

# A Local Search for Numerical Optimisation based on Covariance Matrix Diagonalisation

Ferrante Neri<sup>1</sup>[0000-0002-6100-6532] and Shahin Rostami<sup>2</sup>[0000-0001-8977-1239]

<sup>1</sup> Computational Optimisation and Learning (COL) Lab,  
School of Computer Science, University of Nottingham, United Kingdom  
`ferrante.neri@nottingham.ac.uk`

<sup>2</sup> Department of Computing and Informatics,  
Bournemouth University, United Kingdom  
`srostami@bournemouth.ac.uk`

**Abstract.** Pattern Search is a family of optimisation algorithms that improve upon an initial solution by performing moves along the directions of a basis of vectors. In its original definition Pattern Search moves along the directions of each variable. Amongst its advantages, the algorithm does not require any knowledge of derivatives or analytical expression of the function to optimise. However, the performance of Pattern Search is heavily problem dependent since the search directions can be very effective on some problems and lead to poor performance on others. The present article proposes a novel enhancement of Pattern Search that explores the space by using problem-dependent search directions. Some points are sampled within the basin of attraction and the diagonalisation of the covariance matrix associated with the distribution of these points is performed. The proposed Covariance Pattern Search improves upon an initial point by varying it along the directions identified by the eigenvectors of the covariance matrix.

**Keywords:** Hooke-Jeeves · Pattern Search · Covariance Matrix · Eigenvectors · Numerical Optimisation.

## 1 Introduction

In the context of optimisation, a basin of attraction of a search algorithm is the set of points of the decision space that would converge to the same local optimum, see [15]. For example, if we consider a multimodal problem in the continuous domain, a gradient based algorithm would process some randomly sampled solutions by converging to the nearest local optimum. If two points converge to the same optimum, they belong to the same basin of attraction with respect to the gradient based optimiser. Thus, for a given search algorithm (with its variation operator) a decision space can be seen as mapped into several (and possibly overlapping) subsets, with each of them being a basin of attraction.

This concept is fundamental in optimisation, especially when complex, multivariate, and multimodal fitness landscapes are taken into consideration, see

[19, 31, 28, 1]. A popular strategy to address these issues is the use of multiple search operators within the same framework. This idea has been extensively applied over the past three decades in both continuous and combinatorial domains. Some of the keywords associated with this idea are Metaheuristics [9, 25], Hyper-heuristics [2], Memetic Computing [19], Genetic Programming [11], Agent Systems [24], and Algorithm's Portfolio [33, 22].

In these frameworks a distinction between global and local search operators has been traditionally presented as part of the nomenclature to describe algorithmic operations. Whilst the distinction between global and local optima is mathematically rigorous, the distinction between global and local search is more subtle, especially in the context of heuristic optimisation where there is no theoretical guarantee of the detection of an optimum. More specifically, global and local search algorithms differ in purpose: whilst global search algorithms search for a candidate solution with the lowest (or highest) function value within the entire decision space, local search algorithms search for a candidate solution with the lowest (or highest) function value within a subset of the entire space, often referred to as a neighbourhood, see [10].

In mathematical optimisation, and with reference to the continuous domain that is the focus of this paper, the concepts of basins of attraction and local search are well-defined. When considering a minimisation problem a basin of attraction is a set containing one or more infinite contiguous (saddle) points with null gradient (unless the optimum is on the bounds), surrounded by points with non-null gradient and higher function values. Local search algorithms that calculate the gradient, e.g. descent methods, Newtonian methods, Quasi-Newtonian methods, and conjugate gradient methods, would (approximately) detect the local optimum in a given time-frame from any starting point in the basin of attraction, see [21].

When the derivatives are not available and a heuristic method is applied, the concepts of basins of attraction and local search become unclear. For example, the Nelder-Mead algorithm [18], which is often considered a local search algorithm [17], can search for the optimum in an area with only one local optimum, a larger area, or potentially the entire decision space (depending on the initial simplex).

In the absence of derivatives, one of the simplest ideas to optimise a function is to vary one design variable at a time by steps of the same magnitude and calculate the function value at each step. Then, when no increase or decrease in any one design variable improves upon the performance of the current best solution, the algorithm halves the step size and repeats the process until the steps are sufficiently small. This simple algorithm was employed in the 1940s in Los Alamos laboratories by Fermi and Metropolis, see [8]. This idea has been refined and modified over the decades, and its most famous implementation is the **Pattern Search** (PS) algorithm by Hooke and Jeeves [14] which proved to be convergent in [34]. This idea has been conceptualised in [29] where Pattern Search Algorithms are introduced as a family of optimisation algorithms, and the term Generalised Pattern Search was coined as a Pattern Search using any

generic basis of a vector. The search logic of PS is used as part of many modern algorithms. One example is [30] where a greedy version of the Pattern Search is used within a three algorithm portfolio for large scale problems. Furthermore, other recent examples of Pattern Search implementations have re-named the algorithm as ‘‘S’’ and put it into the context of Memetic Algorithms [5, 7] and in a restarting scheme in [6].

The present article exploits this idea to propose a novel enhancement of Generalised Pattern Search. The proposed algorithm attempts to enhance the original scheme by using an alternative and more convenient reference system to move within the search space. This alternative system is provided by the eigenvectors of a covariance matrix of a set of points crowding a basin of attraction.

The remainder of this article is organised as follows. Section 2 introduces the notation, presents the naive Pattern Search and highlights its limitations. Section 3 presents the proposed algorithm including its theoretical justification, as well as its limitations. Section 4 provides the numerical results of this study. Finally, Section 5 gives the conclusions to this work.

## 2 Notation and background

In order to clarify the notation used, we refer to the minimisation of an objective function  $f(\mathbf{x})$ , where the candidate solution  $\mathbf{x}$  is a vector of  $n$  design variables in a decision space  $D \subset \mathbb{R}^n$ :

$$\mathbf{x} = (x_1, x_2, \dots, x_n).$$

The Pattern Search (PS) processes a single solution and, whilst moving along the axes, improves upon its performance (objective function value). PS can be viewed as a simple deterministic local search algorithm which can be part of a more complex framework, see [3, 5].

In this paper we will refer to the naive implementation of this idea proposed in one of the searchers in [30]. Starting from an elite solution  $\mathbf{x}$ , this local search generates a trial solution  $\mathbf{x}^t$  by performing steps along each of the variables. For each design variable  $i$ ,

$$\mathbf{x}^t = \mathbf{x} - \rho \cdot \mathbf{e}^i$$

is calculated, where  $\rho$  is the exploratory radius,  $\cdot$  indicates the product of a scalar and a vector, and  $\mathbf{e}^i$  is the  $i^{th}$  versor that is a vector composed of zeros and one 1 in the  $i^{th}$  position. Subsequently, if  $\mathbf{x}^t$  outperforms  $\mathbf{x}$ , the trial solution  $\mathbf{x}^t$  is updated (taking the value of  $\mathbf{x}$ ), otherwise a half-step in the opposite direction is performed:

$$\mathbf{x}^t = \mathbf{x} + \frac{\rho}{2} \cdot \mathbf{e}^i.$$

This exploration is repeated for all the design variables and stopped when a prefixed budget is exceeded. For the purpose of this paper we will refer to the naive implementation used in [30, 5]. The pseudo-code displaying the working principles of this PS implementation is given in Algorithm 1.

---

**Algorithm 1** Pattern Search according to the implementation in [30]

---

```

INPUT  $\mathbf{x}$ 
while local budget condition do
   $\mathbf{x}^t = \mathbf{x}$ 
  for  $i = 1 : n$  do
     $\mathbf{x}^t = \mathbf{x} - \rho \cdot \mathbf{e}^i$ 
    if  $f(\mathbf{x}^t) \leq f(\mathbf{x})$  then
       $\mathbf{x} = \mathbf{x}^t$ 
    else
       $\mathbf{x}^t = \mathbf{x} + \frac{\rho}{2} \cdot \mathbf{e}^i$ 
      if  $f(\mathbf{x}^t) \leq f(\mathbf{x})$  then
         $\mathbf{x} = \mathbf{x}^t$ 
      end if
    end if
  end for
  if  $\mathbf{x}$  has not been updated then
     $\rho = \frac{\rho}{2}$ 
  end if
end while
RETURN  $\mathbf{x}$ 

```

---

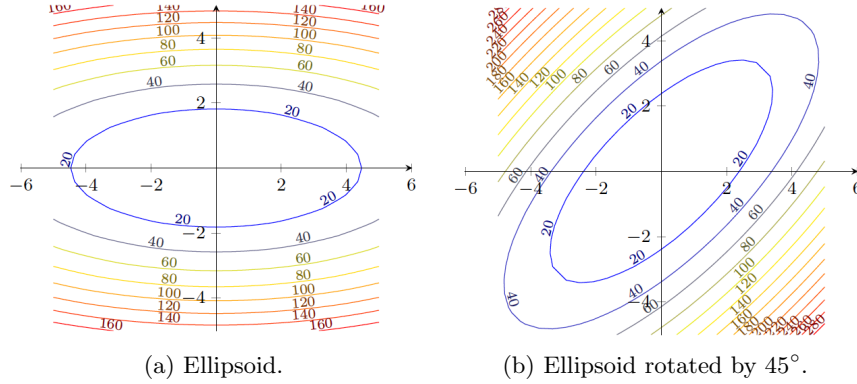
## 2.1 Limitation of Pattern Search

The naive PS algorithm, albeit easy to implement, is characterised by heavy problem dependent performance due to its variation operators. Since the variables are perturbed/modified one by one, PS can lead to very good performance if the function is separable. In practice, PS can be successfully applied to various partly separable and non-separable problems (see [7]), but its performance would generally be poor. In order to illustrate this statement let us consider the ellipsoid function in two dimensions shown in Figures 1. The PS algorithm would quickly solve the problem in Fig. 1a from any starting point, since one of the directions of the search (one of the axes) coincide with the direction of maximum gradient. However, the PS algorithm would not be efficient in optimising the problem in Fig. 1b, since the search directions are not along the maximum gradient. The algorithm would halve the radius in the early stages of the search and move slowly towards the optimum. In the case of optimisation in higher dimensions the difference in performance between the two scenarios would be significant.

## 3 The proposed Covariance Pattern Search

If we could move along the direction of maximum gradient for every problem, PS would work efficiently regardless of the problem. Unfortunately, in real-world optimisation, problems are unknown and there is no prior knowledge of the most convenient reference system. This consideration is not novel and is common to several algorithms for numerical optimisation, such as the Rosenbrock Algorithm, see [27], where the gradient is estimated adaptively and by a new reference system determined by Gram-Schmidt orthogonalisation.

On the basis of this classical consideration, the present article proposes a novel local search implementation of Pattern Search. This section outlines and



**Fig. 1.** Non-rotated and 45° rotated ellipsoid in two dimensions: PS would be efficient at solving 1a and inefficient at solving 1b.

discusses the proposal. Subsection 3.1 provides a theoretical justification for the method, Subsection 3.2 describes the implementation details of the methods, and Subsection 3.4 highlights the limitations of our proposal.

### 3.1 Theoretical justification

Unlike the Rosenbrock Algorithm, the proposed method makes use of pre-processing of the problem under investigation to determine the most convenient search directions. In order to understand the theoretical foundation of the proposed method let us consider a population of  $m$  vectors/candidate solutions in an  $n$ -dimensional space

$$\begin{aligned} \mathbf{x}^1 &= (x_1^1, x_2^1, \dots, x_n^1) \\ \mathbf{x}^2 &= (x_1^2, x_2^2, \dots, x_n^2) \\ &\dots \\ \mathbf{x}^m &= (x_1^m, x_2^m, \dots, x_n^m). \end{aligned}$$

These points can be interpreted as a statistical distribution characterised by a mean vector

$$\mu = (\mu_1, \mu_2, \dots, \mu_n) = \frac{1}{m} \left( \sum_{i=1}^m x_1^i, \sum_{i=1}^m x_2^i, \dots, \sum_{i=1}^m x_n^i \right)$$

and a covariance matrix

$$\mathbf{C} = \begin{pmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,n} \\ c_{2,1} & c_{2,2} & \dots & c_{2,n} \\ \dots & \dots & \dots & \dots \\ c_{n,1} & c_{n,2} & \dots & c_{n,n} \end{pmatrix}$$

where

$$c_{j,j} = \frac{1}{m} \sum_{i=1}^m ((x_j^i - \mu_j) (x_j^i - \mu_j))$$

and

$$c_{j,k} = \frac{1}{m} \sum_{i=1}^m ((x_j^i - \mu_j) (x_k^i - \mu_k)).$$

Due to the commutative property of the product of numbers, it follows that  $\forall j, k : c_{j,k} = c_{k,j}$ , i.e. the covariance matrix is symmetric.

The vector  $\mu$  represents the barycentre of the distribution. The covariance matrix  $\mathbf{C}$  describes the geometry of the distribution with respect to the reference system. More specifically, the diagonal elements represent the deviation of the design variable from the barycentre whilst the extradiagonal elements represent a measurement of how two design variables vary together.

Let us consider an optimisation problem and let us imagine to sample  $m$  points, with  $m$  arbitrarily large, within the decision space  $D$ . With reference to Fig. 1a, let us imagine to crowd the elliptic inner contour with points and calculate the mean vector and covariance matrix. It can be verified that  $\mu$  would be the optimum and the covariance matrix  $\mathbf{C}$  would be diagonal. However, if we crowded with points the elliptic inner contour in Fig. 1b we would observe that the corresponding covariance matrix  $\mathbf{C}$  would be full.

This observation could be extended to the general case: a basin of attraction that has the highest gradient along one of the axes is characterised by a diagonal covariance matrix of the points crowding it. Hence, we propose to use the reference system corresponding to a diagonal covariance matrix as move directions for PS in order to find the highest gradient direction. That is, we propose to run PS along the directions identified by the eigenvectors of  $\mathbf{C}$ , see [20]. These directions are the columns of a non-singular matrix  $\mathbf{P}$  such that

$$\mathbf{D} = \mathbf{P}^{-1}\mathbf{C}\mathbf{P}$$

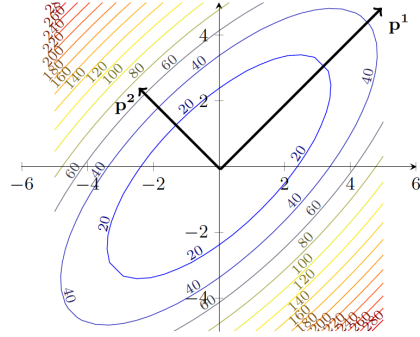
where  $\mathbf{D}$  is a diagonal matrix.

**Observation 1** *Since the covariance matrix  $\mathbf{C}$  is symmetric, it follows that*

1.  $\mathbf{C}$  is always diagonalisable and hence there always exists a non-singular transformation matrix  $\mathbf{P}$  that diagonalises  $\mathbf{C}$ .
2. each pair of the eigenvectors of  $\mathbf{C}$  (column of matrix  $\mathbf{P}$ ) is orthogonal, the proposed search directions compose an orthogonal system of coordinates.

The matrix  $\mathbf{P}$  is the linear transformation that changes the variables of the problem. For example, the directions of the eigenvectors associated with the covariance matrix or the rotated ellipsoid in Fig. 1b are shown in Fig. 2.

Equivalently to what is stated above, the proposed algorithm is based on the consideration that the nonseparability is not just a feature of the function to optimise. Nonseparability is a feature of the function within its reference system. Hence, a (local) change in the reference system can (locally) lead to a much easier optimisation problem.



**Fig. 2.** Ellipsoid in two dimensions rotated by  $45^\circ$ : PS moving along the thick directions would be efficient.

### 3.2 Algorithmic outline of the proposed method

Let us assume that we have a reliable matrix  $\mathbf{C}$  available of the basin of attraction under investigation. The matrix  $\mathbf{C}$  would be diagonalised and the eigenvectors would be the columns of a transformation matrix  $\mathbf{P}$ :

$$\mathbf{P} = (\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^n).$$

In this article the diagonalisation is performed by the numerical method described and implemented in [20].

In order to better illustrate the relation between the change of coordinates and the diagonalisation of the covariance matrix, let us consider the ellipsoid function

$$f(x, y) = a(\cos(\alpha)x + \sin(\alpha)y)^2 + b(\sin(\alpha)x - \cos(\alpha)y)^2.$$

Figure 3 illustrates this function (with  $a = 1$  and  $b = 6$ ), for  $\alpha = 0^\circ$  and  $\alpha = 30^\circ$  respectively, a sample of points, the eigenvectors, and the directions identified by them. It can be observed that for  $\alpha = 0^\circ$  the directions of the eigenvectors coincide with those of the reference axes whilst for  $\alpha = 30^\circ$  the axes appear rotated.

The new version of PS operates on a starting point  $\mathbf{x}$  and generates trial solutions for the  $i^{th}$  design variable by computing

$$\mathbf{x}^t = \mathbf{x} - \rho \cdot \mathbf{P}\mathbf{e}^i = \mathbf{x} - \rho \cdot \mathbf{p}^i$$

and

$$\mathbf{x}^t = \mathbf{x} + \frac{\rho}{2} \cdot \mathbf{P}\mathbf{e}^i = \mathbf{x} + \frac{\rho}{2} \cdot \mathbf{p}^i$$

where the product of a matrix by a vector  $\mathbf{P}\mathbf{e}^i$  is equal to  $\mathbf{p}^i$ .

The pseudocode of the proposed Covariance Pattern Search (CPS) Algorithm is shown in Algorithm 2

**Algorithm 2** The proposed Covariance Pattern Search

---

```

INPUT  $\mathbf{x}$ 
INPUT the covariance matrix  $\mathbf{C}$ 
Process the covariance matrix  $\mathbf{C}$  and calculate the transformation matrix  $\mathbf{P} = (\mathbf{p}^1, \mathbf{p}^2, \dots, \mathbf{p}^n)$ 
whose columns are the eigenvectors of  $\mathbf{C}$ 
while local budget condition do
   $\mathbf{x}^t = \mathbf{x}$ 
  for  $i = 1 : n$  do
     $\mathbf{x}^t = \mathbf{x} - \rho \cdot \mathbf{p}^i$ 
    if  $f(\mathbf{x}^t) \leq f(\mathbf{x})$  then
       $\mathbf{x} = \mathbf{x}^t$ 
    else
       $\mathbf{x}^t = \mathbf{x} + \frac{\rho}{2} \cdot \mathbf{p}^i$ 
      if  $f(\mathbf{x}^t) \leq f(\mathbf{x})$  then
         $\mathbf{x} = \mathbf{x}^t$ 
      end if
    end if
  end for
  if  $\mathbf{x}$  has not been updated then
     $\rho = \frac{\rho}{2}$ 
  end if
end while
RETURN  $\mathbf{x}$ 

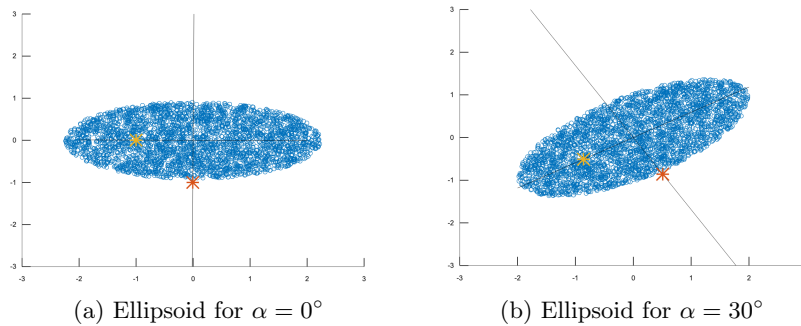
```

---

**3.3 Working example**

In order to highlight the difference in functioning between Algorithm 1 and Algorithm 2, and the benefits of the proposed CPS with respect to its standard counterpart, we have run both of the algorithms on the ellipsoid function above in this case with the arbitrary values  $a = 1$  and  $b = 76$ . We arbitrarily chose an angle  $\alpha = \frac{\pi}{3.5} \approx 51.43^\circ$  and a starting point  $(71.4, -49.1)$ . The search directions of CPS are the columns of the following matrix  $\mathbf{P}$ :

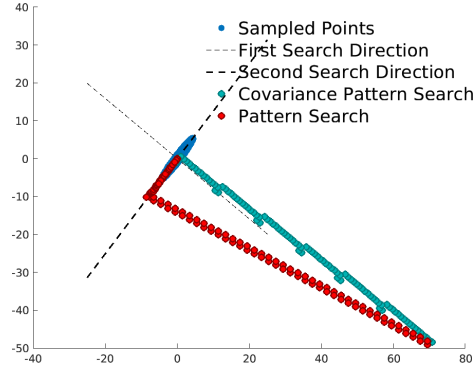
$$\mathbf{P} = \begin{pmatrix} -0.7821 & 0.6231 \\ 0.6231 & 0.7821 \end{pmatrix}.$$



**Fig. 3.** Sampling of points (blue circles) within the basin of attraction of the ellipsoid, resulting eigenvectors (asterisks), and search directions of the proposed method (lines)



We let both the algorithms run until  $\rho < 10^{-75}$ , which was considered to be when the problem to be solved. Whilst PS requires the calculation of 16169 objective function evaluations, CPS solves the problem after only 2040 evaluations. Fig. 4 comparatively illustrates the functioning of the two algorithms. The proposed CPS moves along more convenient directions than the variable directions of PS, thus quickly reaching the optimum.



**Fig. 4.** Illustration of the functioning of the proposed Covariance Pattern Search and advantages with respect to the standard Pattern Search (the proposed algorithm solves the problem in 2040 whilst its standard counterpart requires 16169 steps)

### 3.4 Limitations of the proposed Covariance Pattern Search

The most evident limitation of the proposed CPS is that a reliable matrix  $\mathbf{C}$  of points of the basin of attraction must be estimated. In the present paper, when a basin of attraction is identified, a set of points populating this basin is found by treating the optimisation problem as a level set problem: random points are generated and those whose objective function values fall below a threshold are saved in a data structure, see Figures 3. These points are then used to calculate the covariance matrix  $\mathbf{C}$ .

The major issue associated with this approach is that a suitable threshold is problem dependent and empirically determined in each case. Besides being inelegant, this procedure can also be computationally expensive in the high dimensional domain. Although, the initial computational effort may be paid off by a much faster solution to the optimisation problem, see Fig. 4.

The second limitation is that for multimodal optimisation problems promising basins of attraction must be identified before one basin is selected and CPS is executed. This limitation would make CPS unsuitable as a stand-alone algorithm and/or would require a preprocessing algorithm to estimate the locations of potential basins of attraction. However, CPS just like other local search algorithms can be embedded in a metaheuristic framework and can exploit the

function calls performed by a global optimiser (or other local search algorithms) as part of the preprocessing, see [19].

Alternatively, CPS can be applied to a multimodal problem at the end of preprocessing for multimodal optimisation, see e.g. the method in [26] or the intelligent sampling proposed in [23]. In the latter, an initial population of  $m$  candidate solutions ( $n$ -dimensional vectors) is sampled within the decision space  $D$ . Each candidate solution undergoes the application, with a limited budget, of the two local searchers. The solutions returned by the two local search algorithms are then processed by the K-means clustering algorithm enhanced by a control on the Silhouette to decide the correct number of clusters, as explained in [23]. Each cluster represents a basin of attraction and its candidate solutions are the sampled points on which the covariance matrix  $\mathbf{C}$  is computed.

## 4 Numerical Results

In order to experimentally demonstrate the effectiveness of CPS, and the idea inspiring it, we have tested CPS against PS in multiple scenarios. To simulate the local search conditions we have considered and adapted a sample of the functions from the CEC 2013 benchmark (focussing on unimodal problems), see [16]. We have used two versions of the ellipsoid since the use of two versions was relevant to demonstrate the performance of the local search. Each problem has been scaled to 10, 30, and 50 dimensions and has been studied in  $[-100, 100]^n$ . Table 1 displays the functions and depicts the shape of the corresponding basins of attraction. For each problem, a shift and a rotation has been applied: with reference to Table 1 the variable

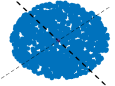
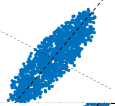
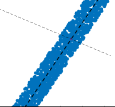
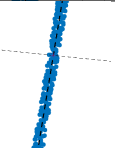
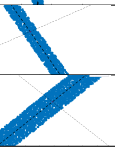
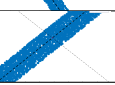
$$\mathbf{z} = \mathbf{Q}_k (\mathbf{x} - \mathbf{o})$$

where  $\mathbf{o}$  is the shifting vector (the same used in [16]) and  $\mathbf{Q}_k$  is a rotation matrix (a randomly generated orthogonal matrix) set for the  $k^{th}$  problem, see [4].

The PS in Algorithm 1 has been executed with a budget of  $10000 \times n$  function calls where  $n$  is the problem dimensionality. In order to guarantee a fair comparison, the budget of the proposed CPS in Algorithm 2 has been split into two parts:  $5000 \times n$  function calls have been used to build the covariance matrix  $\mathbf{C}$  whilst  $5000 \times n$  function calls have been spent to execute the algorithm. Due to the nature of PS and CPS, i.e. deterministic local search, the bound handling has been performed by saturating the design variable to the bound. We preferred the saturation to the bound over the toroidal insertion or reflection [10] since the latter two mechanisms would be equivalent to the sampling of a point. This sampling would disrupt the gradient estimation logic of Pattern Search.

Although PS and CPS are deterministic algorithms, their performance can depend on the initial point and, for CPS, on the sampled points used to estimate the covariance matrix. Thus, for each scenario, we sampled 51 initial points within the entire domain and used them to execute PS and CPS. The average objective function values and standard deviations over the 51 runs have been calculated. The statistical significance of the results has been enhanced by the

**Table 1.** Basins of attraction functions

function name	function formula	basin of attraction in 2D
sphere	$f_1(\mathbf{x}) = \sum_{i=1}^n z_i^2$	
ellipsoid	$f_2(\mathbf{x}) = \sum_{i=1}^n 50 (i^2 z_i)^2$	
ill-conditioned ellipsoid	$f_3(\mathbf{x}) = \sum_{i=1}^n (10^6)^{\frac{i-1}{n-1}} z_i^2$	
bent cigar	$f_4(\mathbf{x}) = z_1^2 + 10^6 \sum_{i=2}^n z_i^2$	
discus	$f_5(\mathbf{x}) = 10^6 z_1^2 + \sum_{i=2}^n z_i^2$	
sum of powers	$f_6(\mathbf{x}) = \sqrt{\sum_{i=1}^n  z_i ^{(2+4\frac{i-1}{n-1})}}$	

application of the Wilcoxon rank sum test, see [32, 12]. In the Tables in this section, a “+” indicates that CPS significantly outperforms PS, a “-” indicates that PS significantly outperforms CPS, and a “=” indicates that there is no significant difference in performance. Regarding CPS, the threshold values used in this study are reported in Table 2.

**Table 2.** Thresholds  $thr$  in 10, 30, and 50 dimensions

$n$	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$
10	$10^4$	$10^9$	$5 \times 10^8$	$10^9$	$10^9$	$10^4$
30	$5 \times 10^4$	$5 \times 10^{11}$	$2 \times 10^9$	$2 \times 10^9$	$10^8$	$10^5$
50	$10^5$	$5 \times 10^{13}$	$5 \times 10^9$	$2 \times 10^9$	$5 \times 10^7$	$3 \times 10^5$

Numerical results show that, besides  $f_1$ , the proposed CPS consistently outperforms the standard PS across the three dimensions under consideration. The problem  $f_1$  is characterised by a central symmetry. This means that the rotation is ineffective and any reference system would broadly perform in the same way. The use of the eigenvectors as search directions not only systematically improves upon PS but appears to solve some problems such as  $f_5$  (discus).

In order to ensure that the results of proposed CPS are not biased by specific rotation matrices, the experiments have been repeated on the problems in Table

**Table 3.** Average error  $\text{avg} \pm$  standard deviation  $\sigma$  over 51 runs for the problems listed in Table 1

	Pattern Search		Covariance Pattern Search		
	avg	$\sigma$	avg	$\sigma$	W
10 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	2.7768e-29	1.4329e-29	-
$f_2$	1.8198e+03	1.9132e+03	<b>1.1062e-03</b>	3.4915e-03	+
$f_3$	7.0241e+04	9.5716e+04	<b>4.1064e+03</b>	6.6534e+03	+
$f_4$	5.3135e+03	3.7177e+03	<b>3.3017e-06</b>	9.9933e-06	+
$f_5$	9.3944e+03	1.2996e+04	<b>3.0984e-25</b>	5.3080e-25	+
$f_6$	5.5475e+01	9.4900e+01	<b>2.6276e-05</b>	1.3087e-05	+
30 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	1.3562e-28	5.1167e-29	-
$f_2$	1.1636e+08	2.3036e+08	<b>6.9435e+05</b>	6.4717e+05	+
$f_3$	2.9155e+05	2.4771e+05	<b>1.8116e+04</b>	1.1143e+04	+
$f_4$	5.2064e+03	5.3776e+03	<b>4.6182e-13</b>	3.2833e-12	+
$f_5$	8.4596e+03	3.4177e+04	<b>5.3340e-28</b>	3.2325e-27	+
$f_6$	1.2427e+02	1.7049e+02	<b>6.4462e-05</b>	1.7549e-05	+
50 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	4.1998e-28	9.9033e-29	-
$f_2$	4.3878e+08	8.6339e+08	<b>1.6151e+07</b>	1.3937e+07	+
$f_3$	3.7522e+05	2.7408e+05	<b>5.1237e+04</b>	3.4165e+04	+
$f_4$	5.6398e+03	7.4905e+03	<b>5.5627e-22</b>	1.4649e-21	+
$f_5$	2.8261e+01	1.3792e+02	<b>7.8822e-27</b>	1.2792e-26	+
$f_6$	1.8153e+02	2.0042e+02	<b>1.0584e-04</b>	2.3244e-05	+

1 with a modified experimental condition. For each run a new rotation matrix has been generated and both PS and CPS have been run on the rotated problem. An additional 51 independent runs have been executed under this new condition. Table 4 displays the results for this set of experiments with random matrices.

Numerical results in Table 4 show that CPS maintains the same performance irrespective of the rotation matrix. These results allow us to conjecture that the proposed mechanism of optimising along the direction of the eigenvectors is effective and the generation of the covariance matrix is robust.

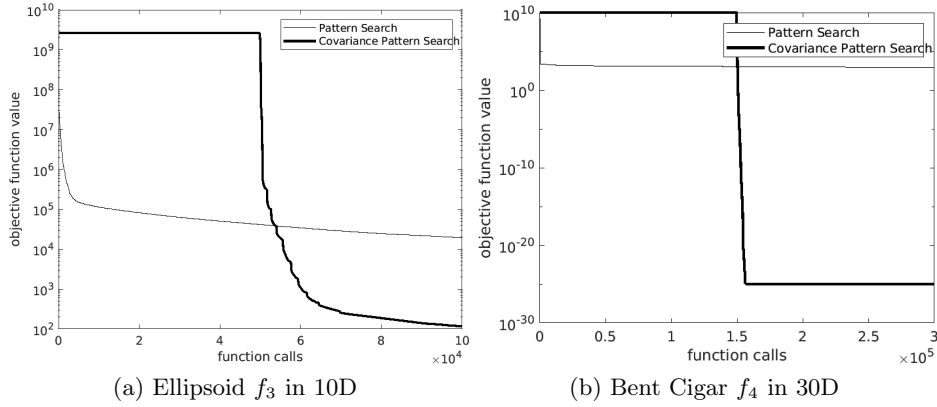
To further illustrate the functioning of CPS with respect to PS, Figures 5 show the performance trend of two specific runs.

The results in Figures 5 show that CPS achieves results orders of magnitude better than PS. It must be noted that we represented the performance of CPS as a constant for the first half of the trend. This is due to the objective function evaluation budget used to calculate the covariance matrix and the corresponding eigenvectors, with the optimisation only occurring in the second part of the budget. At half of the budget CPS starts the optimisation by exploiting the directions suggested by the preliminary phase. A dramatic decrease in the fitness value is evident in the illustration. When the proposed logic is considered within a local search/optimisation algorithm, this logic can be efficiently implemented by using the samples produced by a global searcher, whilst the local search can run with a short budget, just to exploit the abrupt improvement.

Finally, in order to highlight the potential and limitations of CPS, we have compared it against the Covariance Matrix Adaptation Evolution Strategy (CMAES), according to the implementation in [13] and the default parameters therein (ini-

**Table 4.** Average error  $\text{avg} \pm$  standard deviation  $\sigma$  over 51 runs for the problems listed in Table 1 subject to random rotation at each run

	Pattern Search		Covariance Pattern Search		W
	avg	$\sigma$	avg	$\sigma$	
10 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	3.6603e-29	2.0133e-29	-
$f_2$	1.2340e+03	1.2462e+03	<b>3.7683e-04</b>	1.1634e-03	+
$f_3$	1.3739e+05	2.3991e+05	<b>2.9764e+03</b>	2.7313e+03	+
$f_4$	3.5615e+03	6.8974e+03	<b>3.8205e+01</b>	1.0569e+02	+
$f_5$	7.8684e+03	1.1461e+04	<b>1.0087e-23</b>	3.1002e-23	+
$f_6$	1.3676e+02	4.3249e+02	<b>2.6698e-05</b>	1.0627e-05	+
30 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	1.3810e-28	5.9062e-29	-
$f_2$	9.8109e+07	2.6281e+08	<b>7.2563e+05</b>	9.0521e+05	+
$f_3$	1.7506e+05	1.8632e+05	<b>1.7248e+04</b>	1.0875e+04	+
$f_4$	5.8954e+03	1.0909e+04	<b>2.2632e-01</b>	1.6163e+00	+
$f_5$	1.1985e+03	5.2373e+03	<b>7.9322e-26</b>	4.2633e-25	+
$f_6$	1.3485e+02	1.8275e+02	<b>6.4239e-05</b>	1.8127e-05	+
50 dimensions					
$f_1$	<b>0.0000e+00</b>	0.0000e+00	4.2394e-28	9.7414e-29	-
$f_2$	8.2205e+08	1.7628e+09	<b>1.5302e+07</b>	1.2051e+07	+
$f_3$	3.1913e+05	2.8232e+05	<b>5.8034e+04</b>	3.0679e+04	+
$f_4$	9.3975e+03	1.0959e+04	<b>3.8535e-21</b>	1.4037e-20	+
$f_5$	1.2639e+02	4.5309e+02	<b>2.8811e-26</b>	1.2290e-25	+
$f_6$	1.8761e+02	1.8060e+02	<b>9.1000e-05</b>	1.8870e-05	+

**Fig. 5.** Performance trend (logarithmic scale) of Pattern Search vs Covariance Pattern Search for two runs: for half of the budget CPS analyses the problem to generate the covariance matrix and then optimises the problem.

tial  $\sigma$  set to one third of the domain). Table 5 displays the results of this comparison for the same problems in Table 3.

**Table 5.** Average error  $\text{avg} \pm$  standard deviation  $\sigma$  over 51 runs for the problems listed in Table 1

	CMAES		Covariance Pattern Search		
	avg	$\sigma$	avg	$\sigma$	W
10 dimensions					
$f_1$	2.1589e-15	2.9083e-15	<b>2.7768e-29</b>	1.4329e-29	+
$f_2$	<b>1.4746e-15</b>	1.1443e-15	1.1062e-03	3.4915e-03	-
$f_3$	<b>1.0489e-15</b>	8.2847e-16	4.1064e+03	6.6534e+03	-
$f_4$	<b>1.8441e-14</b>	8.2847e-16	3.3017e-06	9.9933e-06	=
$f_5$	1.5539e-14	2.8869e-14	<b>3.0984e-25</b>	5.3080e-25	+
$f_6$	<b>9.8517e-13</b>	9.4715e-13	2.6276e-05	1.3087e-05	-
30 dimensions					
$f_1$	1.2862e-15	2.3923e-16	<b>1.3562e-28</b>	5.1167e-29	+
$f_2$	<b>1.2574e-15</b>	3.6076e-16	6.9435e+05	6.4717e+05	-
$f_3$	<b>1.1737e-15</b>	2.4995e-16	1.8116e+04	1.1143e+04	-
$f_4$	<b>1.2556e-14</b>	2.0370e-14	4.6182e-13	3.2833e-12	=
$f_5$	8.7935e-15	1.1728e-14	<b>5.3340e-28</b>	3.2325e-27	+
$f_6$	<b>1.1134e-11</b>	9.1361e-12	6.4462e-05	1.7549e-05	-
50 dimensions					
$f_1$	1.1017e-15	4.7700e-16	<b>4.1998e-28</b>	9.9033e-29	+
$f_2$	<b>9.1890e-15</b>	2.0913e-15	1.6151e+07	1.3937e+07	-
$f_3$	<b>1.2302e-15</b>	7.1779e-16	5.1237e+04	3.4165e+04	-
$f_4$	1.2606e+04	3.3045e+04	<b>5.5627e-22</b>	1.4649e-21	+
$f_5$	9.0248e-03	4.4211e-02	<b>7.8822e-27</b>	1.2792e-26	+
$f_6$	<b>4.8125e-11</b>	2.8066e-11	1.0584e-04	2.3244e-05	-

Table 5 shows that for about half of the problems CPS is competitive ( $f_1$  and  $f_4$ ) or outperforms CMAES ( $f_5$ ), whilst for some other problems CMAES achieves results that are orders of magnitudes better in performance than CPS. This fact happens especially in cases of steep gradients, such as the ellipsoid functions  $f_2$  and  $f_3$ . However, this paper proposes a principle about search directions rather than a full algorithm. For example, unlike CMAES, CPS employs a naive search operation based on the simple halving of a radius. Nonetheless, the proposed logic can be exported to other schemes and more sophisticated operators, e.g. an adaptive search radius, can enhance upon the current performance.

## 5 Conclusion

This article provides a proof of principle of a conjecture: the search directions identified by the eigenvectors of the covariance matrix of a population of points filling a basin of attraction are efficient to search for the local optimum. This idea has been tested on a naive implementation of pattern search, which has displayed a major systematic improvement over the standard version of pattern search for the problems considered in this study.

The main limitation of the proposed approach is that in order to build the covariance matrix a threshold value has to be set. This value is currently set

manually. Our future work will include a protocol for setting this threshold to ensure that a sample of points representing the basin of attraction is detected.

Other future developments will include the extension of the proposed logic to more complex algorithms such as Hooke-Jeeves Pattern search and population based algorithms. Furthermore the proposed algorithm will also be tested within frameworks composed of multiple algorithms, such as Memetic Algorithms and Hyperheuristics.

## References

1. Al-Dabbagh, R.D., Neri, F., Idris, N., Baba, M.S.: Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy. *Swarm and Evolutionary Computation* **43**, 284 – 311 (2018)
2. Burke, E.K., Kendall, G., Soubeiga, E.: A Tabu Search hyperheuristic for Timetabling and Rostering. *Journal of Heuristics* **9**(6), 451–470 (2003)
3. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for on-line and off-line control design of PMSM drives. *IEEE Transactions on System Man and Cybernetics-part B* **37**(1), 28–41 (2007)
4. Caraffini, F., Neri, F.: A study on rotation invariance in differential evolution. *Swarm and Evolutionary Computation* **50**, 100436 (2019)
5. Caraffini, F., Neri, F., Iacca, G., Mol, A.: Parallel memetic structures. *Information Sciences* **227**(0), 60 – 82 (2013)
6. Caraffini, F., Neri, F., Passow, B.N., Iacca, G.: Re-sampled inheritance search: high performance despite the simplicity. *Soft Computing* **17**(12), 2235–2256 (Dec 2013)
7. Caraffini, F., Neri, F., Picinali, L.: An analysis on separability for memetic computing automatic design. *Information Sciences* **265**, 1–22 (2014)
8. Davidon, W.C.: Variable metric method for minimization. *SIAM Journal on Optimization* **1**(1), 1–17 (1991)
9. Dumitrescu, I., Stützle, T.: Combinations of local search and exact algorithms. In: *EvoWorkshops 2003: Applications of Evolutionary Computing, Lecture Notes in Computer Science* 2611, pp. 211–223 (2003)
10. Eiben, A.E., Smith, J.E.: *Introduction to Evolutionary Computation*. Springer, Berlin (2015), second edition
11. Esparcia-Alcázar, A.I., Almenar, F., Vos, T.E.J., Rueda, U.: Using genetic programming to evolve action selection rules in traversal-based automated software testing: results obtained with the testar tool. *Memetic Computing* **10**(3), 257–265 (Sep 2018)
12. Garcia, S., Fernandez, A., Luengo, J., Herrera, F.: A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Computing* **13**(10), 959–977 (2008)
13. Hansen, N.: The CMA Evolution Strategy (2012), <http://www.lri.fr/hansen/cmaesintro.html>
14. Hooke, R., Jeeves, T.A.: Direct search solution of numerical and statistical problems. *Journal of the ACM* **8**, 212–229 (Mar 1961)
15. Krasnogor, N.: Toward Robust Memetic Algorithms. In: Hart, W.E., Krasnogor, N., Smith, J.E. (eds.) *Recent Advances in Memetic Algorithms*, pp. 185–207. *Studies in Fuzziness and Soft Computing*, Springer, Berlin, Germany (2004)

16. Liang, J.J., Qu, B.Y., Suganthan, P.N., Hernandez-Daz, A.G.: Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization. Tech. Rep. 201212, Zhengzhou University and Nanyang Technological University, Zhengzhou China and Singapore (2013)
17. Luersen, M.A., Riche, R.L.: Globalized neldermead method for engineering optimization. *Computers & Structures* **82**(23), 2251 – 2260 (2004)
18. Nelder, A., Mead, R.: A simplex method for function optimization. *Computation Journal* **Vol 7**, 308–313 (1965)
19. Neri, F., Cotta, C.: Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation* **2**, 1–14 (2012)
20. Neri, F.: *Linear Algebra for Computational Sciences and Engineering*. Springer, second edn. (2019)
21. Nocedal, J., Wright, S.: *Numerical Optimization*. Springer (2006), second edition
22. Peng, F., Tang, K., Chen, G., Yao, X.: Population-Based Algorithm Portfolios for Numerical Optimization. *IEEE Tr. Evol. Comp.* **14**(5), 782–800 (2010)
23. Poikolainen, I., Neri, F., Caraffini, F.: Cluster-based population initialization for differential evolution frameworks. *Information Sciences* **297**, 216 – 235 (2015)
24. Powers, S.T., Ekárt, A., Lewis, P.R.: Modelling enduring institutions: The complementarity of evolutionary and agent-based approaches. *Cognitive Systems Research* **52**, 67 – 81 (2018)
25. Puchinger, J., Raidl, G.R.: Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In: *International Work-Conference on the Interplay Between Natural and Artificial Computation (IWINAC 2005)*, Lecture Notes in Computer Science, vol. 3562, pp. 41–53 (2005)
26. Qu, B.Y., Suganthan, P.N., Liang, J.J.: Differential evolution with neighborhood mutation for multimodal optimization. *IEEE Transactions on Evolutionary Computation* **16**(5), 601–614 (Oct 2012)
27. Rosenbrock, H.H.: An automatic Method for finding the greatest or least Value of a Function. *The Computer Journal* **3**(3), 175–184 (1960)
28. Rostami, S., Neri, F.: A fast hypervolume driven selection mechanism for many-objective optimisation problems. *Swarm and Evolutionary Computation* **34**, 50 – 67 (2017)
29. Torczon, V.: On the convergence of pattern search algorithms. *SIAM Journal on Optimization* **7**(1), 1–25 (1997)
30. Tseng, L.Y., Chen, C.: Multiple trajectory search for Large Scale Global Optimization. In: *Proceedings of the IEEE Congress on Evolutionary Computation*. pp. 3052–3059 (2008)
31. Wang, Y., Li, H.X., Huang, T., Li, L.: Differential evolution based on covariance matrix learning and bimodal distribution parameter setting. *Applied Soft Computing* **18**, 232 – 247 (2014)
32. Wilcoxon, F.: Individual comparisons by ranking methods. *Biometrics Bulletin* **1**(6), 80–83 (1945)
33. Xu, L., Hutter, F., Hoos, H., Leyton-Brown, K.: SATzilla: Portfolio-based algorithm selection for SAT. *Jour. of Artificial Intelligence Research* **32**, 565–606 (2008)
34. Yu., W.C.: Positive basis and a class of direct search techniques. *Scientia Sinica (in Chinese)* **9**(S1), 53–67 (1979)