Nonlinear Analysis: Modelling and Control, Vol. 27, No. 3, 534–555
https://doi.org/10.15388/namc.2022.27.26558

Vilnius University Press

A hybrid of Bayesian-based global search with Hooke–Jeeves local refinement for multi-objective optimization problems

Linas Litvinas[a]

Institute of Computer Science, Vilnius University,
Didlaukio str. 47, LT-08303 Vilnius, Lithuania
linas.litvinas@mif.vu.lt

Received: August 18, 2021 / **Revised:** February 2, 2022 / **Published online:** March 28, 2022

Abstract. The proposed multi-objective optimization algorithm hybridizes random global search with a local refinement algorithm. The global search algorithm mimics the Bayesian multi-objective optimization algorithm. The site of current computation of the objective functions by the proposed algorithm is selected by randomized simulation of the bi-objective selection by the Bayesian-based algorithm. The advantage of the new algorithm is that it avoids the inner complexity of Bayesian algorithms. A version of the Hooke–Jeeves algorithm is adapted for the local refinement of the approximation of the Pareto front. The developed hybrid algorithm is tested under conditions previously applied to test other Bayesian algorithms so that performance could be compared. Other experiments were performed to assess the efficiency of the proposed algorithm under conditions where the previous versions of Bayesian algorithms were not appropriate because of the number of objectives and/or dimensionality of the decision space.

Keywords: global optimization, Bayesian algorithm, Hooke–Jeeves, local refinement.

1 Introduction

The Bayesian approach is one of the most active in the development of methods for non-convex black-box optimization [1, 4, 27]. Despite active research, some challenges remain unresolved; see, e.g., the review in [27]. A serious challenge in widening the application area of Bayesian algorithms is their inner complexity. One way to reduce the complexity could be by the partition-based implementation which was successful in Lipschitz optimization [12, 17, 19–21, 32]. Indeed, the inner complexity of partition-based Bayesian algorithms for single and multi-objective optimization was proved to be lower than that of the standardly implemented algorithms [25, 29, 31]. Nevertheless, the complexity of these algorithms still limits the number of evaluations of the objective functions which would be appropriate for solving problems of higher dimensionality and problems with a larger number of objectives. In the present paper a multi-objective

optimization algorithm implementing heuristically the ideas related to the Bayesian approach without using a formal model of uncertainty about the objective functions is presented. A version of the bi-objective selection of a site for the current computation of the objective functions is applied following the idea proposed in [24]. Two heuristic criteria are used for the selection of a computation point: an estimate of its distance to the Pareto front and the uncertainty of the estimate. These criteria mimic the stochastic model-based assessments used in [24] which assess the exploitation and exploration character of the search strategy. The balance between exploration and exploitation can be varied rather broadly by accepting an appropriate compromise between the expectation and uncertainty criteria. However, the local refinement of the approximate solutions is frequently more efficient using a local search method than by a locally biased global search method [18, 22, 26]. The hybridization of the single-objective Bayesian global search with local optimization methods is shown to be efficient in [25, 30]. In the present paper a local search algorithm for the refinement of the approximation of the Pareto front found by the global search algorithm was applied. A version of the Hooke–Jeeves algorithm is adapted to multi-objective optimization. The proposed hybrid algorithm is tested under conditions previously applied to test other Bayesian algorithms [28, 31] so that performance could be compared. Other experiments were performed to assess the efficiency of the proposed algorithm under conditions where the previous versions of Bayesian algorithms were not appropriate. The results were compared with the results of the popular evolutionary optimization algorithms NSGA2 and NSGA3 [3, 7, 8] using test problems up to 15 objective functions with up to 30 variables.

2 The proposed hybrid algorithm

Black-box multi-objective minimization problems:

$$\min F(x), \quad x \in A \subset \mathbb{R}^d,\ F(x) = \big(f_1(x), \ldots, f_m(x)\big)^{\mathrm{T}},$$

are considered assuming that the feasible region (decision space) is a unit hypercube $A = [0,1]^d$ to which any hyperrectangular region can be rescaled. Ranges of the objective functions are equal and rescaling is provided by the algorithm.

The algorithm consists of two alternating counterparts carried out multiple times: random global search and local refinement of the Pareto front approximation found by the global search algorithm. The Bayesian global optimization strategy would be preferable because of the rational balancing of exploration and exploitation. However, the number of iterations (computations of the objective functions) of the Bayesian algorithms is considerably limited due to their inner computational complexity. The number of iterations is limited to several hundred for the standardly implemented Bayesian algorithms, and several thousand for the Bayesian partition-based algorithms [27]. The idea is to mimic the search strategy of the Bayesian algorithm without using a stochastic model of the objective functions and thus avoiding the basic computational burden. A randomized algorithm is proposed where the criteria for selecting a point for computing the objective functions are similar but much simpler than the Bayesian approach-based criteria [24].

The reduced computational burden allows a considerably larger number of iterations; see the section on testing results. The global search phase interchanges with the local refinement phase. The approximation of the Pareto front is refined by a version of the Hooke–Jeeves algorithm adapted to multi-objective optimization. The process of alternating phases of global and local searches is terminated by a user-defined termination condition.

The considered algorithm is initialized by the predefined number of computations of the objective functions at the random points uniformly distributed in the feasible region A.

2.1 Global search

The proposed algorithm consists of two alternating counterparts carried out multiple times: global search and local refinement. Global search runs after the initialization or after previous global-local stages and thereafter is altered with local search. Let us consider the current start of global search; the set of points where the objective functions are already computed is denoted by

$$U_A = \{x_i \in A \mid i = 1, \ldots, N\},$$

and the corresponding set of function values is denoted by

$$U = \big\{F(x) \;\big|\; x \in U_A\big\},$$

where non-dominated points of U constitute current Pareto optimal solutions set in objective space $P \subseteq U$ and corresponding Pareto optimal solutions set in decision space $P_A \subseteq U_A$. This kind of global search with the uniform distribution of function evaluation points is a worst-case optimal algorithm in case function evaluation budget is predefined and optimized multi-objective functions are Lipschitz functions [23]. The main disadvantage of this method is that when a decision space dimensionality increases the function evaluation count increases exponentially. However, the worst-case is very specific: the Pareto optimal set of optimized multi-objective function have a single point, and the solutions are represented by a single point [23]. Most optimization problems in engineering and science are not worst-case so some strategies improving the method's performance by imitating Bayesian search can be applied [2, 14, 29]. Bayesian approach-based optimization algorithms search new location in decision space for a new function evaluation based on trade-off values of statistically assessed function's mean value and variance [16, 28]. The location of decision space with near-optimal function's mean value has a chance to be a new optimal point even in the case of little improvement of the function's evaluation value. Otherwise, a location with a high function's variance value has a chance of being a new optimal point in the case a vast improvement of the function's evaluation value. However, the computational burden of searching location in decision space for next function evaluation is limiting a Bayesian multi-objective optimization algorithm's application area. Coping with such computational burden the implementation based on the rectangular partition of the feasible region is proposed in [29, 31]. The vertices count of hyperrectangle doubles with every new dimension. The algorithm is

an iterative hyperrectangle subdivision by bisection. The hyperrectangle is bisected by a hyperplane orthogonal to its longest edges. The function evaluations are computed at intersection points and the evaluation count is only half of the hyperrectangle vertice count. This way rectangular partition-based Bayesian algorithm is not competitive for optimization problems having high dimensional decision space.

Strategies to mimic Bayesian search should combine simplicity and numerical substantiation. Combining both mentioned properties the exploration-exploitation strategy can be used, i.e., let us say new random points $U_A^{\text{new}} = \{x_i \in A \mid i = 1, \ldots, q \cdot N\}$ (q is parameter value) are uniformly generated in feasible region A and for every point $x_i \in U_A^{\text{new}}$ bi-objective selection functions are calculated:

$$\theta_1(x_i) = \|x_i - x_{\min}\|, \qquad x_{\min} = \arg\min_{x \in U_A} \|x_i - x\|, \tag{1$_1$}$$

$$\theta_2(x_i) = \big\|F(x_{\min}) - F_{\min}\big\|, \qquad F_{\min} = \arg\min_{F \in P} \big\|F(x_{\min}) - F\big\|, \tag{1$_2$}$$

where $\theta_1(x_i)$ is a generated point's $x_i \in U_A^{\text{new}}$ Euclidean distance to the closest known point $x_{\min} \in U_A$, and $\theta_2(x_i)$ is the closest known point's $x_{\min} \in U_A$ function evaluation vector's $F(x_{\min}) \in U$ Euclidean distance to the closest current Pareto optimal solution $F_{\min} \in P$. In case when the closest known point's function evaluation vector $F(x_{\min}) \in U$ is non-dominated Pareto optimal solution $F(x_{\min}) \in P \subseteq U$, the function $\theta_2(x_i)$ has zero value. Please note that $\theta_2(x_i)$ value is calculated using min-max normalized function $F(x) = (f_1(x), \ldots, f_m(x))^{\mathrm{T}}$ values. An ideal $F^* = (f_1^*, \ldots, f_m^*)^{\mathrm{T}}$ and a nadir $F^{\text{nad}} = (f_1^{\text{nad}}, \ldots, f_m^{\text{nad}})^{\mathrm{T}}$ point values are taken as minimum and maximum function values [3]. Normalized function values can be expressed by the following formula:

$$F^{\text{norm}}(x) = \left(\frac{f_1(x) - f_1^*}{f_1^{\text{nad}} - f_1^*}, \ldots, \frac{f_m(x) - f_m^*}{f_m^{\text{nad}} - f_m^*}\right)^{\mathrm{T}}. \tag{2}$$

Function $\theta_1(x_i)$ value can be viewed as the radius of hypersphere with center point $x_i \in U_A^{\text{new}}$ and with no function evaluations inside of it. In case when function $\theta_1(x_i)$ has a large value, a large unexplored hypersphere is found, and new function evaluation in its center (point x_i) is reasonable as this corresponds to the exploration strategy. On the other hand, the second function $\theta_2(x_i)$ value can be viewed as the minimal distance of optimized function's $F(x)$ nearest-neighbor interpolation value at point x_i to current Pareto optimal solutions. In the case of a small second objective function $\theta_2(x_i)$ value, the generated point's $x_i \in U_A^{\text{new}}$ closest known point $x_{\min} \in U_A$ has near-optimal function value $F(x_{\min}) \in U$, so the new function evaluation at point x_i is reasonable as this corresponds to the exploitation strategy. The exploration-exploitation strategy is achieved by making new function evaluations at points $x_i \in U_A^{\text{new}}$ having minimal trade-off Pareto optimal values of selection functions $(-\theta_1(x_i), \theta_2(x_i))^{\mathrm{T}}$. Function $\theta_1(x_i)$ has a negative sign because it is maximized. After new function evaluations are made, the sets U, U_A, P, P_A are updated and reused in next phases or next iterations of the algorithm. Reuse of search information is shown to be efficient for multi-objective optimization problems [11].

To induce function evaluations clustering near Pareto optimal solutions, the global search has two random points generation modes: global uniform generation in all feasible

region A and local uniform generation near current Pareto optimal solutions, i.e., for all Pareto optimal solutions $x_P \in P_A$, a series of small hypercubes $A_1, \ldots, A_n$ having center point x_P is created, and the above-described procedure is applied to a particular subregion of feasible region $A_n \subset \cdots \subset A_1 \subset A$. Hypercube A_i has a double-sized edge compared to next A_{i+1} hypercube, and the shrinking of hypercubes stops when there are no known point solutions in A_{i+1} except center point $x_P \in P_A$.

2.2 Local refinement

The proposed exploration-exploitation global search strategy lacks effective local refinement since randomly generated points lack improvement direction. On the contrary, the Hooke–Jeeves single-objective optimization algorithm searches function improvement direction by taking a step in all decision variables [13]. A multi-objective optimization problem can be reduced to a single-objective optimization problem by scalarization methods such as Chebyshev scalarization [15]. The Chebyshev scalarization requires a weight vector to get a single optimal trade-off solution. To get an approximation of Pareto optimal front, a multiple uniformly generated weight vectors should be used. Although weight vectors are uniformly generated, the resulting Pareto front approximation may be non-uniformly distributed, and human expert intervention may be needed to generate additional weight vectors to get uniformly distributed Pareto front approximation [2, 14]. To avoid complicated weight vectors selection, a weight-free approach to multi-objective optimization problems using a modified version of Hooke–Jeeves direct search is presented [5]. In this paper a novel approach for conversion of a multi-objective optimization problem to a single-objective optimization problem without the use of the weight vectors is suggested.

The multi-objective function is converted to a single-objective surrogate function $f_s(x)$, $x \in A$, which has a current solution: a decision vector x_{cur} and multi-objective function evaluation vector $F(x_{\mathrm{cur}})$. Initially, the surrogate function has predefined value $f_s(x_{\mathrm{cur}}) = 0$. When the new solution x_{new} dominates the current solution $F(x_{\mathrm{cur}}) \prec F(x_{\mathrm{new}})$, the surrogate function value at new location x_{new} decreases $f_s(x_{\mathrm{new}}) = f_s(x_{\mathrm{cur}}) - 1$, and the current solution is updated $x_{\mathrm{cur}} = x_{\mathrm{new}}$. Otherwise, when the new solution x_{new} does not dominate the current solution $F(x_{\mathrm{cur}}) \not\prec F(x_{\mathrm{new}})$, the surrogate function value at new location x_{new} remains the same $f_s(x_{\mathrm{new}}) = f_s(x_{\mathrm{cur}})$, and the current solution x_{cur} is not updated, i.e.,

$$f_s(x) = \begin{cases} f_s(x_{\mathrm{cur}}) - 1 & \text{when } F(x_{\mathrm{cur}}) \prec F(x) \\ & \text{(current solution updated: } x_{\mathrm{cur}} = x\text{)}, \\ f_s(x_{\mathrm{cur}}) & \text{when } F(x_{\mathrm{cur}}) \not\prec F(x). \end{cases}$$

For every current Pareto optimal solution $x_P \in P_A$, a separate surrogate function having Pareto optimal solution as current solution $x_{\mathrm{cur}} = x_P$ is defined. Every defined surrogate function is optimized using the Hooke–Jeeves optimization algorithm taking x_P as the start point:

$$\min f_s(x), \quad x \in A \subset \mathbb{R}^d.$$

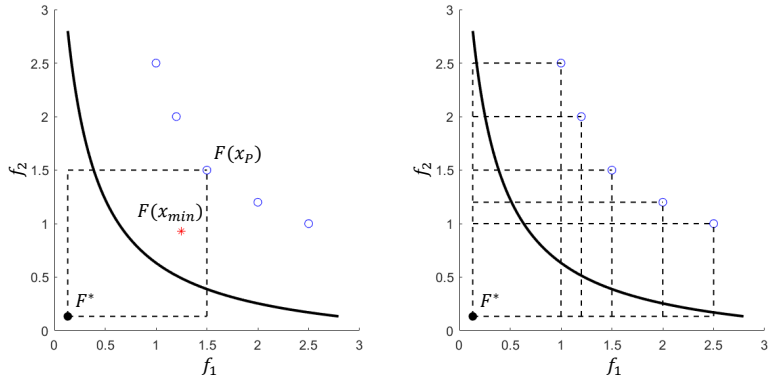

Figure 1. Example Pareto front (a) of minimized bi-objective function $F(x) = (f_1(x), f_2(x))^{\mathrm{T}}$. Blue circles are the set of current Pareto optimal function evaluations – P. For Pareto optimal solution x_P, a minimized surrogate $f_s(x)$ function is defined ($f_s(x_P) = 0$). A surrogate function value at minimal point is negative $f_s(x_{\min}) < 0$, $x_{\min} = \arg\min_{x \in A} f_s(x)$, so the value of the original bi-objective function at the newly found solution $F(x_{\min})$ (red asterisk) dominates bi-objective function's value at the initial point $F(x_P)$. Function evaluation $F(x_{\min})$ at solution found by surrogate function minimization $x_{\min} = \arg\min_{x \in A} f_s(x)$ is in the area bounded by the Pareto front and rectangle defined by points: $F(x_P)$ and ideal point of minimal function values $F^* = (f_1^*, f_2^*)^{\mathrm{T}}$ (black dot). On the right (b), surrogate functions are defined for all current Pareto optimal solutions (blue circles). Possible solutions found using surrogate functions minimization cover almost the entire Pareto front except for small parts near minimal values of functions $f_1(x)$ and $f_2(x)$.

In the case of negative value of surrogate function at minimal point $f_s(x_{\min}) < 0$, $x_{\min} = \arg\min_{x \in A} f_s(x)$, the value of original multi-objective function at the newly found solution $F(x_{\min})$ dominates multi-objective function's value at Hooke–Jeeves optimization start point $F(x_P)$, i.e., the newly found solution's value $F(x_{\min})$ is located closer to the true Pareto front compared with the initial function value $F(x_P)$. Function evaluation $F(x_{\min})$ at solution found by surrogate function minimization $x_{\min} = \arg\min_{x \in A} f_s(x)$ is in the area bounded by true Pareto front and hyperrectangle defined by points: $F(x_P)$ and ideal point of minimal function values $F^* = (f_1^*, \ldots, f_m^*)^{\mathrm{T}}$. An example of the bi-objective case is shown in Fig. 1.

2.3 Implementation and pseudo-code

The basic concepts of the proposed optimization algorithm are presented in previous subsections, but a more detailed explanation of the whole picture is still needed. Therefore the pseudo-code of the proposed optimization algorithm is given in Algorithm 1. List of the notations used:

1. F – a multi-objective optimized function.
2. d – count of variables.
3. $A = [0, 1]^d$ – a feasible region of unit hypercube.
4. U_A – set of all points in a feasible region $A = [0, 1]^d$.

5. U – set of all function evaluations $\{F(x) \mid x \in U_A\}$.
6. P_A – points in a feasible region $A = [0,1]^d$ of current non-dominated Pareto optimal solutions.
7. P – function evaluations of current non-dominated Pareto optimal solutions.
8. $N_{\max}$ – maximum allowed number of function $F(\cdot)$ evaluations.
9. $I_{\max}$ – maximum iteration number of global search with local refinement.
10. N – number of initial random solutions.
11. $q \cdot N$ – number of randomly generated candidate points in decision space for new function evaluations.
12. p – part $p \in [0,1]$ of function evaluations get by local generation near current Pareto optimal solutions compared to function evaluations get by global generation in the entire feasible region A.
13. h_0 and h_n – step size parameters of the Hooke–Jeeves optimization algorithm. The full set of step sizes from largest to smallest are calculated using the following expression: $\{0.8 \cdot 2^{-i} \mid h_0 \leqslant i \leqslant h_n\}$, where the largest step size is $0.8 \cdot 2^{-h_0}$ and the smallest step size is $0.8 \cdot 2^{-h_n}$.
14. *Update* – a Boolean parameter value if it is set to true, the step size parameters h_0 and h_n are updated at the second and following iterations of global search with local refinement. For every Pareto optimal solution $x_P \in P_A$ the value h_0 is updated so that the largest step size would be approximately equal to minimal Euclidean distance to another Pareto optimal solution: $0.8 \cdot 2^{-h_0} \approx \min_{x \in P_A} \|x_P - x\|$. The value h_n is also updated so that the smallest step size would be smaller than the largest: $h_n = \max\{h_0 + 2,\, h_n\}$.

Some remarks explaining the pseudo-code of the proposed optimization algorithm will be given. At the first algorithm's iteration ($I = 1$), the local refinement phase optimizes not only the surrogate function but also every single-objective function of the optimized multi-objective function $f_i(\cdot) \in F(\cdot)$ to get limit solutions of Pareto front approximation. Solutions that have been found using the surrogate function or single criteria function optimization are not used to define and optimize the new surrogate function anymore because the surrogate function value at this point is near-optimal already. Instead, a non-dominated Pareto optimal solutions found by the global search phase are used to define and optimize new surrogate functions this way inducing a search in the unexplored area to find new Pareto optimal solutions. The step size parameters h_0 and h_n are updated (value of parameter *Update* must be set to true) so that the Hooke–Jeeves optimization algorithm's largest step size would be approximately equal to minimal Euclidean distance to another Pareto optimal solution. In the case of a small distance to another Pareto optimal solution, only small adjustments are needed so small step sizes should be used. In the case of a large distance to another Pareto optimal solution, the new distantly located solution is found, therefore more excessive local search is needed during surrogate function optimization so that initially large step sizes should be used. Please note that the global search phase can be reduced to simple uniform distribution of function evaluation points in case when the following parameters are selected: $q = 1/N$, $p = 0$.

Algorithm 1. The pseudo-code of the proposed optimization algorithm.

procedure OPTIMIZE(F)
  $U_A \leftarrow$ randomly generated N points in feasible region $A = [0,1]^d$,
  $U \leftarrow$ function evaluation $F(x)$ of randomly generated points $x \in U_A$,
  $P_A \leftarrow$ points in feasible region $A = [0,1]^d$ of current non-dominated Pareto optimal solutions,
  $P \leftarrow$ function evaluations of current non-dominated Pareto optimal solutions,
  $HJ \leftarrow$ set of solutions found by the Hooke–Jeeves algorithm, $I \leftarrow 0$ iteration count,
  while $|U_A| \leqslant N_{\max}$ and $I \leqslant I_{\max}$ **do**
    $I \leftarrow I + 1$,
    if $p > 0$ **then**
      $P_{\text{old}} \leftarrow P_A$,
      for $x_P \in P_{\text{old}}$ **do**
        $A_1 \leftarrow x_P$ centered hypercube having edge size 0.2,
        while A_1 have inside only one point x_P **do**
          Increase the edge size of hypercube A_1 by 0.2,
        end while
        $i \leftarrow 1, A_i \leftarrow A_1$,
        while A_i has inside more than one point x_P and edge size $\geqslant 2^{-h_n}$ **do**
          $U_A^{\text{new}} \leftarrow$ generated random points $\{x_i \mid i = 1, \ldots, q \cdot N\}$ inside of hypercube A_i,
          Make function $F(\cdot)$ evaluations at points $x_i \in U_A^{\text{new}}$ having minimal trade-off values of selection functions $(-\theta_1(x_i), \theta_2(x_i))^{\mathrm{T}}$,
          Update the sets $U,\ U_A,\ P,\ P_A$ with new function evaluations,
          $A_{i+1} \leftarrow$ halve edge of A_i, $i \leftarrow i + 1$,
        end while
      end for
    end if
    while $(1 - p) >$ part of function evaluations get by generation in the entire feasible region A **do**
      $U_A^{\text{new}} \leftarrow$ generated random points $\{x_i \mid i = 1, \ldots, q \cdot N\}$ inside of hypercube A,
      Make function $F(\cdot)$ evaluations at points $x_i \in U_A^{\text{new}}$ having minimal trade-off values of selection functions $(-\theta_1(x_i), \theta_2(x_i))^{\mathrm{T}}$,
      Update the sets $U,\ U_A,\ P,\ P_A$ with new function evaluations,
    end while
    $P_{\text{old}} \leftarrow P_A \setminus HJ$,
    for $x_P \in P_{\text{old}}$ **do**
      Define surrogate function $f_s(x)$ having current solution $x_{\text{cur}} \leftarrow x_P$,
      if $I > 1$ and $Update \equiv true$ **then**
        Update step size parameters h_0 and h_n of Hooke–Jeeves algorithm,
      end if
      Optimize $x_{\min} = \arg\min_{x \in A} f_s(x)$ using Hooke–Jeeves and taking x_P as start point,
      Complement set of solutions with new solution $HJ \leftarrow HJ \bigcup \{x_{\min}\}$,
      Update the sets U, U_A, P, P_A with new function evaluations,
    end for
    if $I \equiv 1$ **then**
      $P_{\text{old}} \leftarrow P_A$,
      for $f_i(\cdot) \in F(\cdot)$ **do**
        Find solution with minimal value of current function $x_P = \arg\min_{x \in P_{\text{old}}} f_i(x)$,
        Optimize $x_{\min} = \arg\min_{x \in A} f_i(x)$ using Hooke–Jeeves and taking x_P as start point,
        Complement set of solutions with new solution $HJ \leftarrow HJ \bigcup \{x_{\min}\}$,
        Update the sets U, U_A, P, P_A with new function evaluations,
      end for
    end if
  end while
  return $\{U, U_A, P, P_A\}$.
end procedure

The global search phase interchanges with the local refinement phase multiple times. The algorithm works until the maximum allowed number of function $F(\cdot)$ evaluations or the maximum iteration number of global searches with local refinement is reached. The algorithm returns a set of all function evaluations and a set of non-dominated Pareto optimal solutions.

3 Performance analysis

This section first presents theoretical convergence analysis of the proposed algorithm, and then the performance of the proposed algorithm is evaluated by the numerical experiments. Many optimization problems in engineering and science have very limited function evaluation budget, so the proposed optimization algorithm was tested in case of extremely low function evaluation budget, and the results were compared with the results of Bayesian rooted optimization algorithms [31]. On the other hand, a good performance in case of the high dimensionality of feasible region is a desirable feature, so the optimization algorithm was tested with test suite ZDT having many decision variables (up to 30) [33]. Finally, test suite DTLZ was used in case of many-objective (up to 15) optimization problems [9]. The results of the last two cases were compared with the results of popular evolutionary optimization algorithms NSGA2 and NSGA3 [3, 7, 8].

3.1 Convergence analysis

To prove the function evaluations in the decision space of the proposed algorithm are everywhere dense, let part of function evaluations get by generation in the entire feasible region A have a non-zero value ($1 - p > 0$), and a number of generated random sample points inside of a unit hypercube A have $q \cdot N \geqslant 2$ value. Let a maximum allowed number of function evaluations and a maximum iteration number of a global search with a local refinement approach infinite ($N_{\max} \to \infty$, $I_{\max} \to \infty$), then points of function evaluations in a decision space U_A are everywhere dense in the decision space A.

Let us say the opposite, there remains some hypersphere S having inside no points from the set U_A of function evaluations in a decision space. Let the radius value of the hypersphere S be $\varepsilon > 0$, and a center located at a point $x_S \in A$ in a feasible area. Let us define a hypersphere S_0 having a radius value $\varepsilon/3$ and a center located at the same point x_S, hereby this newly defined hypersphere is inside of the hypersphere S, $S_0 \subset S$. Let us define another hypersphere S_1 having a radius value $\varepsilon/3$ and a center located at a point x_{S_1} which is a point of function evaluations in a decision space $x_{S_1} \in U_A$. During every iteration of the proposed algorithm, a number $q \cdot N$ of random sample points are uniformly generated inside of the hypercube A. Let us define an event of the case, then one of random sample points x_{rand} hits inside of the hypersphere S_0, so the Euclidean distance between points x_S and x_{rand} is less than a radius of the hypersphere S_0, i.e., $\|x_S - x_{\text{rand}}\| < \varepsilon/3$, and the remaining generated random points $\{x_i \mid i = 1, \ldots, q \cdot N - 1\}$ hit inside of the hypersphere S_1, so the Euclidean distances between points x_{S_1} and x_i, $i = 1, \ldots, q \cdot N - 1$ are less than a radius of the hypersphere S_1,

i.e., $\|x_{S_1} - x_i\| < \varepsilon/3$, $i = 1, \ldots, q \cdot N - 1$. Such a defined event has a non-zero probability value, and it can be calculated in the following way. Assume that the value of volumes of parts of the hypersphere S_0 and the hypersphere S_1 inside of the feasible area A are noted as V_{S_0}, V_{S_1}. Volumes will have non-zero values $V_{S_0} > 0, V_{S_1} > 0$, as well as the value $V_A = 1 > 0$ of the unit hypercube volume of the feasible area A. Then the probability of the defined event has a non-zero constant value of $C_{q \cdot N}^1 \cdot (V_{S_0}/V_A) \cdot (V_{S_1}/V_A)^{q \cdot N - 1} > 0$. This means the defined event will definitely occur as a maximum iteration number of a global search with a local refinement approach infinite $I_{\max} \to \infty$, and so experiment count approaches infinite. Let us say the defined event occurred at an iteration I_1. Selection functions are calculated at points $(-\theta_1(x_i), \theta_2(x_i))^{\mathrm{T}}$, $i = 1, \ldots, q \cdot N - 1$, and at a point $(-\theta_1(x_{\mathrm{rand}}), \theta_2(x_{\mathrm{rand}}))^{\mathrm{T}}$, taking points having minimal trade-off Pareto optimal values as evaluation points of an optimized function $F(\cdot)$. A selection function $\theta_1(x)$ is the generated point's x Euclidean distance to the closest known point $x_{\min} \in U_A$ (as defined by Eq. (1)). Therefore, these inequalities are valid: $\theta_1(x_{\mathrm{rand}}) > 2\varepsilon/3$ and $\theta_1(x_i) < \varepsilon/3$, $i = 1, \ldots, q \cdot N - 1$, so the selection function $\theta_1(x_{\mathrm{rand}})$ at a point x_{rand} has the biggest value; consequently, a vector of selection functions values $(-\theta_1(x_{\mathrm{rand}}), \theta_2(x_{\mathrm{rand}}))^{\mathrm{T}}$ at a point x_{rand} will belong to minimal trade-off Pareto optimal values of the selection functions, so the optimized function $F(\cdot)$ is evaluated at a point x_{rand}, and a new point is added to the set of function evaluations in the decision space $U_A \leftarrow U_A \bigcup \{x_{\mathrm{rand}}\}$. This means a function evaluation point $x_{\mathrm{rand}} \in U_A$ is added to the hypersphere S_0 and S as $S_0 \subset S$, and this is a contradiction to the statement that there remains some hypersphere S having inside no points from the set of function evaluations in the decision space U_A. This means that points of U_A are everywhere dense in the decision space A.

To prove the convergence of the proposed algorithm, let the optimized multi-objective function $F(x) = (f_1(x), \ldots, f_m(x))^{\mathrm{T}}$, $x \in A$, consist of Lipschitz-continuous functions $f_1(x), \ldots, f_m(x)$, and A is a feasible area of a unit hypercube. Let the part of function evaluations get by generation in the entire feasible region A have a non-zero value $(1 - p > 0)$, and a number of generated random sample points inside of the unit hypercube A have $q \cdot N \geqslant 2$ value, and let a maximum allowed number of function evaluations and a maximum iteration number of a global search with a local refinement approach infinite $N_{\max} \to \infty, I_{\max} \to \infty$, then to any point on the true Pareto front $x^* \in A$, $F(x^*)$ the proposed algorithm will find a solution $x_P \in U_A$, $F(x_P)$ with any small error value $\varepsilon > 0$ as an Euclidean distance between the found solution and a point on the true Pareto front $\|x^* - x_P\| < \varepsilon$.

Let us say the opposite, there is a point on the true Pareto front $x^* \in A$, $F(x^*)$ having no solution of the proposed algorithm with an error value ε. This means there is a hypersphere S with a radius value ε and a center located at a point x^* having inside no points from a set U_A of function evaluations in a decision space. But we already proved that points of U_A are everywhere dense in the decision space A. This means there is a solution $x_P \in U_A$, $F(x_P)$ inside of the hypersphere S and, consequently, inequality $\|x^* - x_P\| < \varepsilon$ is valid, and this is a contradiction to the statement that there is a point on the true Pareto front $x^* \in A$, $F(x^*)$ having no solution of the proposed algorithm with an error value ε. This means to any point on the true Pareto front $x^* \in A$, $F(x^*)$

the proposed algorithm will find a solution $x_P \in U_A$, $F(x_P)$ with any small error value $\varepsilon > 0$.

The investigation of the rate of convergence is more complex. On the other hand, the performance of the proposed algorithm is evaluated by the numerical experiments presented in the next subsections.

3.2 Numerical experiments having low functions evaluation budget

In case when experiments have low functions evaluation budget, the results of the proposed algorithm were compared with the results of Bayesian rooted optimization algorithms: standard and partition-based implementations of the P-algorithm [31]. Several test problems are solved to illustrate the performance of Bayesian rooted optimization algorithms [31]. The same test problems will be used here.

Frequently used multi-objective non-convex test problems were proposed in [10]. The objective functions are defined as follows:

$$f_1(x) = 1 - \mathrm{e}^{-\sum_{i=1}^{d}(x_i - 1/\sqrt{d})^2}, \tag{3$_1$}$$

$$f_2(x) = 1 - \mathrm{e}^{-\sum_{i=1}^{d}(x_i + 1/\sqrt{d})^2}, \tag{3$_2$}$$

where $d = 2$, and the feasible region is A: $-4 \leqslant x_1, x_2 \leqslant 4$. The next bi-objective test problem is two Shekel functions frequently used to evaluate global optimization algorithms [16]:

$$\begin{aligned} f_1(x) = &-\frac{0.1}{0.1 + (x_1 - 0.1)^2 + 2(x_2 - 0.1)^2} \\ &- \frac{0.1}{0.14 + 20((x_1 - 0.45)^2 + (x_2 - 0.55)^2)}, \end{aligned} \tag{4$_1$}$$

$$\begin{aligned} f_2(x) = &-\frac{0.1}{0.15 + 40((x_1 - 0.55)^2 + (x_2 - 0.45)^2)} \\ &- \frac{0.1}{0.1 + (x_1 - 0.3)^2 + (x_2 - 0.95)^2}, \end{aligned} \tag{4$_2$}$$

where the feasible region is A: $0 \leqslant x_1, x_2 \leqslant 1$.

For the comparison performance of the proposed algorithm having low functions evaluation budget, the metrics applied which were used in a recent publication related to Bayesian rooted optimization algorithms [31]. One of the most simple and most popular performance metrics is the number of non-dominated solutions found by an optimization algorithm (NN). The estimation of the distance between the found approximation and the true Pareto front is the so-called generational distance (GD) [6]. GD is computed as the maximum of distances between the found non-dominated solutions and their closest neighbors from the Pareto front. A metric integrating measures of the approximation precision and spread is the so-called epsilon indicator (EI) [34]. EI is computed as the

maximum of distances between the true Pareto front and their closest neighbors from found non-dominated solutions. Metrics GD and EI can be expressed by the following equations:

$$\mathrm{GD} = \max_{F_P \in P} \min_{F_{P^*} \in P_*} \|F_P - F_{P^*}\|,$$
$$\mathrm{EI} = \max_{F_{P^*} \in P_*} \min_{F_P \in P} \|F_P - F_{P^*}\|,$$

where P is a set of non-dominated solutions found by the considered algorithm, and P^* is a set of solutions well representing the Pareto front, i.e., the solutions are sufficiently densely and uniformly distributed over the Pareto front.

The proposed algorithm was tested with 100 function $F(x)$ evaluation budget. The proposed algorithm was run with the following parameters:

$$(N_{\max}, I_{\max}, N, q, p, h_o, h_n, \mathit{Update})^{\mathrm{T}} = (90, \infty, 20, 10000, 0.8, 2, 4, \mathit{true})^{\mathrm{T}}.$$

The parameter $N_{\max}$ value is selected 10% below function evaluation budget since stop condition is checked once every iteration after global search and local refinement phase is finished. On average function evaluation budget of 100 evaluations is not exceeded, but some algorithm's runs make more, and some runs make fewer function evaluations. The same parameter $N_{\max}$ selection strategy will be used in this paper. The number of iteration $I_{\max}$ is selected to be the maximum number that cannot be achieved, so it can be noted as infinite ($I_{\max} = \infty$). More parameters are N and q, where $N = 20$ is the number of initial random function evaluations, and $q = 10000$ is the coefficient value of expression $q \cdot N$ giving the number of randomly generated candidate points in decision space for new function evaluations. The number of randomly generated candidate points should be high in the case of low functions evaluation budget so such a high value of q is selected. Value of $p = 0.8$ is part of function evaluations get by local generation near current Pareto optimal solutions compared to function evaluations get by global generation in all feasible region A. The step size parameters of the Hooke–Jeeves optimization algorithm have the following values: $h_0 = 2$ and $h_n = 4$. The parameter Update is set to true, so the step size parameters h_0 and h_n are updated at second and following iterations of global search with local refinement. The same parameters were used for both test problems except when the parameter's h_o value was $h_o = 4$ for problem (4).

Since the proposed algorithm and P-algorithm are stochastic, the test problems were solved 100 times [16, 28]. The mean values and standard deviations of the considered metrics are present in two columns of Table 1. Otherwise, the hyperrectangle partition-based P-algorithm is deterministic, so its results occupy a single column for each test problem [31]. The proposed algorithm shows decent performance. In case of problem (3) the proposed algorithm gives better NN and EI values than the P-algorithm. In case of problem (4) the proposed algorithm gives the best NN value and gives better GD and EI values than the hyperrectangle partition-based P-algorithm.

For visualization of the proposed algorithm's operation, the solutions of problems (3) and (4) having the best metric EI value are presented in Fig. 2. The Pareto front is well

Table 1. Mean values and standard deviations of performance criteria (NN, GD, EI) of the P-algorithm, partition-based implementations of the P-algorithm and the proposed algorithm for test problems (3) and (4).

Criteria	P-algorithm				Partition-based		Proposed algorithm			
	Problem (3)		Problem (4)		Problem (3)	Problem (4)	Problem (3)		Problem (4)	
NN	9.87	1.4	15.7	2.0	27	18	12.61	3.11	25.35	5.378
GD	0.015	0.0061	0.07	0.051	0.015	0.21	0.052	0.025	0.161	0.084
EI	0.2	0.034	0.13	0.053	0.092	0.25	0.139	0.018	0.204	0.069

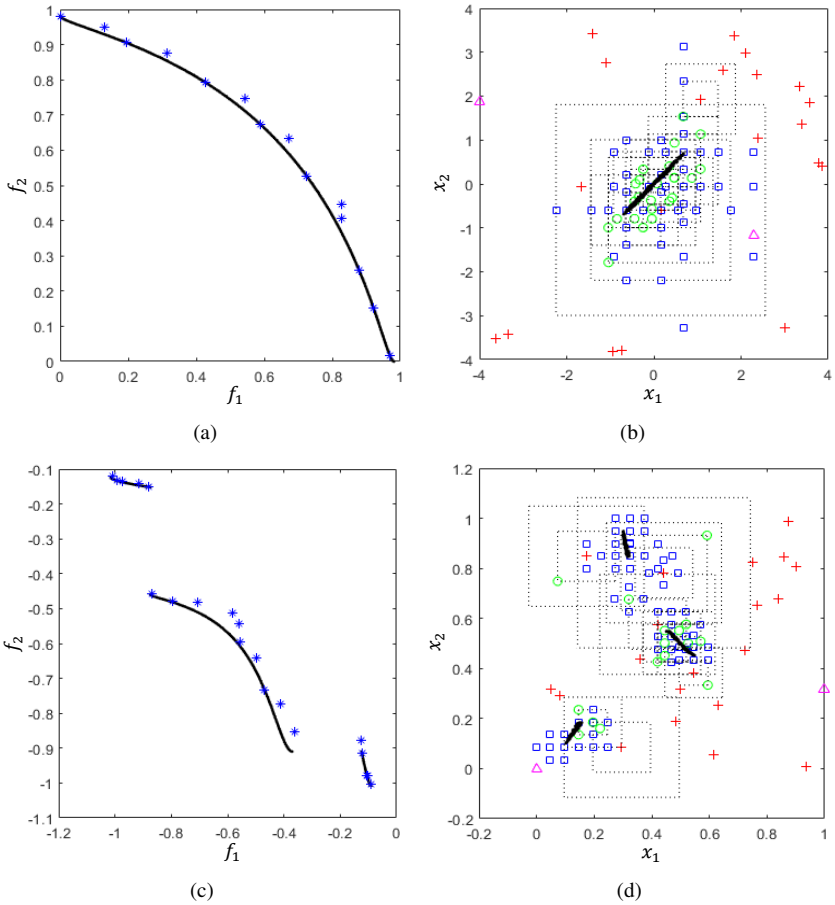

Figure 2. Pareto front (black line) and Pareto optimal solutions (blue asterisk) found by the proposed algorithm for problem (3) (NN = 14, GD = 0.0401, EI = 0.0782) (a) and problem (4) (NN = 20, GD = 0.0549, EI = 0.0733) (c). The function evaluation points made by the proposed algorithm and line of Pareto optimal solutions in decision space (black line) are accordingly in (b) and (d), where red plus marks N initial random evaluations points, green circle marks the evaluation points made by candidate points generation near current Pareto optimal solutions in the square areas marked by a gray dotted line, magenta triangle marks evaluation points made by candidate points generation in all the feasible region, and blue square marks evaluation points made during surrogate function optimization using the Hooke–Jeeves algorithm.

approximated by Pareto optimal solutions found by the proposed algorithm in both test problems (Figs. 2(a) and 2(c)). Function evaluation points made by global search tend to explore the area with no function evaluations (triangle points) and to cluster new function evaluation points near current Pareto optimal solutions (circle points) (Figs. 2(b) and 2(d)). Function evaluation points made by local refinement (square points) are well clustered near the line of Pareto optimal solutions in decision space (Figs. 2(b) and 2(d)).

3.3 Numerical experiments using test problems having many decision variables

In the case of the high dimensionality of the feasible region, the results of the proposed algorithm were compared with the results of the evolutionary optimization algorithm NSGA2 [3, 8]. The performance of the optimization algorithm was tested with the biobjective test suite ZDT having many decision variables. Biobjective test problems ZDT1, ZDT2, ZDT3 have 30 decision variables, and ZDT4, ZDT6 have 10 decision variables [33].

For the comparison performance of the proposed algorithm in the case of the high dimensionality of the feasible region, the metrics applied which were used in publications related to the evolutionary optimization algorithm NSGA2 [3, 8]. The estimation of the average distance between the found approximation and the true Pareto front is the average generational distance ($\mathrm{GD}_{\mathrm{avg}}$) [8]. $\mathrm{GD}_{\mathrm{avg}}$ is computed as the average of distances between the found non-dominated solutions and their closest neighbors from the Pareto front. A metric integrating measures of the approximation precision and spread is so-called inverted generational distance ($\mathrm{IGD}_{\mathrm{avg}}$) [3]. $\mathrm{IGD}_{\mathrm{avg}}$ is computed as the average of distances between the true Pareto front and their closest neighbors from found non-dominated solutions. Metrics $\mathrm{GD}_{\mathrm{avg}}$ and $\mathrm{IGD}_{\mathrm{avg}}$ can be expressed by the following equations:

$$\mathrm{GD}_{\mathrm{avg}} = \frac{1}{|P|} \sum\nolimits_{F_P \in P} \min_{F_{P^*} \in P^*} \|F_P - F_{P^*}\|, \tag{5$_1$}$$

$$\mathrm{IGD}_{\mathrm{avg}} = \frac{1}{|P^*|} \sum\nolimits_{F_{P^*} \in P^*} \min_{F_P \in P} \|F_P - F_{P^*}\|, \tag{5$_2$}$$

where P is a set of non-dominated solutions found by the considered algorithm, and P^* is a set of solutions well representing the Pareto front, i.e., the solutions are sufficiently densely and uniformly distributed over the Pareto front, i.e., $|P^*| = 500$ uniformly distributed points were used as a set of solutions well representing the Pareto front.

The proposed algorithm was tested with fixed function $F(x)$ evaluation budget. The parameter $N_{\max}$ value is selected 10% below the function evaluation budget, so on average the function evaluation budget is not exceeded. The proposed algorithm was run with the following parameters:

$$(I_{\max}, N, q, p, h_o, h_n, \mathit{Update})^{\mathrm{T}} = (\infty, 100, 1, 0.8, 2, 8, \mathit{true})^{\mathrm{T}}.$$

The number of iteration $I_{\max} = \infty$ is selected to be the maximum number which cannot be achieved. The number of initial random function evaluations is $N = 100$, and the

Table 2. Mean values and variance of performance criteria $\mathrm{GD}_{\mathrm{avg}}$ of the real-coded and binary-coded evolutionary optimization algorithm NSGA2 and proposed algorithm for ZDT test problems.

Algorithm	Problem				
	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
NSGA2 real-coded	0.033482	0.072391	0.114500	0.513053	0.296564
	0.004750	0.031689	0.007940	0.118460	0.013135
NSGA2 binary-coded	0.000894	0.000824	0.043411	3.227636	7.806798
	0	0	0.000042	7.307630	0.001667
Proposed algorithm	0.000774	0.000841	0.000910	0.069231	0.005558
	$1.843 \cdot 10^{-9}$	$1.229 \cdot 10^{-8}$	$7.294 \cdot 10^{-10}$	0.004069	$2.831 \cdot 10^{-5}$

Table 3. Mean values and standard deviation of function evaluation count and performance criteria $\mathrm{IGD}_{\mathrm{avg}}$ of evolutionary optimization algorithm NSGA2 and the proposed algorithm for ZDT test problems.

Problem	Algorithm							
	NSGA2				Proposed algorithm			
	μ_{eval}	σ_{eval}	$\mu_{\mathrm{IGD}_{\mathrm{avg}}}$	$\sigma_{\mathrm{IGD}_{\mathrm{avg}}}$	μ_{eval}	σ_{eval}	$\mu_{\mathrm{IGD}_{\mathrm{avg}}}$	$\sigma_{\mathrm{IGD}_{\mathrm{avg}}}$
ZDT1	17098	2393.78	0.006	0.0007	15344	29.24	0.004	0.0021
ZDT2	17657	1528.04	0.006	0.0008	15867	32.51	0.007	0.0031
ZDT3	16559	1899.07	0.007	0.0078	14911	36.83	0.003	0.0013
ZDT4	24451	2864.01	0.006	0.0013	22045	232.90	0.104	0.0431
ZDT6	24833	1084.74	0.004	0.0002	22336	10.41	0.003	0.0017

coefficient value of number $q \cdot N$ of randomly generated candidate points in decision space for new function evaluations is $q = 1$. The number of randomly generated candidate points should not be high in the case of a large functions evaluation budget as it cause a great burden for the proposed algorithm, so a low coefficient value of q is selected. The value of $p = 0.8$ is part of function evaluations get by local generation near current Pareto optimal solutions compared to function evaluations get by generation in the entire feasible region. The step size parameters of the Hooke–Jeeves optimization algorithm have the following values: $h_0 = 2$ and $h_n = 8$. The step size parameters h_0 and h_n are updated at second and following iterations of global search with local refinement as parameter $Update$ is set to true. The same parameters were used for all ZDT test problems.

Since the proposed algorithm and evolutionary optimization algorithm NSGA2 are stochastic, the test problems were solved multiple times [3, 8]. The mean values and variance or standard deviations of the considered metrics are presented in Tables 2 and 3. Note that in this subsection the normalization of function values is performed with ideal and nadir points in the case of $\mathrm{IGD}_{\mathrm{avg}}$ metric calculation; see Eq. (2). Table 2 shows $\mathrm{GD}_{\mathrm{avg}}$ metric results of the proposed algorithm compared to the results of the evolutionary optimization algorithm NSGA2 after 10 experiments having 25000 function evaluation budget [8]. Table 3 shows $\mathrm{IGD}_{\mathrm{avg}}$ metric results of the proposed algorithm compared to the results of the evolutionary optimization algorithm NSGA2 after 51 experiments [3].

The proposed algorithm shows good performance. Table 2 shows $\mathrm{GD}_{\mathrm{avg}}$ metric results; the proposed algorithm has better performance compared to the results of the evolutionary optimization algorithm NSGA2. The proposed algorithm has a lower $\mathrm{GD}_{\mathrm{avg}}$ metric value compared to NSGA2 except in the case of the ZDT2 test problem, where the

binary-coded NSGA2 performs better. Table 3 shows $\mathrm{IGD}_{\mathrm{avg}}$ metric results; the proposed algorithm has better performance compared to the results of the evolutionary optimization algorithm NSGA2 in the case of ZDT1, ZDT3 and ZDT6 test problems. The proposed algorithm has worse performance compared to the results of the evolutionary optimization algorithm NSGA2 in the case of ZDT2 and ZDT4 test problems.

3.4 Numerical experiments using many-objective test problems

Many-objective optimization problems having up to 15 objective functions ($M \leqslant 15$) are considered. Representation of Pareto optimal surface for many-objective optimization problems is usually difficult as representation needs exponentially growing points count for the additional objective function. Multiple predefined reference points can be specified, and Pareto-optimal trade-off solutions corresponding to each reference point are found as it is done in the genetic algorithm NSGA3 [7]. In the case of many-objective optimization problems, adequate trade-off Pareto optimal surface approximation may not be found using a few hundred reference points [7]. Reference points selection may be tricky if trade-off solutions with specific properties are needed, so the newly developed optimization algorithm does not need any predefined reference points. The newly developed optimization algorithm was tested with many-objective test suite DTLZ having up to 15 objective functions ($M \leqslant 15$) [9]. The number of variables is $M + k - 1$, where number $k = 5$ is for DTLZ1, while number $k = 10$ is for DTLZ2, DTLZ3 and DTLZ4 test problems. The results were compared with the results of popular evolutionary optimization algorithms NSGA3 [7].

The proposed algorithm was tuned to find raw solutions since having moderate function evaluation budget an adequate representation of Pareto optimal surface for many-objective optimization problems usually is impossible. The parameter $N_{\max}$ value is selected to be very large, it can be noted $N_{\max} = \infty$, so the number of iteration $I_{\max}$ is specified as a stop condition in Table 4. The proposed algorithm was run with the following parameters:

$$(N, q, p, h_o, h_n, \mathit{Update})^{\mathrm{T}} = (100, 1, 0.8, 2, 6, \mathit{false})^{\mathrm{T}}.$$

The number of initial random function evaluations is $N = 100$, and coefficient value of number $q \cdot N$ of randomly generated candidate points in decision space for new function evaluations is $q = 1$. Value of $p = 0.8$ is part of function evaluations get by local generation near current Pareto optimal solutions compared to function evaluations get by generation in all the feasible region. The step size parameters of the Hooke–Jeeves optimization algorithm have the following values: $h_0 = 2$ and $h_n = 6$. Raw solutions are searched, so step size parameters h_0 and h_n update can drastically increase function evaluation budget, so parameters are not updated at second and following iterations since the parameter Update is set to false.

After a raw trade-off solutions $\{P, P_A\}$ is found, random solutions $\{P', P'_A\}$ are selected to be tuned by the surrogate function optimization using the Hooke–Jeeves algorithm. Table 5 shows how many raw solutions are taken for tuning using surrogate function optimization. For every random solution $x_P \in P'_A$, a separate surrogate function having

Table 4. Mean values of the function evaluation count and the best, median and worst IGD_{ref} values obtained for proposed algorithm and NSGA3 on M-objective DTLZ test problems. The number of iterations $I_{\max}$ is specified as the stop condition of the proposed algorithm.

NSGA3		Proposed algorithm			NSGA3		Proposed algorithm		
μ_{eval}	$\mathrm{IGD}_{\mathrm{ref}}$	μ_{eval}	$\mathrm{IGD}_{\mathrm{ref}}$	$I_{\max}$	μ_{eval}	$\mathrm{IGD}_{\mathrm{ref}}$	μ_{eval}	$\mathrm{IGD}_{\mathrm{ref}}$	$I_{\max}$
DTLZ1, $M=3$					DTLZ2, $M=3$				
36800	$4.880\cdot10^{-4}$	30898	$1.910\cdot10^{-4}$	2	23000	$1.262\cdot10^{-3}$	29495	$5.116\cdot10^{-5}$	1
	$1.308\cdot10^{-3}$		$4.566\cdot10^{-4}$			$1.357\cdot10^{-3}$		$5.806\cdot10^{-5}$	
	$4.880\cdot10^{-3}$		$8.452\cdot10^{-4}$			$2.114\cdot10^{-3}$		$1.495\cdot10^{-4}$	
DTLZ1, $M=5$					DTLZ2, $M=5$				
127200	$5.116\cdot10^{-4}$	93741	$1.691\cdot10^{-4}$	2	74200	$4.254\cdot10^{-3}$	77751	$5.184\cdot10^{-5}$	1
	$9.799\cdot10^{-4}$		$4.188\cdot10^{-4}$			$4.982\cdot10^{-3}$		$5.705\cdot10^{-5}$	
	$1.979\cdot10^{-3}$		$5.985\cdot10^{-4}$			$5.862\cdot10^{-3}$		$6.481\cdot10^{-5}$	
DTLZ1, $M=8$					DTLZ2, $M=8$				
117000	$2.044\cdot10^{-3}$	86933	$2.568\cdot10^{-4}$	1	78000	$1.371\cdot10^{-2}$	82594	$5.331\cdot10^{-5}$	1
	$3.979\cdot10^{-3}$		$4.717\cdot10^{-4}$			$1.571\cdot10^{-2}$		$5.708\cdot10^{-5}$	
	$8.721\cdot10^{-3}$		$2.939\cdot10^{-3}$			$1.811\cdot10^{-2}$		$6.230\cdot10^{-5}$	
DTLZ1, $M=10$					DTLZ2, $M=10$				
276000	$2.215\cdot10^{-3}$	159425	$2.137\cdot10^{-4}$	1	207000	$1.350\cdot10^{-2}$	132556	$5.211\cdot10^{-5}$	1
	$3.462\cdot10^{-3}$		$4.553\cdot10^{-4}$			$1.528\cdot10^{-2}$		$5.715\cdot10^{-5}$	
	$6.869\cdot10^{-3}$		$1.325\cdot10^{-3}$			$1.697\cdot10^{-2}$		$5.988\cdot10^{-5}$	
DTLZ1, $M=15$					DTLZ2, $M=15$				
204000	$2.649\cdot10^{-3}$	128085	$1.165\cdot10^{-4}$	1	136000	$1.360\cdot10^{-2}$	115563	$5.146\cdot10^{-5}$	1
	$5.063\cdot10^{-3}$		$4.679\cdot10^{-4}$			$1.726\cdot10^{-2}$		$5.738\cdot10^{-5}$	
	$1.123\cdot10^{-2}$		$2.495\cdot10^{-3}$			$2.114\cdot10^{-2}$		$6.189\cdot10^{-5}$	
DTLZ3, $M=3$					DTLZ4, $M=3$				
92000	$9.751\cdot10^{-4}$	62713	$1.373\cdot10^{-3}$	2	55200	$2.915\cdot10^{-4}$	65811	$5.745\cdot10^{-5}$	3
	$4.007\cdot10^{-3}$		$2.250\cdot10^{-3}$			$5.970\cdot10^{-4}$		$6.165\cdot10^{-5}$	
	$6.665\cdot10^{-3}$		$3.922\cdot10^{-3}$			$4.286\cdot10^{-1}$		$6.607\cdot10^{-5}$	
DTLZ3, $M=5$					DTLZ4, $M=5$				
212000	$3.086\cdot10^{-3}$	183641	$1.382\cdot10^{-3}$	2	212000	$9.849\cdot10^{-4}$	210900	$5.430\cdot10^{-5}$	2
	$5.960\cdot10^{-3}$		$1.939\cdot10^{-3}$			$1.255\cdot10^{-3}$		$5.782\cdot10^{-5}$	
	$1.196\cdot10^{-2}$		$3.132\cdot10^{-3}$			$1.721\cdot10^{-3}$		$8.157\cdot10^{-5}$	
DTLZ3, $M=8$					DTLZ4, $M=8$				
156000	$1.244\cdot10^{-2}$	153307	$2.597\cdot10^{-3}$	1	195000	$5.079\cdot10^{-3}$	94550	$4.280\cdot10^{-5}$	1
	$2.375\cdot10^{-2}$		$4.739\cdot10^{-3}$			$7.054\cdot10^{-3}$		$5.131\cdot10^{-5}$	
	$9.649\cdot10^{-2}$		$1.517\cdot10^{-2}$			$6.051\cdot10^{-1}$		$7.030\cdot10^{-5}$	
DTLZ3, $M=10$					DTLZ4, $M=10$				
414000	$8.849\cdot10^{-3}$	267519	$2.387\cdot10^{-3}$	1	552000	$5.694\cdot10^{-3}$	150992	$4.770\cdot10^{-5}$	1
	$1.188\cdot10^{-2}$		$5.396\cdot10^{-3}$			$6.337\cdot10^{-3}$		$5.404\cdot10^{-5}$	
	$2.083\cdot10^{-2}$		$1.682\cdot10^{-2}$			$1.076\cdot10^{-1}$		$7.262\cdot10^{-5}$	
DTLZ3, $M=15$					DTLZ4, $M=15$				
272000	$1.401\cdot10^{-2}$	201526	$3.628\cdot10^{-3}$	1	408000	$7.110\cdot10^{-3}$	141192	$4.687\cdot10^{-5}$	1
	$2.145\cdot10^{-2}$		$8.220\cdot10^{-3}$			$3.431\cdot10^{-1}$		$5.450\cdot10^{-5}$	
	$4.195\cdot10^{-2}$		$2.970\cdot10^{-2}$			1.073		$1.096\cdot10^{-4}$	

Table 5. Number of reference points used in NSGA3 and raw solutions used in proposed algorithm.

Number of objectives (M)	Reference points	Raw solutions
3	91	91
5	210	210
8	156	156
10	275	275
15	135	135

current solution $x_{\text{cur}} = x_P$ is defined. Every defined surrogate function is optimized using the Hooke–Jeeves optimization algorithm taking x_P as the start point:

$$\min f_s(x), \quad x \in A \subset \mathbb{R}^d,$$

where step size parameters are set to $h_0 = 2$ and $h_n = 12$. All test problems (Table 4) were optimized with the same parameters, except in the case of DTLZ2 and DTLZ4 the proposed algorithm's step size parameters were set to $h_0 = 2$ and $h_n = 4$, and step size parameters of raw solutions tuning phase was set to $h_0 = 2$ and $h_n = 6$ to have larger step sizes, and therefore function evaluation budget is of similar size with NSGA3. Also, in the case of M-objective DTLZ4 test problems having $M = 3$ and $M = 5$ objective functions, a number of initial random function evaluations was set to $N = 3000$, and coefficient value of number $q \cdot N$ of randomly generated candidate points was set to $q = 0.03$. This way enough raw solutions was found, and the needed count of raw solutions is given in Table 5.

For the comparison performance of the proposed algorithm in the case of a many-objective optimization problem, the metrics was applied which was used in publications related to the evolutionary optimization algorithm NSGA3 [7]. A metric similar to inverted generational distance IGD_{avg} defined by (5) equation, but instead of computing the average of distances between the true Pareto front and their closest neighbors from found non-dominated solutions, a new metric IGD_{ref} compute average of distances between reference points on the true Pareto front and their closest neighbors from found non-dominated solutions. Metric IGD_{ref} can be expressed by the following equation:

$$\text{IGD}_{\text{ref}} = \frac{1}{|P^*|} \sum\nolimits_{F_{P^*} \in P^*} \min_{F_P \in P_{\text{tuned}}} \|F_P - F_{P^*}\|,$$

where P_{tuned} is non-dominated solutions tuned to reference points, P^* is a set of reference points on the true Pareto front. In the case of NSGA3 a multiple predefined Pareto-optimal reference points P^* are specified, and Pareto-optimal trade-off solutions corresponding to each reference point are found [7]. In the case of the newly developed optimization algorithm a predefined reference points selection is not needed. Raw solutions after final iterations of the proposed algorithm are randomly selected $\{P', P'_A\}$, and closest points on the true Pareto front are selected as a set of reference points P^*. Selected non-dominated raw solutions $\{P', P'_A\}$ are tuned by the surrogate function optimization using the Hooke–Jeeves algorithm to get a non-dominated solution set P_{tuned}. As raw solutions are selected

randomly $\{P', P'_A\}$, so this is equivalent to raw solutions selection by a human expert to get solutions with specific properties. The sizes of sets of raw solutions $\{P', P'_A\}$ and reference points P^* are given in Table 5.

Since the proposed algorithm and evolutionary optimization algorithm NSGA3 are stochastic, the test problems were solved 20 times [7]. The best median and worst values of the considered metric $\mathrm{IGD}_{\mathrm{ref}}$ are presented in Table 4. The results of the proposed algorithm were compared to the results of the evolutionary optimization algorithm NSGA3 [7]. The proposed algorithm gives a good performance compared to the results of the evolutionary optimization algorithm NSGA3. Mean values of function evaluation count are lower in the case of the proposed algorithm except in the case of M-objective DTLZ2 ($M = 3, 5, 8$) and DTLZ4 ($M = 3$) test problems. The best median and worst IGD_{ref} values are better in the case of the proposed algorithm except in case of the M-objective DTLZ3 ($M = 3$) test problem.

4 Conclusions

The hybrid multi-objective optimization algorithm is proposed combining random global search and local refinement of the found approximation of the Pareto front. The global search algorithm mimics the Bayesian algorithm. The Hooke–Jeeves algorithm is used for local refinement. At the local optimization phase, the multi-objective optimization problem is converted to a single-objective optimization problem by introducing a surrogate function without the use of the weight vectors. The developed algorithm was tested in the case of extremely low functions evaluation budget, and the proposed algorithm gives decent performance comparing results with the results of Bayesian rooted optimization algorithms. Also, the proposed optimization algorithm was tested with many decision variables test suite and with many-objective test suite, and the results of numerical experiments showed good performance compared with the results of popular evolutionary optimization algorithms NSGA2 and NSGA3. In the case of many-objective optimization, the proposed algorithm need no predefined reference points, so raw solutions selection can be done by a human expert to be tuned to final solutions of needed properties. Future plans include the development parallel version of the proposed algorithm to optimize functions with an extreme computational burden. Another research theme of interest is integrating the proposed algorithm execution with human expert decisions to get solutions of needed properties.

References

1. F. Archetti, A. Candelieri, *Bayesian Optimization and Data Science*, Springer, Cham, 2019.
2. R. Baronas, A. Žilinskas, L. Litvinas, Optimal design of amperometric biosensors applying multi-objective optimization and design visualization, *Electrochim. Acta*, **211**:586–594, 2016.
3. J. Blank, K. Deb, A running performance metric and termination criterion for evaluating evolutionary multi- and many-objective optimization algorithms, in *2020 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Piscataway, NJ, 2020, pp. 1–8.

4. J. Cui, B. Yang, Survey on Bayesian optimization methodology and applications, *J. Softw.*, **29**(10):3068–3090, 2007 (in Chinese), https://doi.org/10.13328/j.cnki.jos.005607.
5. A.L. Custódio, J.F.A. Madeira, MultiGLODS: global and local multiobjective optimization using direct search, *J. Glob. Optim.*, **72**(2):1–23, 2018, https://doi.org/10.1007/s10898-018-0618-1.
6. K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, Wiley, New York, 2001.
7. K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints, *IEEE Transactions on Evolutionary Computation*, **18**(4):577–601, 2014, https://doi.org/10.1109/TEVC.2013.2281535.
8. K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.*, **6**(2):182–197, 2002, https://doi.org/10.1109/4235.996017.
9. K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in A. Abraham, L. Jain, R. Goldberg (Eds.), *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Springer, London, 2005, pp. 105–145, https://doi.org/10.1007/1-84628-137-7_6.
10. C.M. Fonseca, P.J. Fleming, On the performance assessment and comparison of stochastic multiobjective optimizers, in H.-M. Voigt, W. Ebeling, I. Rechenberg, H.-P. Schwefel (Eds.), *Parallel Problem Solving from Nature—PPSN IV*, Springer, Berlin, Heidelberg, 1996, pp. 584–593, https://doi.org/10.1007/3-540-61723-X_1022.
11. V. Gergel, E. Kozinov, Efficient multicriterial optimization based on intensive reuse of search information, *J. Glob. Optim.*, **71**(1):73–90, 2018, https://doi.org/10.1007/s10898-018-0624-3.
12. D.R. Jones, C.D. Perttunen, B.E.Stuckman, Lipschitzian optimization without the Lipschitz constant, *J. Optim. Theory Appl.*, **79**:157–181, 1993, https://doi.org/10.1007/BF00941892.
13. C.T. Kelley, *Iterative Methods for Optimization*, SIAM, Philadelphia, 1999.
14. L. Litvinas, R. Baronas, A. Žilinskas, Application of two phase multi-objective optimization to design of biosensors utilizing cyclic substrate conversion, in *Proceedings of the 31st European Conference on Modelling and Simulation, ECMS 2017*, European Council for Modelling and Simulation, 2017, pp. 469–474, https://doi.org/10.7148/2017-0469.
15. K. Miettinen, *Nonlinear Multiobjective Optimization*, Springer, Boston, MA, 1998, https://doi.org/10.1007/978-1-4615-5563-6.
16. P.M. Pardalos, A. Žilinskas, J. Žilinskas, *Non-Convex Multi-Objective Optimization*, Springer, Cham, 2017.
17. R. Paulavičius, Y.D. Sergeyev, D.E. Kvasov, J. Žilinskas, Globally-biased DISIMPL algorithm for expensive global optimization, *J. Glob. Optim.*, **59**:1–23, 2014, https://doi.org/10.1007/s10898-014-0180-4.
18. R. Paulavičius, Y.D. Sergeyev, D.E. Kvasov, J. Žilinskas, Globally-biased BIRECT algorithm with local accelerators for expensive global optimization, *Expert Syst. Appl.*, **144**:113052, 2020, https://doi.org/10.1016/j.eswa.2019.113052.

19. R. Paulavičius, J. Žilinskas, *Simplicial Global Optimization. Springer Briefs in Optimization*, Springer, New York, 2014, https://doi.org/10.1007/978-1-4614-9093-7.

20. J. Pinter, *Global Optimization in Action*, Springer, Boston, MA, 1996, https://doi.org/10.1007/978-1-4757-2502-5.

21. Y.D. Sergeyev, D.E. Kvasov, Global search based on efficient diagonal partitions and a set of Lipschitz constants, *SIAM J. Optim.*, **16**(3):910–937, 2006, https://doi.org/10.1137/040621132.

22. Y.D. Sergeyev, M.S. Mukhametzhanov, D.E. Kvasov, D. Lera, Derivative-free local tuning and local improvement techniques embedded in the univariate global optimization, *J. Optim. Theory Appl.*, **171**(1):186–208, 2016, https://doi.org/10.1007/s10957-016-0947-5.

23. A. Žilinskas, On the worst-case optimal multi-objective global optimization, *Optim. Lett.*, **7**:1921–1928, 2013, https://doi.org/10.1007/s11590-012-0547-8.

24. A. Žilinskas, J. Calvin, Bi-objective decision making in global optimization based on statistical models, *J. Glob. Optim.*, **74**:599–609, 2019, https://doi.org/10.1007/s10898-018-0622-5.

25. A. Žilinskas, G. Gimbutienė, A hybrid of Bayesian approach based global search with clustering aided local refinement, *Commun. Nonlinear Sci. Numer. Simul.*, **78**:104857, 2019, https://doi.org/10.1016/j.cnsns.2019.104857.

26. C. Zheng, J. Calvin, C. Gotsman, A DIRECT-type global optimization algorithm for image registration, *J. Glob. Optim.*, **79**(2):431–445, 2021, https://doi.org/10.1007/s10898-020-00914-y.

27. A. Zhigljavsky, A. Žilinskas, *Bayesian and High-Dimensional Stochastic Optimization*, Springer, Cham, 2021, https://doi.org/10.1007/978-3-030-64712-4.

28. A. Žilinskas, A statistical model-based algorithm for 'black-box' multi-objective optimisation, *Int. J. Syst. Sci., Princ. Appl. Syst. Integr.*, **45**(1):82–93, 2014, https://doi.org/10.1080/00207721.2012.702244.

29. A. Žilinskas, R. Baronas, L. Litvinas, L. Petkevičius, Multi-objective optimization and decision visualization of batch stirred tank reactor based on spherical catalyst particles, *Nonlinear Anal. Model. Control*, **24**(6):1019–1033, 2019, https://doi.org/10.15388/NA.2019.6.10.

30. A. Žilinskas, L. Litvinas, A hybrid of the simplicial partition-based Bayesian global search with the local descent, *Soft Comput.*, **24**(23):17601–17608, 2020, https://doi.org/10.1007/s00500-020-05095-0.

31. A. Žilinskas, L. Litvinas, A partition based Bayesian multi-objective optimization algorithm, in Y.D. Sergeyev, D.E. Kvasov (Eds.), *Numerical Computations: Theory and Algorithms*, Springer, Cham, 2020, pp. 511–518, https://doi.org/10.1007/978-3-030-40616-5_50.

32. A. Žilinskas, J. Žilinskas, Adaptation of a one-step worst-case optimal univariate algorithm of bi-objective Lipschitz optimization to multidimensional problems, *Commun. Nonlinear Sci. Numer. Simul.*, **21**(1–3):89–98, 2015, https://doi.org/10.1016/j.cnsns.2014.08.025.

33. E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, *Evol. Comput.*, **8**(2):173–195, 2000, https://doi.org/10.1162/106365600568202.

34. E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, *IEEE Trans. Evol. Comput.*, **7**(2):117–132, 2003, https://doi.org/10.1109/TEVC.2003.810758.