computationally expensive optimization problem. A prototype scheme is shown in Algorithm 1.

Algorithm 1 Sequential surrogate-assisted optimization framework

1: (*Sample selection*) Let $t = 0$. Select and evaluate a set X^0 of starting points.
2: (*Construct a surrogate model*) From the data $\left\{(x, \widehat{f}(x))|x \in X^t\right\}$, construct a surrogate model $S_{X^t}(\cdot)$ that approximates the black-box function $f(x)$.
3: (*Search infill criterion*) Select a **new point** y using the surrogate model $S_{X^t}(\cdot)$ and evaluate it in the expensive black-box objective function $\widehat{f}(x)$. Update the data set $X^{t+1} := X^t \cup \{y\}$. Set t:= $t + 1$.
4: (*Stopping criterion*) Go to step 2, unless a stopping criterion is met.

The procedure begins with an initialization phase in which a sample of points is selected. In this phase, one usually uses experiment design techniques such as Symmetric Latin Hypercube Designs (SLHD) [34], CORNER [19] or Minimax and Maximin distance designs [12].

In Step 2, an approximation of the expensive objective function is built using the set of sampled points; it is the so-called *surrogate model*. Several techniques have been proposed for building this surrogate model, such as polynomial response surface models, the method of moving least-squares, radial basis functions, the kriging methods, artificial neural networks, support vector regression or mixtures of them. A review of these methods is given in [32] and [4]. This approximation of the objective function may have a local character such as polynomial response surfaces which are defined in a region of interest. In contrast, with global approximations such as artificial neural networks, radial basis functions or kriging methods, all the points for which the objective value are known are used to build the surrogate model for the expensive function.

In Step 3, a search infill criterion is designed using the surrogate model. The point selection criterion should balance the information from the unexplored feasible region with the search in promising areas of the design space (according to the surrogate model). From a global optimization view, these issues are respectively known as *exploration* and *exploitation* stages. Depending on the weights of these factors, the search is driven more towards optimization or towards filling of feasible region. This is a sequential scheme in which a single point is introduced in each major iteration.

To enable the incorporation of multiple new samples at each updating cycle, parallel infill strategies have been proposed in recent years to reduce the optimization wall clock time. A taxonomy of these methods can be considered according to two fundamental features: i) the use of single/multiple infill criteria and ii) how they approach the exploration/exploitation stages. In essence, it is a bi-objective optimization problem and this can be approached through Pareto dominance or by weighting to balance exploration and exploitation. A rough taxonomy classifies algorithms into three large groups:

- *Single infill criterion.* These methods address the bi-objective nature of the exploration/exploitation dilemma through scalarization methods in multi-objective optimization such as the weighting methods. This group of methods uses one optimization on one parametrized infill criterion to select a new point and leads to a sequential point generation scheme. Within these methods, a distinction should be made between those that employ an uncertainty-based criterion or a distance-based criterion.
- *Multiple infill criteria addressing with a multi-objetive approach.* The use of multiple infill criteria allows to state a multi-objective optimization problem. These methods select q-points from the Pareto frontier as the new set of points to be sampled.
- *Multiple independent infill criteria.* These techniques employ multiple infill criteria, derived from multiple surrogate models and/or multiple measures, and get q-points from q different criteria.

We will now review the above methods starting with the first group for uncertainty-based measures. We have not considered other types of algorithms, such as parallel surrogate-assisted evolutionary algorithms, as they deviate from the proposed methods in this article and interested readers can consult a more comprehensive review of the literature that is made in [8].

The simplest infill criterion considers the addition of a single point at the current iteration. In [13] *Efficient Global Optimization* (EGO) is proposed, which is one of the most widespread methods. This method is based on kriging basis functions [14], which provide the error in the estimates of the surrogate model. EGO uses the *Expected Improvement* (EI) metric to define the search infill criterion which balances the need for a surrogate objective value (exploitation) together with the uncertainty of the model (exploration).

The first parallelization strategy of EI was based on introducing all the maxima found in the EI by the search algorithm ([28], [29]). In [6] the EI approach is generalized to

a multi-point optimization criterion, the so-called *q-points Expected Improvement* (q-EI). [6] analyse the analytic formula for the case $q = 2$ but solving for this case $q > 2$ requires expensive Monte Carlo simulations of Gaussian vectors. To reduce the corresponding computational burden, two heuristics, *Kriging Believer* (KB) and *Constant Liar* (CL), are introduced to obtain approximately q-EI optimal designs. [21] use the q-EI criterion to handle constraints using a probabilistic approach. [35] propose a new infill criterion named *Pseudo Expected Improvement* (PEI) defined by the multiplication of EI criterion by an influence function of the sampled points. This method selects sequentially the q candidate points by the optimization of the PEI criterion. The resulting algorithm is called EGO-PEI and numerically it is shown that EGO-PEI gains significant improvements compared against CL.

The methods based on a single infill criterion that do not have an uncertainty structure use distance-based refinements. For general surrogate models, [23] introduce the *Metric Stochastic Response Surface* (MSRS) method to choose the candidate point as the best weighted score from two criteria: estimated function value obtained from the response surface model, and minimum distance from previously evaluated points. [26] propose a parallel extension of MSRS to reduce the total elapsed time required by response surface-based global optimization methods. The numerical experiments show that the so-called *Local Metric Stochastic RBF with Restart* (LMSRBF-R) is competitive with the alternative parallel RBF methods.

The *Constrained Optimization using Response Surfaces* (CORS-RBF) algorithm was introduced in [22], [24]. This method chooses the next point by optimizing the surrogate model but restricts the feasible region, requiring the new point to be away from the current points by a certain threshold. This threshold is iterated between a set of values allowing exploration stages for large values and, in other iterations, exploitation stages for small values. This sampling strategy is dense and converges to the global optimum.

[7] proposes a radial basis function method for global optimization (Gutmann-RBF). The infill criterion used is simply to add a new single point and it is based on the "least bumpy" of the interpolation surface. This criterion requires an estimate of the optimal value of the problem. This value changes from iteration to iteration in order to balance the exploration and exploitation stages. [24], [25] parallelize the Gutmann-RBF based on the parametrization of the *Bumpiness Minimization Subproblem* (BMS).

Under a multi-objective perspective, the multiple infill criteria are addressed simultaneously instead of aggregating them into a single criterion. These methods obtain an

approximate Pareto frontier of q-points. In [1] a *multi-objective infill criterion* (MOI) for *model-based optimization* (MBO), named MOI-MBO, is based on kriging models and it takes into account both the diversity and the expected improvement of the proposed points. The numerical experiments show that MOI-MBO outperforms single-step EGO. [10] apply an extension of this algorithm to the hyperparameter tuning problems in machine learning algorithms.

[15] propose a parallel surrogate-based algorithm where simultaneous candidate searches are performed around the Pareto centers, called *Surrogate Optimization with Pareto Selection* (SOP), which considers the trade-off between exploration and exploitation stages as a bi-objective optimization problem where the two objectives are the expensive function value of the point and the minimum distance of the point to previously evaluated points. In SOP, unlike in LMSRBF-R, the new points are randomly obtained from the different centers.

An example of algorithms from the third group is the *Multiple Surrogate Efficient Global Optimization* (MSEGO) algorithm, given in [31]. MSEGO uses q general surrogate models. MSEGO imports error estimates from different instances of kriging models and uses them with all other surrogates, as result, it is obtained a different EI for each surrogate one, and maximizing EIs, it is provided up to q points per cycle. [17] address the aerodynamic shape optimization of transonic wings by using a combination of multiple infill criteria, with each criterion choosing a different sample point. This method does not establish coordination mechanisms between criteria.

3. A surrogate-assisted cooperative optimization framework

The section is structured as follows. First, we define the class of algorithms that can be used to cooperate and find the optimal solution. Each algorithm of this class is named *search agent*. The definition of a search agent is based on bi-objective optimization to derive q-points infill search methods using a single surrogate model. The algorithms of this class (the search agents) are those that may be chosen to work cooperatively. Then, we generalize this initial scheme to allow the possibility of considering simultaneously multiple search agents defined on a set of surrogate models and infill criteria. The cooperative framework establishes an asynchronous coordination mechanism among the search agents. The distributed computing can be used to run each search agent in different computers and a master computer guides the search process.

3.1. Search agent: a type of q-points infill search method based on a single surrogate model

In order to solve the optimization problem described in Section 2, we are interested in algorithms that have a general structure like that shown in Figure 1. The essential characteristic is that the sampling strategy does not generate a single point y but a set $Y^t = \{y_1^*, \ldots, y_q^*\}$ of q points. This allows us to introduce sequentially q points in each major iteration t.

In these algorithms, the number of retained points may be up to k. This is because, in the infill criterion with q points, the number of generated points can become high, thus involving longer time in updating the surrogate model. We propose a criterion to restrict the number of retained points to update the surrogate model, called the *pruning strategy*. This aspect is secondary and has been analyzed in the supplementary material of the paper.

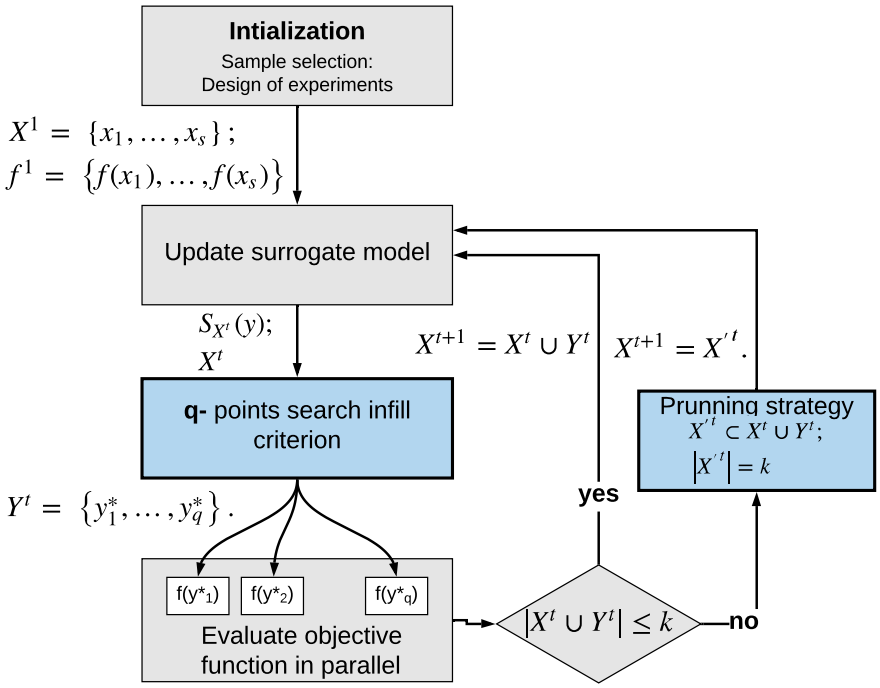

Figure 1: Parallel surrogate-assisted optimization framework

The search agents exhibit the structure of parallel search infill methods shown in Figure 1. A first step in the formalization of a *search agent* is the stating of the sampling problem which is based on two essential criteria:

i) reduction of the level of uncertainty in the region and
ii) the sampled area is close to the most promising regions.

The issues i) and ii) will define respectively the *exploration* and *exploitation* capacities of the resulting parallel search infill method. Mathematically, it has a bi-objective nature and it can be stated by:

$$\underset{x\in\mathcal{D}}{\text{Maximization}}\ \mathbf{F} = (\text{Exploration}(x), \text{Exploitation}(x))^{\top} \tag{2}$$

The formalization of the problem (2) requires the definition of indexes to measure the issues i) and ii). In this paper, we name *exploration measure* to an index that allows us to evaluate the quality of a new solution with respect to the uncertainty of whole search space, on the other hand, we name *exploitation measure* to an index that allows us to estimate the improvement of the objective function in a new point.

Definition 1 (Exploration measure) *An exploration measure of a set $\mathcal{D}$ sampled on the set Z is any function*

$$\begin{aligned} d : \mathcal{P}(\mathcal{D}) \times \mathcal{D} &\mapsto \mathbb{R} \\ (Z, x) &\rightsquigarrow d_Z(x) \end{aligned}$$

where $P(\mathcal{D})$ is the power set of $\mathcal{D}$ and for all $x \in \mathcal{D}$ and for all $Z, Z' \subseteq \mathcal{D}$, the following conditions are satisfied:

i) $d_{Z\cup Z'}(x) \leq d_Z(x)$,
ii) $d_Z(x) \geq 0$,
iii) $d_Z(x) = 0 \Longrightarrow x \in cl(Z)$, *being* $cl(Z)$ *the closure set of* Z

Condition i) indicates that the uncertainty decreases if new points are added to the sampled set, Condition ii) expresses that there is always uncertainty and, when the cardinality of Z is finite, Condition iii) imposes that if there is no uncertainty at a point, then it must have been sampled.

The canonical example is:

$$d_Z(x) = \underset{z\in Z}{\text{minimize}}\ \|z - x\|_2 \tag{3}$$

where $\|\cdot\|_2$ is the Euclidean norm.

The *exploitation measures* estimate the objective function in a point x or alternatively, the improvement in the objective function with respect to a set of previously sampled points. Several measures of this type have been defined in the literature. We have provided a definition of the exploitation measure using the lowest common denominator to all them to be able to accommodate them in a common definition, the essential element is that they use a decreasing transformation of the surrogate model to do so.

Definition 2 (Exploitation measure) *Let $Z \subseteq \mathcal{D}$ and let $S_Z(x)$ be a surrogate model of $f(x)$ building on the set Z, an exploitation measure is a transformation of $S_Z(x)$, i.e. $w_Z(x) := \phi(S_Z(x), x)$, in which the function $\phi(s, x)$ is decreasing on the variable s for all $x \in \mathcal{D}$.*

The canonical example in a deterministic context is:

$$w_Z(x) = -S_Z(x). \tag{4}$$

In this example $\phi(s, x) = -s$ and it holds $\frac{\partial \phi}{\partial s} = -1 < 0$ and it decreases for all x.

Another example is provided by *qualSolve* algorithm (given in [11]) that uses as exploitation measure:

$$w_Z(x) = \exp\left(-\gamma \frac{S_Z(x) - \min\limits_{y \in \mathcal{D}} S_Z(y)}{\max\limits_{y \in \mathcal{D}} S_Z(y) - \min\limits_{y \in \mathcal{D}} S_Z(y)}\right) \tag{5}$$

where $\gamma \geq 0$. The function $w_Z(x)$ assigns higher values to areas with low surrogate function values whereas areas with high surrogate function values are assigned lower values.

In a stochastic context, suppose that the surrogate model $S_Z(x)$ is defined by a stochastic regression process such as the kriging model. In this case, an estimate of the standard error $\widehat{\sigma}_Z(x)$ is obtained and the *probability of improvement* (PI) function ([16], [18]) is defined by:

$$\mathrm{PI}_Z(x) = \mathbb{P}(S_Z(x) \leq T) = \Phi\left(\frac{T - S_Z(x)}{\widehat{\sigma}(x)}\right) \tag{6}$$

where Φ is the normal cumulative distribution function and T is a real number smaller than the current best function value, i.e. $T < \widehat{f}_{min} = \min\limits_{x \in Z} \widehat{f}(x)$. In the stochastic case, to define the exploitation measure the user would choose:

$$w_Z(x) = PI_Z(x). \tag{7}$$

In the same way, *expected improvement*, ([13]), would be chosen as improvement measure, i.e $w_Z(x) = \mathrm{EI}_Z(x)$., where:

$$\mathrm{EI}_Z(x) = \mathbb{E}\left(\max\{\widehat{f}_{min} - S_Z(x), 0\}\right) =$$
$$(\widehat{f}_{min} - S_Z(x))\,\Phi\left(\frac{\widehat{f}_{min} - S_Z(x)}{\widehat{\sigma}_Z(x)}\right) + \widehat{\sigma}_Z(x)\,\phi\left(\frac{\widehat{f}_{min} - S_Z(x)}{\widehat{\sigma}_Z(x)}\right) \tag{8}$$

where Φ and ϕ are the normal cumulative distribution and density functions, respectively. Note that, in the stochastic case, the noisy function $\widehat{f}(x)$ is considered, instead of $f(x)$, in the calculation of $w_Z(z)$. Note that, $\frac{\partial \phi}{\partial s} = \frac{\partial EI}{\partial s} < 0$ is hold.

The search infill criterion has a bi-objective nature: i) a goal defined by means of the exploration measure $d_X(y)$ to weigh the current uncertainty level of the different parts of the region $\mathcal{D}$ and ii) a second goal expressed by the exploitation measure $w_X(y)$ to assess the quality of y to be a minimum of the objective function. The search infill criterion, i.e the sampling problem, is stated as:

Bi-objective search infill criterion

$$\begin{aligned} &\text{Maximize } \mathbf{F} = (d_X(y), w_X(y))^\top \\ &\text{Subject to: } y \in \mathcal{D} \end{aligned} \tag{9}$$

The parallel infill criterion consists of the selection of q points of the Pareto frontier of the bi-objective problem (9). A basic multi-objective optimization method is the ε-constrained method, which maximizes one objective subject to the additional constraint derived from the other objective. One issue with this approach is that it is necessary to preselect which objective to maximize. We consider:

$$\begin{aligned} \text{Maximize} \quad & w_X(y_j) \\ \text{Subject to:} \quad & d_X(y_j) \geq \varepsilon_j \\ & y_j \in \mathcal{D} \end{aligned} \tag{10}$$

where ε_j represents the *worst* value d_X that is allowed to take. It has been shown that if the solution to the problem (10) is unique, then it is an Pareto optimal solution.

Remark. Note that, in opposition to what happens with the exploitation measures, the exploration measures are independent of the function $f(x)$. In each major iteration, the surrogate model $S_X(x)$ is updated on the set of sampled points X, needing to know the value of the objective function $f(x)$ in every point $x \in X$ but the exploration measure $d_X(y)$ can be updated every time in a problem (10) that is solved without the need to evaluate the objective function in these new points. This fact can be taken advantage when the resolution of the q optimization problems (10) is carried out in a sequential manner. It allows the update of the exploitation measure each time that a new point is obtained, let's formalize it. Let $\{y_1^*, \ldots, y_{j-1}^*\}$ be the previous generated points and let $Z_j \subseteq X \cup \{y_1^*, \ldots, y_{j-1}^*\}$, we define the $j-$th point y_j^* to be added to the sampled points as an optimal solution of the problem:

Modified ε-constrained method $\rightarrow$ Hard multi-point search infill criterion

$$\begin{aligned} &\text{Maximize } w_X(y_j) \\ &\text{Subject to: } d_{Z_j}(y_j) \geq \varepsilon_j \\ &y_j \in \mathcal{D} \end{aligned} \tag{11}$$

The selection of values of ε_j is problematic as for many values of ε_j there will be no feasible solution. A way to choose ε_j to prevent this disadvantage is set $\varepsilon_j = \beta_j \Delta_j$ with $\Delta_j = \underset{x \in \mathcal{D}}{\text{Maximize}}\ d_{Z_j}(x)$ and $0 \leq \beta_j < 1$.

The problem (11) may contain many local minima and optimizers such as metaheuristics may be good choices to address it. These methods have been developed intensively for unconstrained optimization [5] and it leads to consider another way to scalarize a bi-objective optimization problem. Asumming that the exploitation measure satisfies $w_X(y) > 0$ for all $y \in \mathcal{D}$, the following sampling problem is stated:

Alternative scalarization approach $\rightarrow$ Soft multi-point search infill criterion

$$\begin{aligned} &\text{Maximize } d_{Z_j}(y_j) w_X(y_j) \\ &\text{Subject to: } y_j \in \mathcal{D} \end{aligned} \tag{12}$$

If we compare both problems (11) and (12), we can observe that the problem (12) attempts to satisfy the constraints of the problem (11) in a soft way. For this reason we refer to the approach (11) as **hard** one and the criterion (12) as **soft** one.

The sequential multi-point infill search criterion is summarized in Algorithm 2.

Algorithm 2 Sequential multi-point search infill criterion

Require: Set of sampled points X

Ensure: New set of sampled points $Y = \{y_1^*, \ldots, y_q^*\}$

function q-POINTS SEARCH INFILL CRITERION(X)
  for all $j = 1, \ldots, q$ **do**
    Update $Z_j \subseteq X \cup \{y_1^*, \ldots, y_{j-1}^*\}$
    Let y_j^* be the optimal solution obtained from (soft or hard) search infill criterion given in (11) or (12).
  end for
end function

Once the above definitions have been introduced, we will formalize the concept of search agent.

Definition 3 (Search agent) *A search agent is any algorithm of the parallel surrogate-assisted optimization framework given in Figure 2 in which the q-points search infill criterion is defined by Algorithm 2.*

3.1.1. Instances of search agents

There are several algorithms described in the literature that fall within this framework. Two noteworthy examples are: EGO-PEI, developed in [35] and the Parallel CORS-RBF, proposed in [25].

The Parallel CORS-RBF is a hard approach in which the exploitation measure $w_X(s)$ is chosen as minus the surrogate model, being it a radial basis function. The search infill problems for Parallel CORS-RBF become:

Parallel CORS-RBF

$$\begin{aligned} \text{Minimize } & S_X(y_j) \\ \text{Subject to: } & d_{Z_j}(y_j) \geq \varepsilon_j \\ & y_j \in \mathcal{D} \end{aligned} \tag{13}$$

where the set Z_j is defined by $Z_j = X \cup \{y_1^*, \ldots, y_{j-1}^*\}$ and the parameters ε_j are derived from the above parameters β_j.

The EGO-PEI algorithm, see [35], is based on kriging models. A simple kriging model can be built as follows:

$$\widehat{f}(x) = \mu + \varepsilon(x), \tag{14}$$

where μ is the mean of the Gaussian process, and $\varepsilon(x)$ is the noise term which is normally distribute with mean zero and variance σ^2. The errors on two points $x, z \in \mathcal{D}$, i.e. $\varepsilon(x)$ and $\varepsilon(z)$, are correlated and the correlation depends on the distance between these points:

$$\text{Corr}\left[\varepsilon(x), \varepsilon(z)\right] = \exp\left(-\sum_{k=1}^{\nu} \theta_k |x_k - z_k|^{p_k}\right)$$

EGO-PEI uses a soft multi-point search infill criterion in which the exploitation measure $w_X(y_j)$ is the *expected improvement* (EI) and the exploration measure is the product of correlations between the sampled points and the new point:

$$d_Z^{\text{EGO-PEI}}(x) = \prod_{z \in Z} \left(1 - \text{Corr}[\varepsilon(x), \varepsilon(z)]\right). \tag{15}$$

Note that, $d_Z^{\text{EGO-PEI}}(x)$ holds the properties i)-iii) of a exploration measure.

Finally, the sampling problem is sated:

EGO-PEI

$$\begin{aligned} &\text{Maximize } d_{Z_j}^{\text{EGO-PEI}}(y_j) EI_X(y_j) \\ &\text{Subject to: } y_j \in \mathcal{D} \end{aligned}$$

where $Z_j = \{y_1^*, \ldots, y_{j-1}^*\}$.

3.2. Coordination of the search agents

In the case of a single search agent, the search infill problems (11) and (12) are solved in a sequential manner. Note that, the sampling of each point requires the solution of an optimization problem and the potentially large number of points generate a substantial computational load. In this section, the use of multiple search agents is analysed to parallel the sampling stage.

We assume the case shown in Figure 2 in which multiple surrogate models and multiple exploitation/exploration measures are being used in the surrogate-assisted optimization. Certain combinations of these options, in conjunction with a soft or hard scheme, provide the available search agents.

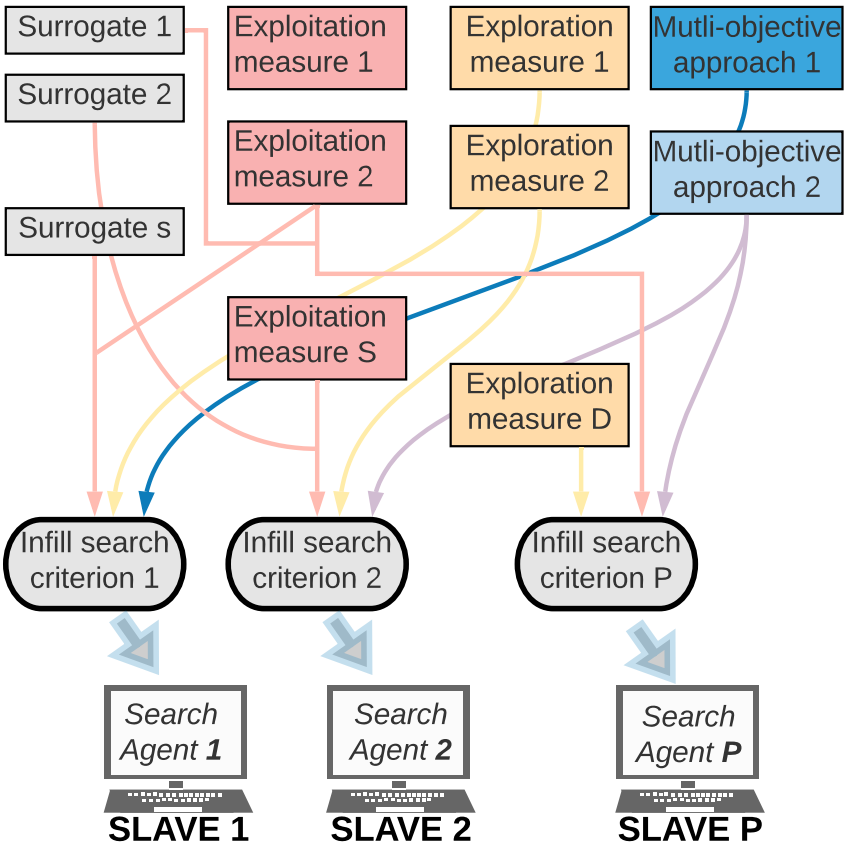

Figure 2: Multiple search infill criteria

We assume a master-slave architecture to carry out the computation, as it is shown in Figure 3. Each slave device runs a search agent using a different infill criterion defined by: i) a surrogate model, ii) an exploitation measure, iii) an exploration measure and iv) a soft/hard approach. The (slave) device i obtains a new point and it is sent to the master, in which it is stored. The master checks the list of previously points generated by the others computers from the last instant that the device i was communicated with the master and sends the new list of sampled points to the device i. The device i adds this set of points to Z_j and updates its exploration measure d_{Z_j} in order to generate the next sampled point. This process is asynchronous and the stopping criteria should be defined for each device i and for the master. For example, the criterion can be based on the maximum number of generated points by each computer i and the master will finish the

whole process when exactly an amount of q-points is generated. The master coordinates the search carried out by the search agents, interchanging the explored zones to avoid the overemphasis in some zones. Each search agent uses itself judgment (defined by its surrogate model, exploitation measure and type of approach) but it shares with the other search agents the points already explored to guide and to coordinate the search.

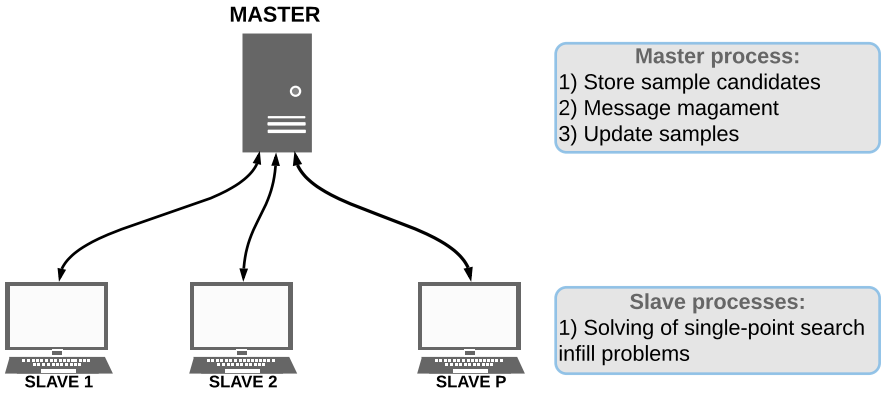

Figure 3: Asynchronous Master-slave cooperative framework

4. Convergence analysis

In this section, we analyze the convergence of the cooperative (asynchronous) algorithm using multiple search agents. First, we study the case where the objective function is deterministic and, after that, we consider the case of noise in the objective function.

4.1. Deterministic case

In this subsection, we will assume that there is no noise in the calculation of the objective function $f(x)$. We denote by $x^* \in \mathcal{D}$ a solution of our optimization problem, i.e., $f(x^*) = \underset{x \in \mathcal{D}}{\text{minimize}}\ f(x)$.

The starting point for our analysis is provided by the following theorem showing that it is sufficient for an algorithm to generate a dense set of points in the feasible region to demonstrate its convergence.

Theorem 1 ([30]) *Let $\mathcal{D}$ be a compact set. Then an algorithm converges to the global minimum of every continuous function on $\mathcal{D}$ if and only if its sequence of iterations is everywhere dense in $\mathcal{D}$.*

Since every dense set has an infinite number of points, we will assume that the search agents do not perform the pruning operation and so retains all generated points (i.e $k = +\infty$). We introduce the superscript s associated with the search agent and the subscript n to refer the nth internal iteration of the algorithm s. In this way, the following notation y_n^s for all n and for $s = 1, \ldots, p$ refers to the nth generated point by the search agent s. We will begin our discussion on the convergence of the cooperative approach assuming that a particular search algorithm s' employs the hard approach (11).

Theorem 2 (Convergence of the cooperative algorithm (case a.)) *Let $f(x)$ be continuous functions on a compact set $\mathcal{D}$. Suppose that there exists a search agent s', satisfying the following assumptions:*

i) the exploration measure $d_{Z_n}^{s'}(y)$ is given by (3),

ii) the rule to select the parameters $\varepsilon_n^{s'}$ is:
$\varepsilon_n^{s'} = \beta_n^{s'}\Delta_n^{s'}$ *where* $\Delta_n^{s'} = \underset{x\in\mathcal{D}}{Maximize}\ d_{Z_n}^{s'}(x)$ *and* $0 \leq \beta_n^{s'} < 1$ *and* $\limsup_{n\to\infty} \beta_n^{s'} > 0$ *and*

iii) the relationship $\{y_1^{s'}, \ldots, y_{n-1}^{s'}\} \subseteq Z_n^{s'}$ is hold,

iv) it generates an infinite sequence $\{y_1^{s'}, y_2^{s'}, \ldots\}$;

then the proposed cooperative algorithm converges to the global minimum.

Proof. We will show that the set $Y^\infty = \bigcup_{s=1}^{p} \bigcup_{n\geq 1} y_n^s$ is dense in $\mathcal{D}$ and, by using Theorem 1, the cooperative algorithm is convergent.

By contradiction, we assume that the set Y^∞ is not dense on $\mathcal{D}$, thus the set $Y^{s'} \cup X^0$, where $Y^{s'} = \bigcup_{n\geq 1} y_n^{s'}$, is not dense. We focus on only the search agent s' in the proof and we will delete the superscript s' associated with this search agent to simplify the notation.

Due to the set $Y \cup X^0$ is not dense on $\mathcal{D}$, there exist a radius $\delta_1 > 0$ and $x_1 \in \mathcal{D}$ such as the open ball $B(x_1, \delta_1)$ satisfies:

$$\left(Y \cup X^0\right) \cap B(x_1, \delta_1) = \{\emptyset\}, \tag{16}$$

and the equation (16) leads to:

$$\Delta_n = \underset{x\in\mathcal{D}}{\text{Maximize}}\ d_{Z_n}(x) \geq \delta_1 \text{ for all } n \geq 1. \tag{17}$$

From the hypothesis ii), $\limsup_{n\to\infty} \beta_n > 0$ and thus there exist a parameter $\delta_2 > 0$ and a subsequence $\{n_v\}$ holding:

$$\beta_{n_v} > \delta_2 \text{ for all } v. \tag{18}$$

Using the equations (17) and (18), it is hold:

$$\varepsilon_{n_v} = \Delta_{n_v}\beta_{n_v} > \delta_1\delta_2 \text{ for all } v. \tag{19}$$

From here, we call $\delta = \delta_1\delta_2 > 0$ and we denote the subsequence n_v simply by v.

If $v < v'$, from the assumption i) and iii), it is hold:

$$\left.\begin{array}{l} y_v \in Z_{v'} \\ d_{Z_{v'}}(y_{v'}) \geq \varepsilon_{v'} > \delta \end{array}\right\} \Rightarrow d(y_v, y_{v'}) > \delta. \tag{20}$$

The equation (20) guarantees that the open balls with centers y_v and $y_{v'}$, and radius $\frac{\delta}{2}$ have an empty intersection, i.e

$$B\left(y_v, \frac{\delta}{2}\right) \cap B\left(y_{v'}, \frac{\delta}{2}\right) = \{\emptyset\}, \text{for all } v < v'. \tag{21}$$

On the other hand, it is hold the following inclusion:

$$\mathcal{D} \subseteq \bigcup_{x\in\mathcal{D}} B\left(x, \frac{\delta}{4}\right) \tag{22}$$

From the compactness of the set $\mathcal{D}$, this covering has a finite sub-covering, i.e. there exist a finite set of centers $z_1, \ldots, z_m$, such as:

$$\mathcal{D} \subset B\left(z_1, \frac{\delta}{4}\right) \cup \cdots \cup B\left(z_m, \frac{\delta}{4}\right) \tag{23}$$

The sequence $\{y_v\}$ contains an infinite number of distinct terms because $d(y_v, y_{v'}) > \delta > 0$ for all $v < v'$. Thus, there exists a ball containing two or more terms of $\{y_v\}$. Let v_1 and v_2 be two different terms belonging to the same ball, thus it is hold:

$$d(y_{v_1}, y_{v_2}) \leq \frac{\delta}{2}. \tag{24}$$

The relationship (24) is a contradiction with the equation (20). This contradiction proves that the set Y^{∞} is dense and the proof is completed.

□

Next, we will analyze the convergence of another configuration of the cooperative algorithms, consisting in a search agent using a soft scheme.

Theorem 3 (Convergence of the cooperative algorithm (case b.)) *Let $f(x)$ be a continuous function on a compact set $\mathcal{D}$. Suppose that there exists a search agent s', satisfying the following assumptions:*

i) the exploration measures $d^{s'}_{Z_n}(y)$ is given by (3),

ii) the exploitation measure verifies

$$0 < m < w_X(x) < M \text{ for all } x \in \mathcal{D} \text{ and for all } X \subset \mathcal{D}, \tag{25}$$

iii) the relationship $\{y^{s'}_1, \ldots, y^{s'}_{n-1}\} \subseteq Z^{s'}_n$ is hold,

iv) it generates an infinite sequence $\{y^{s'}_1, y^{s'}_2, \ldots\}$;

then this cooperative algorithm converges to an global minimum.

Proof. Let's think the case *a* by analogy. We will show that the set $Y^{\infty} = \bigcup_{s=1}^{p} Y^s$, where $Y^s = \bigcup_{n \geq 1} y^s_n$, is dense in $\mathcal{D}$ and, by using Theorem 1, the algorithm is convergent.

By contradiction, we assume that the set Y^{∞} is not dense on $\mathcal{D}$, thus $Y^{s'} \cup X^0$ is not a dense set and it implies that there exist a radius $\delta > 0$ and a point $x_1 \in \mathcal{D}$ satisfying, $B(x', \delta) \subseteq \mathcal{D}$ and

$$B(x', \delta) \cap \left(Y^{s'} \cup X^0\right) = \{\emptyset\}. \tag{26}$$

The proof is exclusively based on the search agent s' and for this reason we will omit the superscript s' to simplify the notation.

Since the sequence $\{y_n\}$ is contained in the compact set $\mathcal{D}$, there is a subsequence y_{n_v} that converges to $y^* \in \mathcal{D}$. To simplify the notation we simply denote the subsequence by $\{y_v\}$. Since it is a convergent sequence, then it is a Cauchy sequence, and let $\varepsilon = \frac{m}{M}\delta > 0$ there is a positive integer v_0 such that:

$$d(y_v, y_{v'}) < \frac{m}{M}\delta, \qquad \text{for all } v, v' \geq v_0. \tag{27}$$

From what we obtain:

$$d_{Z_v}(y_v) < \frac{m}{M}\delta, \qquad \text{for all } v > v_0. \tag{28}$$

Let $v_1 > v_0$ be a positive integer and let t be the main iteration for the search agent s' in which the point y_{v_1} has been obtained, then since the equation (28) and the assumption ii) :

$$d_{Z_{v_1}}(y_{v_1}) w_{X^t}(y_{v_1}) < \frac{m}{M}\delta M = m\delta. \tag{29}$$

Since the equation (26), it follows that $d_{Z_{v_1}}(x') > \delta$ and using the assumption ii),

$$d_{Z_{v_1}}(x') w_{X^t}(x') > \delta m. \tag{30}$$

Finally, from the optimality of y_{v_1}, we get:

$$d_{Z_{v_1}}(y_{v_1}) w_{X^t}(y_{v_1}) \geq d_{Z_{v_1}}(x') w_{X^t}(x') > \delta m, \tag{31}$$

which contradicts the equation (29) and the proof is completed.

□

4.2. Noisy case

We next extend the previous convergence results to the case in which the function $f(x)$ is evaluated with noise. We will assume that:

$$\widehat{f}(x) = f(x) + \xi_x \tag{32}$$

where ξ_x is a random variable with expected value $\mathbb{E}(\xi_x) = 0$ and variance $\mathbb{V}(\xi_x) = \sigma_x^2$. We will assume that $\sigma_x^2 < \sigma^2$, $\forall\, x \in \mathcal{D}$. Let t be the counter of iterations associated with the master processor and let X^t be the sampled points at the $t-th$ iteration. In this case, the following does *not* hold:

$$\min_{x \in X^t} \widehat{f}(x) \to f(x^*) \tag{33}$$

In order to properly monitor the convergence of the cooperative algorithm, instead of working directly with $\widehat{f}(x_i)$, we will work with average values calculated on neighbourhoods of x_i. These neighbourhoods are modified as the algorithm proceeds and achieve ever better estimates of $f(x_i)$. This sequence of neighbourhoods will be defined by the corresponding sequence of (positive) radius $\{\delta_n\}_{n=1}^{\infty}$ satisfying $\lim_{n \to +\infty} \delta_n = 0$. In order to properly update the neighbourhoods as the algorithm operates, a sequence $\{M_n\}_{n=1}^{\infty}$ satisfying $\lim_{n \to +\infty} M_n = +\infty$ will be used. The radius of the neighbourhoods are updated by means of the rule:

$$\text{If } \min_{x_i \in X^t} N_{x_i}^t \geq M_n \Rightarrow n = n + 1, \tag{34}$$

where $N_{x_i}^t = |B(x_i, \delta_n) \cap X^t|$ is the number of elements in the neighbourhood of x_i at iteration t.

The following theoretical result is similar to Theorem 1 but in the context of uncertainty. It is necessary to introduce the following smoothing procedure to eliminate noise and to be able to monitor the convergence of the algorithm:

$$\overline{f}_t(x_i) = \frac{1}{N_{x_i}^t} \sum_{x \in B(x_i, \delta_n) \cap X^t} \widehat{f}(x). \tag{35}$$

Theorem 4 *Suppose we have a cooperative algorithm that generates a sequence of sets X^t and that the limit $\lim_{t \to +\infty} X^t = X^{\infty}$ is dense on the compact set $\mathcal{D}$. Then, if $f(x)$ is continuous on $\mathcal{D}$ the following must hold:*

$$\min_{x \in X^t} \mathbb{E}\left(\overline{f}_t(x)\right) \to f(x^*). \tag{36}$$

Proof. The first assertion to be proven (by contradiction) is that the neighbourhood updating criterion (34) is activated infinitely many times. (i.e. variable $n \to +\infty$ when $t \to +\infty$).

Suppose that there is a given $n \in \mathbb{N}$ that remains fixed from the iteration $t' \in \mathbb{N}$ onwards. This will be because at all iterations $t \geq t'$ there exists a point i_t satisfying:

$$\left|B(x_{i_t}), \delta_n) \cap X^t\right| < M_n. \tag{37}$$

In this way, we construct a sequence $\{x_{i_t}\}_{t \geq t_1}$ contained in the compact set $\mathcal{D}$. Due to the set $\mathcal{D}$ is compact, there exists a convergent subsequence of $\{x_{i_t}\}_{t \geq t_1}$. Without loss of generality we assume that the whole sequence converges to x'. By the convergence of the sequence $\{x_{i_t}\}_{t \geq t_1}$, given $\frac{\delta_n}{2} > 0$ there exists some $t_2 > t_1 \in \mathbb{N}$ satisfying:

$$d(x', x_{i_t}) < \frac{\delta_n}{2}, \quad \forall t > t_2. \tag{38}$$

We will now prove that:

$$B\left(x', \frac{\delta_n}{2}\right) \subset B(x_{i_t}, \delta_n), \quad \forall\, t \geq t_2. \tag{39}$$

For any $x \in B\left(x', \frac{\delta_n}{2}\right)$, the relationship $d(x, x') < \frac{\delta_n}{2}$ is hold and using the triangular inequality:

$$d(x, x_{i_t}) \leq d(x, x') + d(x', x_{i_t}) < \frac{\delta_n}{2} + \frac{\delta_n}{2} = \delta_n, \quad \forall\, t \geq t_2. \tag{40}$$

Thus, it follows that:

$$B\left(x', \frac{\delta_n}{2}\right) \cap X^t \subset B(x_{i_t}, \delta_n) \cap X^t, \quad \forall\, t \geq t_2. \tag{41}$$

leading to:

$$\left|B\left(x', \frac{\delta_n}{2}\right) \cap X^t\right| \leq \left|B(x_{i_t}, \delta_n) \cap X^t\right|, \forall\, t \geq t_2. \tag{42}$$

By taking limits in (42), we obtain:

$$\lim_{t\to+\infty} \left|B(x_{i_t}, \delta_n) \cap X^t\right| \geq \left|B\left(x', \frac{\delta_n}{2}\right) \cap X^\infty\right|. \tag{43}$$

On the other hand, by the density property of X^∞, $\left|B\left(x', \frac{\delta_n}{2}\right) \cap X^\infty\right| = +\infty$. This contradicts the relationship (37).

Let $\mu_n^i = \int_{B(x_i,\delta_n)\cap\mathcal{D}} dV(x)$, then it holds that:

$$\min_{x\in B(x_i,\delta_n)\cap\mathcal{D}} f(x) \leq \frac{1}{\mu_n^i} \int_{B(x_i,\delta_n)\cap\mathcal{D}} f(x)dV(x) \leq \max_{x\in B(x_i,\delta_n)\cap\mathcal{D}} f(x). \tag{44}$$

By taking limits in (44) for $n \to +\infty$, we obtain:

$$\lim_{n\to+\infty} \frac{1}{\mu_n^i} \int_{B(x_i,\delta_n)\cap\mathcal{D}} f(x)dV(x) = f(x_i). \tag{45}$$

On the other hand,

$$\mathbb{E}(\overline{f}_t(x_i)) = \frac{1}{N_{x_i}^t} \sum_{x\in B(x_i,\delta_n)\cap X^t} \mathbb{E}(\widehat{f}(x)) = \frac{1}{N_{x_i}^t} \sum_{x\in B(x_i,\delta_n)\cap X^t} f(x). \tag{46}$$

We observe that what we have on the right side of Equation (46) is a Riemann sum of an integrable function, and consequently:

$$\frac{1}{N_{x_i}^t} \sum_{x\in B(x_i,\delta_n)\cap X^t} f(x) \approx \frac{1}{\mu_n^i} \int_{B(x_i,\delta_n)\cap\mathcal{D}} f(x)dV(x) \tag{47}$$

If limits are taken for $t \to +\infty$ in (46),

$$\lim_{t\to+\infty}\mathbb{E}(\overline{f}_t(x_i)) = \lim_{t\to+\infty}\frac{1}{N_{x_i}^t}\sum_{x\in B(x_i,\delta_n)\cap X^t} f(x)$$
$$= \lim_{n\to+\infty}\frac{1}{\mu_n^i}\int_{B(x_i,\delta_n)\cap\mathcal{D}} f(x)dV(x) = f(x_i) \tag{48}$$

As X^∞ is dense, there exists a sequence $\{\widetilde{x}^{t_j}\}_{j=1}^\infty \to x^*$. Without loss of generality we will assume that $\{t_j\}$ is an increasing sequence and $x^{t_j} \in X^{t_j}$. Therefore, we have:

$$\min_{x\in X^{t_j}}\mathbb{E}\left(\overline{f}_{t_j}(x)\right) \leq \mathbb{E}\left(\overline{f}_{t_j}\left(\widetilde{x}^{t_j}\right)\right). \tag{49}$$

By taking limits when $j \to +\infty$, the right-hand side tends to $f(x^*)$. On the other hand, in the left-hand side, $\{t_j\}$ is a increasing monotone sequence, so it will have the same limit as the original sequence, and then:

$$\lim_{t\to+\infty}\min_{x\in X^t}\mathbb{E}\left(\overline{f}_t(x)\right) \leq f(x^*). \tag{50}$$

The other inequality follows from:

$$\mathbb{E}\left(\overline{f}_t(x)\right) = \frac{1}{N_x^t}\sum_{y\in B(x,\delta_n)\cap X^t} f(y) \geq \frac{N_x^t}{N_x^t}\min_{y\in B(x,\delta_n)\cap X^t}\{f(y)\} \geq f(x^*) \text{ for any } x\in\mathcal{D}. \tag{51}$$

□

Note that the proof of convergence of these algorithms is based on the fact that these algorithms generate a dense set of sampled points independently of the surrogate model and the function f. For this reason, the algorithm will continue to generate dense sets when we use the values $\widehat{f}(x)$ to update the surrogate model. Note also that the use of the $\overline{f}_t(x)$ values is only to monitor the objective function, although they can also be used to update the surrogate model.

5. Numerical experiments

5.1. Parallel surrogate-assisted optimization algorithms

The algorithm proposed in this paper develops a cooperative strategy among a family of parallel surrogate-assisted optimization algorithms (it is defined in Subsection 3.1). Each element of this algorithmic class is called search agent. They share the points sampled to update the surrogate models and their exploration measures enable the coordination of them. It is therefore essential to determine search agents that may be complementary in the search tasks.

In this numerical experiment the selection of the parallel surrogate optimization algorithms has taken into account a highlight found in the work of [2]. [2] compared four meta-modeling techniques, polynomial approximation, kriging, radial basis functions (RBF) and support vector regression (SVR), in order to select the most suitable one to be combined with evolutionary optimization algorithms. They found that the best approach to be used in low dimensionality problems can be kriging or even SVR. In contrast if one tries to optimize a high dimensional problem, then, the best technique is RBF. Two types of surrogate models have been used in the numerical experiments: RBF Thin Plate Spline (TPS) and kriging models.

We have chosen the following radial basis-based algorithm: CORS-RBF, introduced in [24]. Regis and Shoemaker reported good results for CORS-RBF compared to a parallel multistart derivative-based algorithm, a parallel multistart derivative-free trust-region algorithm, and a parallel evolutionary algorithm. The CORS-RBF method has a comparable performance to parallel Gutmann-RBF method. The CORS-RBF algorithm can be a good state-of-the-art exponent in algorithms based on radial basis.

A state-of-the-art parallel EGO algorithm is the *Constant Liar* approach. [35] proposed the EGO-PEI algorithm and conducted a numerical experiment on 6 test problems, showing that the EGO-PEI algorithm performs significantly better that Constant Liar approach. This result is the motivation to choose EGO-PEI as a representative of the parallel algorithms based on kriging models.

As shown in Subsection 3.1.1 CORS-RBF and EGO-PEI algorithms are instances of this family of parallel surrogate optimization algorithms and they have been chosen as search agents, the resulting cooperative algorithm is named CPEI. An asynchronous running of the CPEI algorithm allows the search agents (in this case EGO-PEI and CORS-RBF) not to wait for each other, so CPU time is reduced. As the objective is to evaluate the wall clock (number of main cycles) we have considered a synchronous

Table 1: The Dixon-Szgö test bed

Function	$\mathbf{Z}^*$	ν	**# local min**	**Domain $\mathcal{D}$**
Branin	0.3979	2	3	$[-5, 10] \times [0, 15]$
Goldstein-Price	3.0000	2	4	$[-2, 2]^2$
Hartman 3	-3.8628	3	4	$[0, 1]^3$
Hartman 6	-3.3224	6	6	$[0, 1]^6$
Shekel 5	-10.1532	4	5	$[0, 10]^4$
Shekel 7	-10.4029	4	7	$[0, 10]^4$
Shekel 10	-10.5364	4	10	$[0, 10]^4$

running of the same in which alternately each algorithm samples a point. This means that both algorithms are equally represented in the search process. We are interested in coining a strategy that combines the advantages of CORS-RBF and EGO-PEI regarding the dimensionality of the problem.

5.2. Test problems

We use seven benchmark functions taken from Dixon-Szegö test bed [3] and ten benchmark noiseless functions taken from the Real-Parameter Black-Box Optimization Benchmarking (BBOB) test suite [9], namely the functions F15-F24. The first set of functions has been widely used in the literature and is composed of small size problems (see Table 1 for descriptive statistics of them), these test functions have one global minimum. All test functions of the second suite (see Table 2) have 10 dimensions and are multimodal, their actual search region is given as $[-5, 5]^{10}$. The test problems are analytic so that we do not need to worry about the computational time.

5.3. Experimental setup

We use a thin plate spline RBF interpolation model for CORS-RBF and CPEI. The Kriging models for EGO-PEI and CPEI are built using the DACE (Design and Analysis of Computer Experiments) toolbox [20] with the following setting: Zero order polynomial regression function (`regpoly0`), Gaussian correlation function (`corrgauss`) and the initial guess on θ is set to 1.

We use a SLHD sampling [34] to generate the initial design for all three examined algorithms. The size of the initial experimental was set to $2(\nu + 1)$ points. All algorithms use the same initial experimental design in order to mitigate the effect of the initial

Table 2: The BBOB test suite

Function	**Description**
F15	Rastrigin Function
F16	Weierstrass Function
F17	Schaffers F7 Function
F18	Schaffers F7 Function, moderately ill-conditioned
F19	Composite Griewank-Rosenbrock Function F8F2
F20	Schwefel Function
F21	Gallagher's Gaussian 101-me Peaks Function
F22	Gallagher's Gaussian 21-hi Peaks Function
F23	Katsuura Function
F24	Lunacek bi-Rastrigin Function

experiment on the perfomance. We do twenty trials for each algorithm, for each value of q and for each test problem.

The three algorithms are run with $q = 4$, $q = 8$ and $q = 12$ function evaluations per cycle. Each algorithm requires solving q infill search problems per main iteration. The pseudo expected improvement function is optimized by the particle swarm optimization (PSO) (function `particleswarm` in MATLAB). The PSO algorithm is executed with the following setting:

1. Swarm size: 50
2. Maximum iterations: 100
3. Self adjustment weight (c_1): 1.49
4. Social adjustment weight (c_2): 1.49

The infill search problem for CORS-RBF is a constrained optimization problem and we use the interior point (IP) algorithm (function `fmincon` in MATLAB). The interior point method is executed using the default values and setting:

1. Maximum iterations: 1000
2. Maximum function evaluations: 3000

The experiments were done on a computer with the AMD Ryzen 5 1600X processor of 3.6 GHz and 6 cores for parallelization. The RAM available was 16 GB and the version of MATLAB used was R2017b.

5.4. Experiment 1: Analysis of the performance on a limited computational budget

Experiment 1 corresponds to the practical scenario in which computational resources limit the number of functional evaluations that are possible to perform. In this experiment that number has been set to a maximum of 100 principal iterations. The goal has been to test whether the proposed CPEI algorithm achieves an objective function value significantly better than EGO-PEI and CORS-RBF algorithms.

Tables 3 and 4 show respectively the mean and the standard deviation of the objective function values achieved in the 20 trials for Dixon-Szegö and BBOB test suites. In addition, Mann-Whitney-Wilcoxon signed rank test is used to identify whether or not two algorithms are significantly different. The algorithm in each row is compared to the CPEI method and the symbol "$-$" represents that CPEI cannot be compared to itself. When the calculated p-value is smaller than the significance level $\alpha = 0.05$, we reject the null hypothesis and both algorithms are significantly different with respect to the value of the objective function for a given number of cycles. In this case, we indicate with $h = 1$ that CPEI outperforms the other algorithm and with $h = -1$, otherwise. We use the value $h = 0$ to indicate that the null hypothesis can not be rejected.

The first remarkable fact is that when the number of processors q is increased the three algorithms improve their performance. This result is essential for the optimization of computationally expensive black-box problems, as it allows the use of a large number of processors. For this reason, we pay attention to the results obtained with $q = 12$ processors.

The results confirm the initial assumption and agree with what is emphasized in the literature, that kriging models work well for low-dimensional problems such as Dixon-Szegö test bed (less than 6 dimensions) while the radial basis models are a good alternative for the larger ones such as the BBOB test suite (10 dimensions). It has been observed that EGO-PEI outperforms CORS-RBF in Dixon-Szegö test bed whereas the opposite occurs for BBOB test suite. CPEI has behaved as the best alternative between CORS-RBF and EGO-PEI. If we consider the total of the 17 test problems and for $q = 12$, it is observed that CPEI wins in 4 problems to CORS-RBF (this is $h = 1$), behaves like it in 13 problems (this is $h = 0$) and CPEI is never worse than CORS-RBF (this is $h = -1$). If we analyze the results for the EGO-PEI we obtain that CPEI wins to EGO-PEI in 6 problems ($h = 1$), ties in 10 ($h = 0$) and loses in 1 ($h = -1$).

If we consider all values of q and count which configurations achieve the best results, we obtain that CPEI is the best option for 9 problems, CORS-RBF for 7 problems and

the EGO-PEI for 6 problems. When a tie occurs, all tied algorithms are considered the best option.

If we observe which algorithm has been the worst for $q = 12$, we observe that CPEI is only in two problems of the total of 17 problems: F21 and Shekel7. The conclusion drawn from Experiment 1 is that the cooperation strategy improves the robustness of the resulting algorithm, obtaining that for the analyzed case and for the tested algorithms, CPEI is a good choice as much for problems of small dimensions as others with something bigger.

Another aspect observed through this experiment is that CORS-RBF and EGO-PEI sequentially sample the q points while CPEI can parallelize the generation of these points from two to two.

Table 3: The objective function value (mean and standard deviation for 20 trials) and the Wilcoxon signed rank test in Dixon-Szegö test bed

Function	Algorithm	$q = 4$				$q = 8$				$q = 12$			
		Avg.	Std.	p-value	h	Avg.	Std.	p-value	h	Avg.	Std.	p-value	h
Branin	CPEI	0.40	0.00	-	-	0.40	0.00	-	-	0.40	0.00	-	-
	EGO-PEI	0.40	0.00	0.4330	0	0.40	0.00	0.3507	0	0.40	0.00	0.6542	0
	CORS-RBF	0.40	0.00	0.6813	0	0.40	0.00	0.0400	1	0.40	0.00	0.0731	0
Goldprice	CPEI	3.01	0.01	-	-	3.01	0.01	-	-	3.01	0.01	-	-
	EGO-PEI	3.01	0.01	0.9108	0	3.01	0.01	0.5257	0	3.01	0.01	0.8519	0
	CORS-RBF	3.01	0.01	0.4330	0	3.01	0.01	0.0859	0	3.01	0.01	0.0111	1
Shekel5	CPEI	-6.56	3.30	-	-	-7.30	2.92	-	-	-9.73	1.66	-	-
	EGO-PEI	-8.59	2.55	0.0152	-1	-9.59	1.54	0.0045	-1	-9.62	1.55	0.3703	0
	CORS-RBF	-9.66	1.15	0.0169	-1	-10.12	0.03	0.0004	-1	-10.12	0.03	0.0522	0
Shekel7	CPEI	-9.34	2.54	-	-	-9.98	1.70	-	-	-10.09	1.17	-	-
	EGO-PEI	-10.10	1.17	0.2627	0	-9.84	1.62	0.9405	0	-10.10	1.17	0.4115	0
	CORS-RBF	-9.41	1.99	0.3135	0	-10.36	0.04	0.3703	0	-10.36	0.04	0.6813	0
Shekel10	CPEI	-9.32	2.86	-	-	-10.12	1.71	-	-	-10.50	0.03	-	-
	EGO-PEI	-10.49	0.03	0.2322	0	-10.49	0.03	0.8519	0	-10.49	0.03	0.1913	0
	CORS-RBF	-8.80	2.72	0.3135	0	-9.81	1.82	0.0731	0	-10.48	0.06	0.7938	0
Hartman3	CPEI	-3.85	0.01	-	-	-3.85	0.01	-	-	-3.85	0.01	-	-
	EGO-PEI	-3.85	0.01	0.9702	0	-3.85	0.01	0.9702	0	-3.85	0.01	0.4781	0
	CORS-RBF	-3.84	0.01	0.0228	1	-3.84	0.01	0.0080	1	-3.84	0.01	0.0137	1
Hartman6	CPEI	-3.31	0.01	-	-	-3.31	0.01	-	-	-3.31	0.01	-	-
	EGO-PEI	-3.26	0.05	0.0008	1	-3.27	0.05	0.0017	1	-3.28	0.05	0.0090	1
	CORS-RBF	-3.29	0.02	0.0007	1	-3.30	0.01	0.0064	1	-3.30	0.01	0.0022	1

Table 4: The objective function value (mean and standard deviation for 20 trials) and the Wilcoxon signed rank test in BBOB test suite

Function	Algorithm	$q = 4$				$q = 8$				$q = 12$			
		Avg.	Std.	p-value	h	Avg.	Std.	p-value	h	Avg.	Std.	p-value	h
F15	CPEI	15.48	14.01	-	-	5.80	13.27	-	-	3.88	8.34	-	-
	EGO-PEI	51.15	15.46	0.0001	1	53.84	14.84	0.0001	1	52.17	15.44	0.0001	1
	CORS-RBF	26.96	21.37	0.0859	0	0.03	14.47	0.1790	0	2.65	14.53	0.7089	0
F16	CPEI	-247.04	3.77	-	-	-249.21	4.25	-	-	-250.55	3.41	-	-
	EGO-PEI	-248.21	5.12	0.5755	0	-253.11	4.65	0.0124	-1	-254.80	3.88	0.0040	-1
	CORS-RBF	-247.27	4.53	0.7089	0	-247.91	5.28	0.6542	0	-247.85	4.75	0.0276	1
F17	CPEI	-36.93	0.79	-	-	-37.59	0.63	-	-	-38.04	0.85	-	-
	EGO-PEI	-33.23	1.64	0.0001	1	-33.72	1.10	0.0001	1	-33.31	1.11	0.0001	1
	CORS-RBF	-36.62	0.90	0.1560	0	-37.31	1.04	0.2322	0	-38.18	0.51	0.9405	0
F18	CPEI	-31.55	3.19	-	-	-32.47	3.23	-	-	-35.57	1.25	-	-
	EGO-PEI	-19.35	3.35	0.0001	1	-20.47	4.81	0.0001	1	-19.77	4.36	0.0001	1
	CORS-RBF	-29.09	3.15	0.0251	1	-30.73	3.87	0.1005	0	-34.41	4.64	0.7652	0
F19	CPEI	45.44	0.96	-	-	44.99	1.16	-	-	44.14	0.72	-	-
	EGO-PEI	46.30	1.25	0.0333	1	46.26	0.94	0.0017	1	46.36	1.29	0.0002	1
	CORS-RBF	45.80	1.39	0.3135	0	45.12	0.94	0.5503	0	44.65	0.91	0.0731	0
F20	CPEI	186.03	0.31	-	-	185.98	0.25	-	-	185.85	0.29	-	-
	EGO-PEI	187.21	1.12	0.0003	1	186.51	0.39	0.0003	1	186.47	0.32	0.0001	1
	CORS-RBF	185.95	0.34	0.2790	0	185.91	0.28	0.3507	0	185.84	0.18	0.7089	0
F21	CPEI	313.62	2.38	-	-	312.96	1.99	-	-	312.60	1.94	-	-
	EGO-PEI	313.42	2.15	0.5257	0	312.43	1.35	0.8813	0	311.94	0.83	0.9702	0
	CORS-RBF	313.76	2.62	0.6274	0	312.72	1.64	0.9702	0	312.41	1.65	0.6012	0
F22	CPEI	50.14	7.42	-	-	45.91	1.90	-	-	45.83	2.01	-	-
	EGO-PEI	47.46	3.48	0.6542	0	45.88	2.01	0.3507	0	46.02	2.23	0.8813	0
	CORS-RBF	47.87	5.09	0.2471	0	46.37	2.94	0.4115	0	46.47	2.86	0.4781	0
F23	CPEI	213.39	0.61	-	-	213.34	0.54	-	-	212.68	0.63	-	-
	EGO-PEI	213.54	0.66	0.6542	0	213.04	0.55	0.2180	0	212.97	0.43	0.0859	0
	CORS-RBF	213.29	0.81	0.7652	0	212.87	0.63	0.0333	-1	212.77	0.34	0.9405	0
F24	CPEI	127.30	10.90	-	-	118.79	11.77	-	-	116.64	7.61	-	-
	EGO-PEI	120.27	8.92	0.0251	-1	116.32	12.25	0.4553	0	115.49	11.74	0.4553	0
	CORS-RBF	133.60	15.76	0.0930	0	126.17	19.78	0.3317	0	117.97	11.33	0.5755	0

5.5. Results and discussions in Experiment 2

Experiment 1 compares the algorithms in terms of the objective function value for a given number of cycles, on the contrary, in Experiment 2 the number of iterations has not been fixed and what has been set is the relative error to be reached. The comparison is carried out in terms of number of cycles. Due to the great diversity of the dimensions in the test problems, we have considered several stopping criteria. For Dixon-Szegö problems (with dimensions less than 10) we have used the same criteria as the one used in [11], that is, the procedure stops if this relative error is smaller than 1% or a maximum of 100 major iterations has been reached. The difficulty in the resolution of the BBOB test suite is very diverse and has led us to apply several relative errors. We have used a relative error of the 150% for the problems F15 and F24, of 20% for F18, of 5% for F16, of 2% for F21 and F23 and for the rest of the problems we have considered a relative error of 15%. The algorithm also stops if a maximum of 100 major iterations is reached. In order to evaluate the effect of the choice of the relative errors we have shown the progress of the algorithms on BBOB test suite in Figures 4-5. It is observed that the algorithms maintain their ranking throughout the optimization process. Therefore, the choice of a given error affects the rate of convergence but not the ranking between them.

The compared metrics are the number of cycles needed to find the optimum of the benchmark problems and the success convergence rate to achieve a near optimal solution. Both tests are solved 20 times with each algorithm and each function and it is reported the mean and standard deviation of the the number of cycles to achieve the given relative error. In addition, the convergence success rate is calculated, i.e. the quotient between the number of times that a solution within the given relative error is reached and the number of attempts (in our case twenty times), expressed as a percentage.

Tables 5 and 6 show the results for Dixon-Szegö and BBOB test suites, respectively. In addition, Mann-Whitney-Wilcoxon rank sum test is used to identify whether or not two algorithms are significantly different with respect to the number of cycles. For any algorithm $\mathcal{A}$, we write $h = 0$ if $\mathcal{A}$ and CPEI are not significantly different, we write $h = 1$ if CPEI is significantly better than $\mathcal{A}$ at the 5% level of significance, and we write $h = -1$, otherwise. The difference between this test and the one used in the Experiment 1 is that in this case we consider independent samples instead of paired samples.

When doing more functions evaluations per iteration, it should be expected that the algorithm improves the objective function value in less wall-clock time. In literature, see reference [35], it has been observed that when more points are included in one cycle,

the average efficiency of the updating point goes down, causing that the best results are not obtained with values greater than q. This phenomenon may be induced because multi-point search infill problems [as the ones defined in Eqs. (11) and (12)] are highly multi-modal and their optimization becomes increasingly difficult[1]. A highlight support by the numerical results is that this phenomenon is not present in our implementation of the algorithms, and the best results are obtained when $q = 12$. This is a relevant feature of the algorithms that allows to take advantage of higher number of processors and it is the main motivation of this work.

Now we will analyze the best configuration of the algorithms, obtained when $q = 12$. If we compare CPEI with CORS-RBF, we observe that CPEI is significantly better than CORS-RBF in 6 of the 17 problems, in the other 11 problems their differences are not significant. If we compare CPEI with EGO-PEI, we observe that in 4 problems CPEI is significantly better than EGO-PEI, in 3 problems CPEI is significantly worst than EGO-PEI and in the rest of problems (a total of 10) there are no significant differences among them. The advantage of CPEI over EGO-PEI is seen in the convergence rate. Using a significance level of 5% and the McNemar test on paired proportions we have found that CPEI has a significantly better convergence rate than EGO-PEI on problems F15, F17, F18, F19 and Hartman6. There are no significant differences for the rest of the problems.

To summarize, if we analyze which is the best algorithm to solve a given problem, obviating whether or not this improvement is statistically significant, we observe that CPEI is the best at 9 problems, EGO-PEI at 6 and CORS-RBF at 2. This shows that cooperation between CORS-RBF and EGO-PEI improves the baseline algorithms.

[1] [35] show that the performance of the EGO-PEI is sensitive to the quality of the optimizer of the PEI function, defined in (12).

Table 5: The number of cycles (mean and standard deviation for 20 trials), the Wilcoxon rank sum test and the convergence percentage needed to find a solution within 1% in Dixon-Szegö test bed

Function	Algorithm	$q = 4$					$q = 8$					$q = 12$				
		Avg.	Std.	p-value	h	%	Avg.	Std.	p-value	h	%	Avg.	Std.	p-value	h	%
Branin	CPEI	7.20	1.28	-	-	100.0	4.55	0.83	-	-	100.0	3.90	0.79	-	-	100.0
	EGO-PEI	6.25	1.21	0.0193	-1	100.0	3.95	0.60	0.0129	-1	100.0	3.15	0.37	0.0004	-1	100.0
	CORS-RBF	27.15	9.81	0.0000	1	100.0	14.60	4.21	0.0000	1	100.0	9.45	3.36	0.0000	1	100.0
Goldprice	CPEI	22.20	2.12	-	-	100.0	12.20	1.44	-	-	100.0	8.95	0.69	-	-	100.0
	EGO-PEI	20.05	1.82	0.0028	-1	100.0	11.40	0.94	0.0102	-1	100.0	8.50	0.61	0.0374	-1	100.0
	CORS-RBF	34.55	13.73	0.0001	1	100.0	24.15	10.18	0.0000	1	100.0	17.85	6.40	0.0000	1	100.0
Shekel5	CPEI	40.71	31.61	-	-	35.0	50.89	33.12	-	-	45.0	30.16	20.38	-	-	95.0
	EGO-PEI	43.86	25.48	0.7652	0	70.0	35.83	18.25	0.2913	0	90.0	17.83	9.81	0.0176	-1	90.0
	CORS-RBF	56.67	26.78	0.1875	0	60.0	57.55	21.30	0.6541	0	100.0	43.75	16.57	0.0144	1	100.0
Shekel7	CPEI	33.18	24.21	-	-	85.0	25.95	18.85	-	-	95.0	25.63	23.63	-	-	95.0
	EGO-PEI	31.53	21.13	0.7997	0	95.0	21.89	14.39	0.3609	0	90.0	20.79	18.13	0.2298	0	95.0
	CORS-RBF	52.42	16.76	0.0115	1	60.0	36.61	26.56	0.1320	0	90.0	30.30	20.17	0.1593	0	100.0
Shekel10	CPEI	34.00	24.68	-	-	85.0	22.89	14.26	-	-	95.0	16.65	14.51	-	-	100.0
	EGO-PEI	32.00	16.58	0.9513	0	100.0	17.95	9.52	0.3503	0	100.0	13.60	5.92	0.1332	0	100.0
	CORS-RBF	46.00	27.10	0.1381	0	55.0	35.86	18.84	0.0286	1	70.0	33.89	22.71	0.0011	1	95.0
Hartman3	CPEI	4.40	1.19	-	-	100.0	3.55	1.00	-	-	100.0	3.00	0.65	-	-	100.0
	EGO-PEI	4.40	1.27	1.0000	0	100.0	3.50	0.83	0.9647	0	100.0	3.20	0.62	0.3301	0	100.0
	CORS-RBF	16.20	12.45	0.0003	1	100.0	7.60	5.29	0.0015	1	100.0	6.85	3.70	0.0001	1	100.0
Hartman6	CPEI	38.00	31.38	-	-	100.0	17.40	13.38	-	-	100.0	13.00	9.39	-	-	100.0
	EGO-PEI	7.50	2.01	0.0037	-1	50.0	20.17	31.69	0.2030	0	60.0	19.20	31.18	0.6752	0	75.0
	CORS-RBF	27.71	10.86	0.9878	0	85.0	20.00	11.77	0.2028	0	100.0	16.70	10.94	0.0780	0	100.0

Table 6: The number of cycles (mean and standard deviation for 20 trials), the Wilcoxon rank sum test and the convergence percentage in BBOB test suite

Function	Algorithm	$q = 4$					$q = 8$					$q = 12$				
		Avg.	Std.	p-value	h	%	Avg.	Std.	p-value	h	%	Avg.	Std.	p-value	h	%
F15	CPEI	43.00	15.45	-	-	65.0	46.11	22.62	-	-	90.0	35.00	20.76	-	-	100.0
	EGO-PEI	64.00	0.00	0.2857	0	5.0	71.00	0.00	0.4211	0	5.0	41.00	0.00	0.8039	0	5.0
	CORS-RBF	35.29	19.86	0.3618	0	35.0	40.12	22.60	0.1655	0	85.0	37.82	26.10	0.9029	0	85.0
F16	CPEI	71.23	27.39	-	-	65.0	58.00	21.73	-	-	70.0	40.80	25.17	-	-	75.0
	EGO-PEI	51.23	26.69	0.0647	0	65.0	29.35	13.72	0.0006	-1	85.0	28.37	19.77	0.1309	0	95.0
	CORS-RBF	64.55	24.70	0.4005	0	55.0	50.93	24.12	0.4761	0	70.0	48.27	22.55	0.3112	0	55.0
F17	CPEI	14.75	7.22	-	-	100.0	8.30	3.33	-	-	100.0	11.30	12.99	-	-	100.0
	EGO-PEI	66.38	27.53	0.0001	1	65.0	48.67	21.60	0.0000	1	75.0	59.18	24.45	0.0000	1	55.0
	CORS-RBF	21.85	14.28	0.1710	0	100.0	12.95	5.18	0.0014	1	100.0	8.65	4.15	0.7445	0	100.0
F18	CPEI	74.47	17.51	-	-	75.0	58.80	21.63	-	-	75.0	44.10	21.75	-	-	100.0
	EGO-PEI	-	-	-	1	0.0	-	-	-	1	0.0	-	-	-	1	0.0
	CORS-RBF	78.60	15.34	0.7266	0	25.0	58.90	23.45	0.9336	0	50.0	44.61	24.08	0.8951	0	90.0
F19	CPEI	41.17	22.80	-	-	90.0	26.00	22.47	-	-	95.0	11.60	8.95	-	-	100.0
	EGO-PEI	43.36	32.33	0.9641	0	55.0	53.09	32.42	0.0238	1	55.0	40.83	22.32	0.0002	1	60.0
	CORS-RBF	41.08	25.58	0.9521	0	65.0	21.74	15.10	0.8723	0	95.0	13.35	6.98	0.3499	0	100.0
F20	CPEI	11.00	5.15	-	-	100.0	7.75	3.81	-	-	100.0	6.25	2.94	-	-	100.0
	EGO-PEI	24.30	18.82	0.0027	1	100.0	13.50	11.74	0.0973	0	100.0	14.75	6.58	0.0001	1	100.0
	CORS-RBF	20.00	7.17	0.0002	1	100.0	13.40	5.37	0.0011	1	100.0	7.90	3.08	0.0775	0	100.0
F21	CPEI	30.68	28.39	-	-	95.0	20.55	18.60	-	-	100.0	12.00	10.93	-	-	95.0
	EGO-PEI	22.26	20.44	0.6503	0	95.0	26.50	28.62	0.8709	0	100.0	16.35	17.97	0.5139	0	100.0
	CORS-RBF	40.39	23.02	0.0499	1	90.0	32.50	23.80	0.0302	1	100.0	25.25	14.28	0.0005	1	100.0
F22	CPEI	22.21	22.24	-	-	70.0	24.21	29.38	-	-	95.0	20.79	27.31	-	-	95.0
	EGO-PEI	35.75	32.42	0.1450	0	80.0	24.26	19.73	0.5191	0	95.0	23.37	27.57	0.1431	0	95.0
	CORS-RBF	28.13	7.87	0.0027	1	80.0	20.61	13.79	0.0460	-1	90.0	16.78	19.18	0.1428	0	90.0
F23	CPEI	29.05	23.51	-	-	100.0	16.00	17.18	-	-	100.0	8.00	9.78	-	-	100.0
	EGO-PEI	23.32	17.07	0.6528	0	95.0	10.45	10.02	0.2494	0	100.0	9.35	9.06	0.5684	0	100.0
	CORS-RBF	28.61	23.19	0.9650	0	90.0	10.45	9.33	0.3934	0	100.0	15.20	18.57	0.2488	0	100.0
F24	CPEI	36.40	23.90	-	-	25.0	64.30	20.89	-	-	50.0	43.09	27.73	-	-	55.0
	EGO-PEI	67.00	20.45	0.0530	0	35.0	53.60	26.12	0.5202	0	50.0	44.43	23.47	0.7425	0	70.0
	CORS-RBF	48.00	36.53	0.7302	0	20.0	41.67	22.87	0.1861	0	30.0	44.11	20.98	0.9696	0	45.0

6. Conclusions and future research

Motivation. Parallel surrogate-assisted optimization allows to reduce the wall-clock time needed for optimizing expensive objective functions. The performance of these techniques varies from one problem to another depending on their characteristics, such as their dimensionality, the existence of noise, and so on. As suggested by the "No Free Lunch Theorem" Wolpert and Macready [33] no algorithm is best for solving all types of optimization problems. An issue that is presently undergoing intensive research is to combine the best features from different algorithms to come up with a more effective algorithm in a wider domain of applications.

Contributions. In this paper, we introduce a framework to develop surrogate-assisted cooperative optimization strategies to address computationally expensive black-box problems. The essential methodological contributions of this article are: i) a formal definition of a class of parallel optimization algorithms based on a bi-objective approach, each algorithm of this class is named *search agent*; ii) a coordination mechanism among search agents; iii) a theoretical analysis on their convergence and iv) to our knowledge, it is the first time that the problem of restricting the number of points in the updating of the surrogate model is analysed. This issue is discussed in the section *Prunning strategy* in supplementary material.

Within this framework, we have focused on the combination of a parallel RBF-based algorithm, CORS-RBF, and a parallel kriging-based method, EGO-PEI. The former algorithm is named CPEI, and it has been designed with the goal of obtaining a more robust convergence algorithm in a wider class of optimization problems.

Numerical results. In the numerical experiments, we compared CPEI with CORS-RBF and EGO-PEI on the Dixon-Szegö test bed and the F15-F24 benchmark functions taken from the BBOB test suite. The results show twofold highlight: i) the reduction of wall-clock time with respect to the number of processors q and ii) the cooperative strategy improves the convergence properties of the baseline algorithms in terms of relative error achieved and successful convergence rate.

Future research. This is a generic algorithmic class and allows multiple instances not analyzed in this work. A notable example are *homogeneous cooperative strategies*, i.e. using the same surrogate-assisted optimization method with different targets. The simplest example is to execute different instances of an algorithm, for example CORS-RBF, but using different Latin hypercube designs and/or different hyperparameters for the RBF functions. FRENTE A LA REINICIALIZACION

ACKNOWLEDGMENTS

This research was supported by *Ministerio de Economía, Industria y Competitividad*-FEDER EU grant with number TRA2016-76914-C3-2-P.

References

[1] Bischl, B., S. Wessing, N. Bauer, K. Friedrichs, and C. Weihs (2014). MOI-MBO: Multiobjective Infill for Parallel Model-Based Optimization. In P. M. Pardalos, M. G. C. Resende, C. Vogiatzis, and J. L. Walteros (Eds.), *Learning and Intelligent Optimization: 8th International Conference, Lion 8, Gainesville, FL, USA, February 16-21, 2014. Revised Selected Papers*, pp. 173–186. Cham: Springer International Publishing.

[2] Díaz-Manríquez, A., G. Toscano-Pulido, and W. Gómez-Flores (2011). On the selection of surrogate models in evolutionary optimization algorithms. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pp. 2155–2162.

[3] Dixon, L. C. W. and G. Szegö (1978). The global optimization problem: an introduction. *Towards Global Optimisation 2*, 1–15.

[4] Forrester, A. I. J. and A. J. Keane (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences 45*(1-3), 50–79.

[5] García-Ródenas, R., L. J. Linares, and J. A. López-Gómez (2019). A Memetic Chaotic Gravitational Search Algorithm for unconstrained global optimization problems. *Applied Soft Computing Journal 79*, 14–29.

[6] Ginsbourger, D., R. Riche, and L. Carraro (2010). Kriging Is Well-Suited to Parallelize Optimization. In Y. Tenne and C.-K. Goh (Eds.), *Computational Intelligence in Expensive Optimization Problems*, Chapter 6, pp. 131–162. Berlin, Heidelberg: Springer Berlin Heidelberg.

[7] Gutmann, H.-M. (2001). A Radial Basis Function Method for Global Optimization. *Journal of Global Optimization 19*(3), 201–227.

[8] Haftka, R. T., D. Villanueva, and A. Chaudhuri (2016). Parallel surrogate-assisted global optimization with expensive functions – a survey. *Structural and Multidisciplinary Optimization 54*(1), 3–13.

[9] Hansen, N., S. Finck, R. Ros, and A. Auger (2009). Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Technical Report RR-6829, INRIA.

[10] Horn, D. and B. Bischl (2016). Multi-objective parameter configuration of machine learning algorithms using model-based optimization. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8.

[11] Jakobsson, S., M. Patriksson, J. Rudholm, and A. Wojciechowski (2010). A method for simulation based optimization using radial basis functions. *Optimization and Engineering 11*(4), 501–532.

[12] Johnson, M. E., L. M. Moore, and D. Ylvisaker (1990). Minimax and maximin distance designs. *Journal of Statistical PLanning and Inference 26*, 131–148.

[13] Jones, D. R., M. Schonlau, and W. J. Welch (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization 13*(4), 455–492.

[14] Krige, D. G. (1951). A statistical approach to some basic mine valuation problems on the Witwatersrand. *J. of the Chem., Metal. and Mining Soc. of South Africa 52*(6), 119–139.

[15] Krityakierne, T., T. Akhtar, and C. A. Shoemaker (2016). SOP: parallel surrogate global optimization with Pareto center selection for computationally expensive single objective problems. *Journal of Global Optimization 66*(3), 417–437.

[16] Kushner, H. J. (1964). A New Method of Locating the Maximum of an Arbitrary Multipeak Curve in the Presence of Noise. *Journal of Basic Engineering 86*, 97–106.

[17] Liu, J., W.-P. Song, Z.-H. Han, and Y. Zhang (2017). Efficient aerodynamic shape optimization of transonic wings using a parallel infilling strategy and surrogate models. *Structural and Multidisciplinary Optimization 55*(3), 925–943.

[18] Mockus, J. (1994). Application of Bayesian Approach to Numerical Methods of Global and Stochastic Optimization. *Journal of Global Optimization 4*, 347–365.

[19] Müller, J. (2012). User guide for Modularized Surrogate Model Toolbox.

[20] N. Lophaven, S., H. Nielsen, and J. Sndergaard (2002). DACE - A MATLAB Kriging Toolbox.

[21] Parr, J. M., A. J. Keane, A. I. J. Forrester, and C. M. E. Holden (2012). Infill sampling criteria for surrogate-based optimization with constraint handling. *Engineering Optimization 44*(10), 1147–1166.

[22] Regis, R. G. and C. A. Shoemaker (2005). Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization 31*(1), 153–171.

[23] Regis, R. G. and C. A. Shoemaker (2007a). A stochastic radial basis function method for the global optimization of expensive functions. *INFORMS Journal on Computing 19*(4), 497–509.

[24] Regis, R. G. and C. A. Shoemaker (2007b). Improved strategies for radial basis function methods for global optimization. *Journal of Global Optimization 37*(1), 113–135.

[25] Regis, R. G. and C. A. Shoemaker (2007c). Parallel radial basis function methods for the global optimization of expensive functions. *European Journal of Operational Research 182*(2), 514–535.

[26] Regis, R. G. and C. A. Shoemaker (2009). Parallel stochastic global optimization using radial basis functions. *INFORMS Journal on Computing 21*(3), 411–426.

[27] Rezaveisi, M., K. Sepehrnoori, and R. T. Johns (2014). Tie-simplex-based phase-behavior modeling in an IMPEC reservoir simulator. *SPE Journal 19*(2), 327–339.

[28] Schonlau, M. (1997). *Computer experiments and global optimization*. Ph. D. thesis, University of Waterloo, Waterloo, Ontario, Canada; 1997.

[29] Sóbester, A., S. Leary, and A. Keane (2004). A parallel updating scheme for approximating and optimizing high fidelity computer simulations. *Structural and Multidisciplinary Optimization 27*(5), 371–383.

[30] Torn, A. and A. Zilinskas (1989). *Global Optimization,*, Volume Vol. 350. Lecture Notes in Computer Science, Springer-Verlag, Berlin.

[31] Viana, F. A., R. T. Haftka, and L. T. Watson (2013). Efficient Global Optimization Algorithm Assisted by Multiple Surrogate Techniques. *Journal of Global Optimization 56*(2), 669–689.

[32] Vu, K. K., C. D'Ambrosio, Y. Hamadi, and L. Liberti (2017, 4). Surrogate-based methods for black-box optimization. *International Transactions in Operational Research 24*(3), 393–424.

[33] Wolpert, D. H. and W. G. Macready (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation 1*(1), 67–82.

[34] Ye, K. Q., W. Li, and A. Sudjianto (2000). Algorithmic construction of optimal symmetric Latin hypercube designs. *Journal of Statistical Planning and Inference 90*, 145–159.

[35] Zhan, D., J. Qian, and Y. Cheng (2017). Pseudo expected improvement criterion for parallel EGO algorithm. *Journal of Global Optimization 68*(3), 641–662.

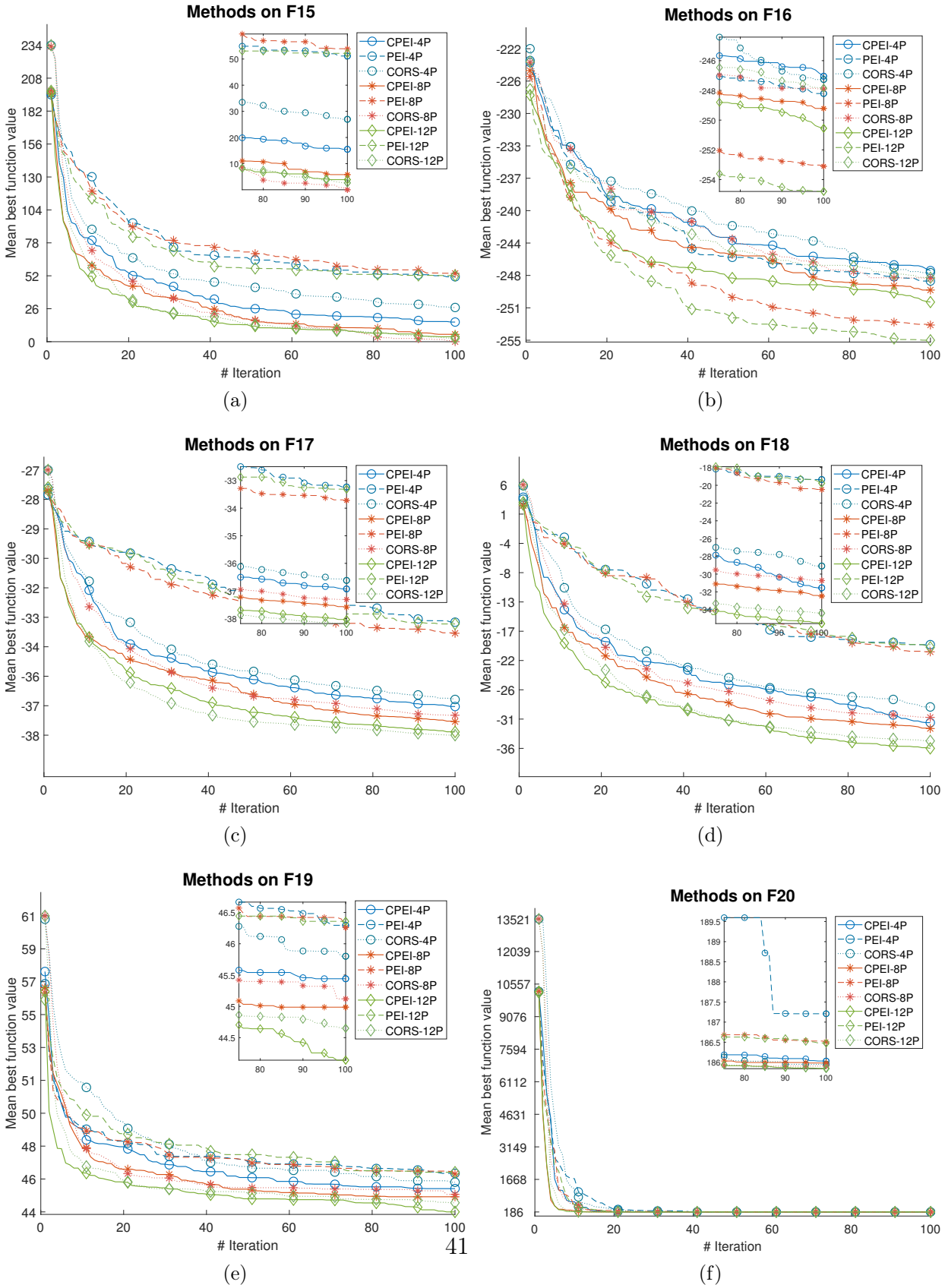

Figure 4: Best objective function value founded average over twenty trials versus major iteration. Global optimization methods on the test functions F15-F20 (taken from BBOB test suite)

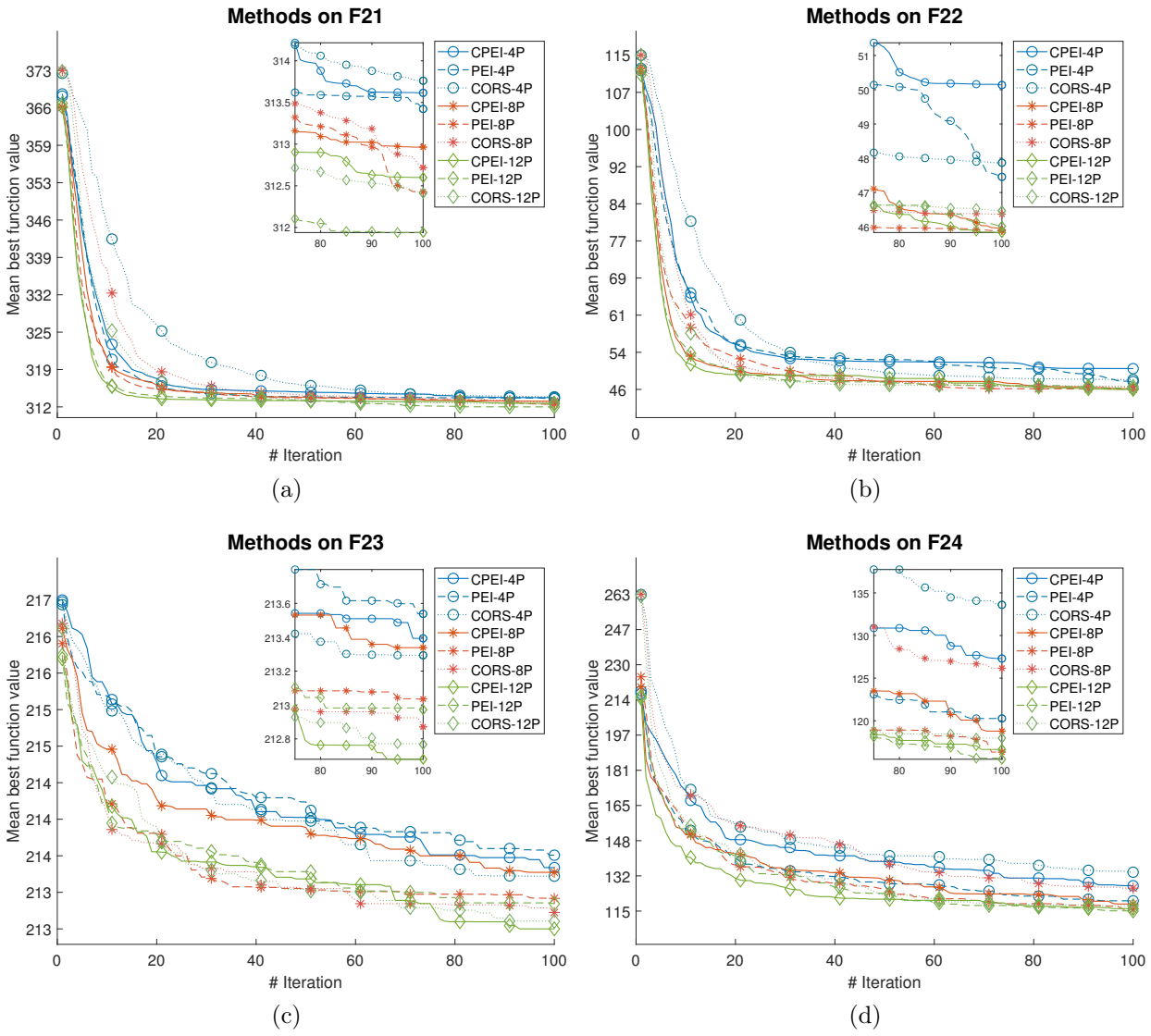

Figure 5: Best objective function value founded average over twenty trials versus major iteration. Global optimization methods on the test functions F21-F24 (taken from BBOB test suite)