Applied Soft Computing 45 (2016) 71–85

Contents lists available at ScienceDirect

Applied Soft Computing

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

Biogeography-based optimization with covariance matrix based migration

Xu Chen [a,*], Huaglory Tianfield [b], Wenli Du [c], Guohai Liu [a]

[a] School of Electrical and Information Engineering, Jiangsu University, Zhenjiang 212013, China
[b] School of Engineering and Built Environment, Glasgow Caledonian University, Glasgow G4 0BA, UK
[c] Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China

ARTICLE INFO

Article history:
Received 12 June 2015
Received in revised form 28 January 2016
Accepted 16 April 2016
Available online 26 April 2016

Keyword:
Biogeography-based optimization
Covariance matrix based migration
Global numeric optimization
Evolutionary algorithm
Rotational invariance

ABSTRACT

Biogeography-based optimization (BBO) is a new evolutionary algorithm. The major problem of basic BBO is that its migration operator is rotationally variant, which leaves BBO performing poorly in non-separable problems. To overcome this drawback of BBO, in this paper, we propose the covariance matrix based migration (CMM) to relieve BBO's dependence upon the coordinate system so that BBO's rotational invariance is enhanced. By embedding the CMM into BBO, we put forward a new BBO approach, namely biogeography-based optimization with covariance matrix based migration, called CMM-BBO. Specifically, CMM-BBO algorithms are developed by the CMM operator being randomly combined with the original migration in various existing BBO variants. Numeric simulations on 37 benchmark functions show that our CMM-BBO approach effectively improves the performance of the existing BBO algorithms.

1. Introduction

Inspired from the nature, a variety of evolutionary algorithms (EAs) has been developed to effectively tackle global optimization problems, for example, genetic algorithms (GA) [1], evolution strategies (ES) [2], differential evolution (DE) [3–5], particle swarm optimization (PSO) [6,7] and so on. EAs have the advantages such as robustness, reliability, global search capability and little or no prior knowledge required.

Biogeography-based optimization (BBO), proposed by Simon [8], is a new EA based on biogeographic evolution. BBO has proven itself a competitive heuristic to other EAs on a wide range of problems [8–12].

To improve the performance of basic BBO, a number of BBO variants have been proposed, which generally fall into three categories, i.e., (i) BBO with new migration or mutation operators, (ii) BBO hybrid with other EAs, and (iii) BBO with multiple populations or local topologies.

BBO with new migration or mutation operators: Gong et al. [13] proposed a real-coded BBO (called rcBBO) with three kinds of mutation operators, namely Gaussian mutation, Cauchy mutation, and Lévy mutation. Li and Yin [14] proposed a multi-operator BBO (called moBBO) with generalized migration operator using multi-parent migration model. Xiong et al. [15] proposed a BBO with polyphyletic migration operator and orthogonal learning strategy, called polBBO. Li et al. [16] proposed a perturbation optimization based BBO (called pBBO) with perturbation migration operator using sinusoidal migration model. Ma and Simon [17] proposed a blended BBO, for constrained optimization, with blended migration operator by analogue to the blended crossover operator in GA. Simon et al. [18] proposed a BBO with linearized migration that makes the migration more rotationally invariant.

BBO hybrid with other EAs: Du et al. [19] incorporated the elitism mechanism of evolutionary strategy and a new immigration refusal scheme into BBO and proposed a BBO/ES/RE algorithm. Gong et al. [20] incorporated DE's mutation operator with BBO's migration operator and proposed a DE/BBO algorithm, taking advantage of BBO's exploitation ability and DE's exploration ability. Boussaid et al. [21] incorporated DE with BBO through a two-stage updating mechanism and proposed

* Corresponding author.
E-mail addresses: xuchen@ujs.edu.cn (X. Chen), h.tianfield@gcu.ac.uk (H. Tianfield), wldu@ecust.edu.cn (W. Du), ghliu@ujs.edu.cn (G. Liu).

http://dx.doi.org/10.1016/j.asoc.2016.04.022
1568-4946/

a DE hybrid BBO algorithm. Kundra and Sood [22] combined PSO with BBO to optimize shortest path problems. Savsani et al. [23] incoprporated artificial immune algorithm and ant colony optimization with BBO and proposed four hybrid BBO variants.

BBO with multiple populations or local topologies: Zheng et al. [24] integrated three different local topologies (i.e., ring, square, and random) in BBO to enhance BBO's exploration ability, and proposed a localized BBO. Zheng et al. [25] divided the whole population into multiple sub-populations with each sub-population being evolved through a separate BBO, and proposed a cooperative coevolutionary biogeography-based optimizer (called cBBO). Ma et al. [26] proposed a BBO with an ensemble of migration models using three parallel populations, each implementing a different migration model.

In addition to the three categories of BBO variants above, Ergeze et al. [27] proposed an oppositional BBO using opposition-based learning. Saremi et al. [28] proposed a chaotic BBO using ten chaotic maps to define selection, emigration, and mutation probabilities.

In BBO algorithms as mentioned above, either basic BBO or variants, the migration operator is crucial. In fact, it is through the migration operator that multiple parents contribute towards generating an offspring. However, the migration operators in the existing BBO algorithms are heavily dependent upon the coordinate systems, which leaves poor performance in dealing with non-separable problems [18]. A non-separable problem is one the fitness of which depends upon the variables combinatorially rather than individually. In other words, variables in a non-separable problem are tightly intermeshed with one another.

Simon et al. pointed out [18] that a major drawback of BBO is that it treats each solution feature independently, which leaves BBO rotationally variant. Rotational variance means that BBO generally performs poorly when applied to non-separable problems. However, most real-world problems are non-separable. Thus, rotational variance restricts BBO's applicability to wider problems.

To address this drawback of BBO, the key question is: how to relieve BBO's dependence upon the coordinate system and enhance BBO's rotational invariance?

Covariance matrix learning (CML) was first adopted in covariance matrix adaptation evolution strategy (CMAES) [2]. CML effectively adapts the search according to the landscape of the optimization function. Basically, CML rotates the coordinate system to make the problem pseudo-separable. CML employed in DE makes the crossover rotationally invariant [29,30], which significantly improves the performance of DE.

In this paper we will propose the covariance matrix based migration (CMM) to relieve BBO's dependence upon the coordinate system so that BBO's rotational invariance is enhanced. By use of our proposed CMM operator, the original coordinate system is rotated into an eigenvectorbased one, in which solutions can share their information more efficiently.

By embedding the CMM into BBO, we put forward a new BBO approach, namely biogeography-based optimization with covariance matrix based migration, called CMM-BBO. Specifically, CMM-BBO algorithms are developed by the CMM operator being randomly combined with the original migration in various exsiting BBO algorithms.

The remainder of the paper is arranged as follows. Section 2 proposes the covariance matrix based migration and puts forward the CMM-BBO approach. Section 3 conducts thorough performance evaluations of four CMM-BBO algorithms through numeric simulations on 37 benchmark functions and comparisons with other EAs. Lastly, Section 4 draws the conclusions.

2. BBO with covariance matrix based migration

2.1. Preliminary: basic BBO

BBO [8] is a new population-based, biogeographically inspired global optimization algorithm. In BBO, each individual is regarded as a "habitat" or "island" with a Habitat Suitability Index (HSI), which is similar to the fitness in EAs. A good solution means a habitat with a high HSI, while a poor solution indicates a habitat with a low HSI.

A solution can be represented by a set of Suitability Index Variables (SIV). In BBO's migration process, high HSI solutions should share their features with low HSI ones; while low HIS solutions take in new features from high HIS ones. In BBO, each individual has its own immigration rate λ and emigration rate μ, which can be calculated based on HSI. A high HSI habitat has a high species emigration rate μ while a low HSI habitat has a high species immigration rate λ. For example, in a linear model of species richness, a habitat H_i's immigration rate λ_i and emigration rate μ_i can be calculated as follows.

$$\lambda_i = I\left(1-\frac{i}{n}\right) \tag{1}$$

$$\mu_i = E\left(\frac{i}{n}\right) \tag{2}$$

where I is the maximum immigration rate, E the maximum emigration rate, n the population size, i the index of the individual in order, where i = 1 denoting the worst individual while i = n denoting the best. Eqs. (1) and (2) are called linear migration model of the migration rates.

Migration modifies habitats by mixing the features within a population. BBO also uses a mutation operator to change the SIV of a habitat itself, and thus increases the diversity of a population. For each habitat H_i, species count probability P_i, computed from λ_i and μ_i, measures the *a priori* likelihood that the habitat is expected to become a solution to the problem. In reality, either a very high HSI habitat or a very low HSI habitat is rarely probable, but most probable is a medium HSI habitat. A habitat's mutation rate π_i is inversely proportional to its probability, i.e.,

$$\pi_i = \pi_{\max}\left(1-\frac{P_i}{P_{\max}}\right) \tag{3}$$

where $\pi_{\max}$ is a control parameter and $P_{\max}$ the maximum habitat probability in a population.

Basic BBO can be formulated as in Algorithm 1, where D is the dimension of the optimization problem, l_d and u_d the lower and upper bounds of the d-th dimension, respectively, and rand a random number function uniformly distributed in [0,1].

Algorithm 1. Basic BBO

Randomly Initialize a population of n habitats $H_k^G = (H_{(k,1)}^G, ..., H_{(k,D)}^G), k = 1, ..., n$;
Initialize generation count $G = 0$;
while the halting criterion is not satisfied **do**
  for $k = 1$ **to** n **do**
    Calculate λ_k, μ_k and π_k according to fitness values;
  end for
  for $k = 1$ **to** n **do** // migration
    $H_k^{G+1} \leftarrow H_k^G$
    for $d = 1$ **to** D **do**
      if $rand < \lambda_k$ **then**
        Select a habitat H_j^G with probability $\propto \mu_j$;
        $H_{(k,d)}^{G+1} \leftarrow H_{(j,d)}^G$
      end if
    end for
  end for
  for $k = 1$ **to** n **do** // mutation
    for $d = 1$ **to** D **do**
      if $rand < \pi_k$ **then**
        $H_{(k,d)}^{G+1} \leftarrow l_d + rand * (u_d - l_d)$
      end if
    end for
  end for
  Evaluate the fitness values of the habitats;
  Perform elitism and update the best known solutions;
  Increase generation count $G = G + 1$;
end while
return the best solutions

2.2. Covariance matrix based migration

The core of the covariance matrix based migration is the original coordinate system being rotated into an eigenvectorbased one, in which habitants can share their information more efficiently. Fig. 1 illustrates in contour plots the original migration and the covariance matrix based migration, respectively. The migration for BBO can be carried out more efficiently in the eigenvector-based coordinate system.

Let's consider a population H,

$$\left.\begin{aligned} H &= \left[H_1^G, \ldots, H_k^G, \ldots, H_n^G\right]^T \\ H_k^G &= \left[H_{(k,1)}^G, \ldots, H_{(k,j)}^G, \ldots, H_{(k,D)}^G\right] \end{aligned}\right\} k = 1, \ldots, n; j = 1, \ldots, D \quad (4)$$

where H is a $n \times D$ matrix, n the population size, D the number of independent variables, G the generation count, and H_k^G the habitant with index k.

The covariance between the i-th and the j-th dimensions of the population in the G-th generation is defined as below:

$$\mathrm{cov}(i,j) = \frac{\sum_{k=1}^{n} \left(H_{(k,i)}^G - \bar{H}_i^G\right)\left(H_{(k,j)}^G - \bar{H}_j^G\right)}{n-1}, i = 1, \ldots, D; j = 1, \ldots, D \quad (5)$$

where $\bar{H}_i^G = \sum_{k=1}^{n} H_{(k,i)}^G / n$ denotes the means of the variables in the i-th dimension. The covariance matrix $\mathrm{Cov}(H)$ can be defined in terms of the covariance, i.e.,

$$\mathrm{Cov}(H) = [c_{ij}]_{D\times D} = [\mathrm{cov}(i,j)]_{D\times D} \quad (6)$$

In order to compute the eigenvectors, we factorize the covariance matrix $\mathrm{Cov}(H)$ into its canonical form, i.e.,

$$\mathrm{Cov}(H) = Q_H \Lambda_H {Q_H}^T \quad (7)$$

where Q_H is the $D \times D$ matrix that has the eigenvector of $\mathrm{Cov}(H)$ as its i-th column, and Λ_H the diagonal matrix that has the corresponding eigenvalues as its diagonal entries, respectively. Factorizing a matrix into its canonical form is called eigenvalue decomposition.

After the eigenvalue decomposition, the habitant in the original coordinate system can be rotated into the eigenvector-based one as follows.

$$\left.\begin{aligned} \mathrm{eig}H_k^G &= H_k^G \times Q_H \\ \mathrm{eig}H_k^G &= (\mathrm{eig}H_{(k,1)}^G, \ldots, \mathrm{eig}H_{(k,j)}^G, \ldots, \mathrm{eig}H_{(k,D)}^G) \end{aligned}\right\} k = 1, \ldots, n; j = 1, \ldots, D \quad (8)$$

where $\mathrm{eig}H_k^G$ denotes the rotated habitant, and $\mathrm{eig}H_{(k,j)}^G$ the rotated SIV in the eigenvector-based coordinate system.

Let's call the migration in the eigenvectorbased coordinate system as covariance matrix based migration. Now, we can perform the BBO migration in the eigenvector-based coordinate system.

Let's denote the new migrated habitant after applying the covariance matrix based migration as $\mathrm{eig}H_k^{G+1}$. It can then be rotated back into the original coordinate system as follows.

$$H_k^{G+1} = \mathrm{eig}H_k^{G+1} \times Q_H^T \quad (9)$$

To sum up, the covariance matrix based migration (CMM) consists of the eigenvalue decompostition based rotation of the original coordinate system and the migration in the eigenvector-based coordinate system. CMM for the habitant with index k can be formulated as in Algorithm 2.

Algorithm 2. Covariance matrix based migration

Calculate the covariance matrix $Cov(H)$;
Perform the eigenvalue decomposition $Cov(H) = Q_H \Lambda_H {Q_H}^T$;
Rotate H_k^G, $k = 1, ..., n$ into eigenvector-based coordinate system $eigH_k^G = H_k^G * Q_H$, $k = 1, ..., n$;
$eigH_k^{G+1} \leftarrow eigH_k^G$;
for $d = 1$ **to** D **do**
  if $rand < \lambda_k$ **then**
    Select a habitat $eigH_j^G$ with probability $\propto \mu_j$;
    $eigH_{(k,d)}^{G+1} \leftarrow eigH_{(j,d)}^G$;
  end if
end for
Rotate habitant $eigH_i^{G+1}$ back to the original coordinate system $H_k^{G+1} = eigH_k^{G+1} * Q_H^T$;
Return the new migrated habitant H_k^{G+1}.

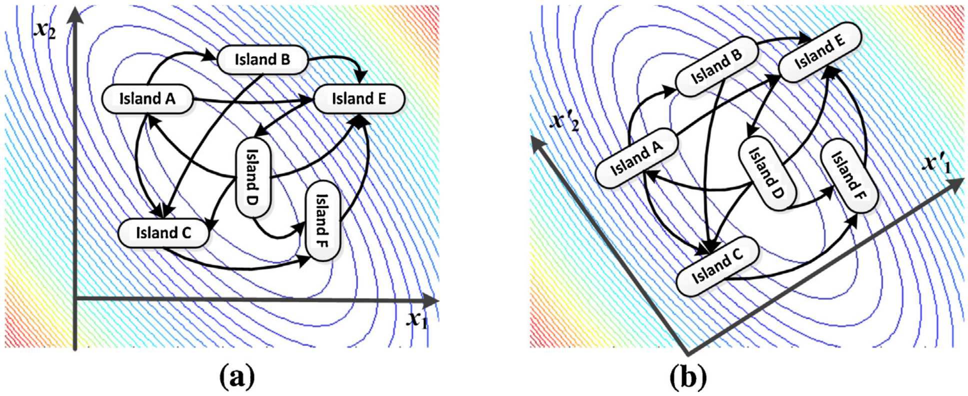

Fig. 1. (a) Original migration (b) Covariance matrix based migration.

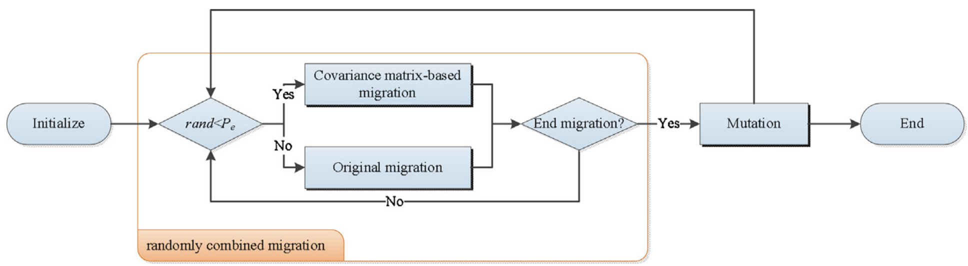

Fig. 2. Framework of CMM-BBO.

2.3. CMM-BBO algorithms

By embedding the covariance matrix based migration into BBO, we can put forward a new BBO approach, namely biogeography-based optimization with covariance matrix based migration, called CMM-BBO. In particular, to prevent ineffective behavior caused by the rotational variance, we devise a parameter P_e in CMM-BBO to control the ratio of CMM to the original migration. The algorithmic structure of CMM-BBO can be illustrated in Fig. 2. Since the switch is based on a random number, i.e., whether rand < P_e or not, the covariance matrix based migration is being randomly combined with the original migration.

It should be pointed out that the BBO in CMM-BBO can be either the basic BBO algorithm or any BBO variant algorithm, and whatever BBO algorithm is used, the framework of CMM-BBO remains the same. Furthermore, normally, $0 < P_e < 1$. There are two extreme cases. When $P_e = 1$, CMM-BBO only uses the covariance matrix based migration to generate offspring. When $P_e = 0$, CMM-BBO reverts to the basic BBO with the original migration. In this sense, our proposed CMM-BBO approach also serves as a unified framework between basic BBO and BBO variant algorithms.

CMM-BBO can be formulated as in Algorithm 3. Various CMM-BBO algorithms can be developed by the CMM operator being randomly combined with the original migration in various BBO algrothims. In this paper, we have selected four existing representative BBO variants, namely, basic real-code BBO(rcBBO) [13], real-code BBO with Gaussian mutation (rcBBOg) [13], perturbation based BBO (pBBO) [16], and BBO hybrid with DE (DE/BBO) [20]. The CMM-BBO algorithms correspondingly developed are denoted as CMM-rcBBO, CMM-rcBBOg, CMM-pBBO, and CMM-DE/BBO, respectively.

Algorithm 3. CMM-BBO

Randomly Initialize a population of n habitats;
while the halting criterion is not satisfied **do**
  for $i = 1$ **to** n **do**
    Calculate λ_i, μ_i, and π_i according to fitness values;
  end for
  for $i = 1$ **to** n **do**
    if $rand < P_e$
      Perform the covariance matrix based migration;
    else
      Perform the original migration;
    end if
  end for
  for $i = 1$ **to** n **do**
    Perform the mutation;
  end for
  Evaluate the fitness values of the habitats;
  Perform elitism and update the best known solutions;
  Increase generation count $G = G + 1$;
end while
return the best solution .

3. Performance evaluations

To conduct the performance evaluations, we employ 37 benchmark functions, as listed in Table 1. The first 23 functions, f01–f23, are the same as in Yao et al. [31], while the rest 14 functions,

Table 1
Benchmark functions for the numeric simulations.

	Name		Dimension	Search space	Optima	Max_FEs
f01	Sphere model	Separable	30	$[-100,100]^D$	0	1.50E+05
f02	Schwefel's problem 2.22	Non-separable	30	$[-10,10]^D$	0	2.00E+05
f03	Schwefel's problem 1.2	Non-separable	30	$[-100,100]^D$	0	5.00E+05
f04	Schwefel's problem 2.21	Non-separable	30	$[-100,100]^D$	0	5.00E+05
f05	Generalized Rosenbrock's functions	Non-separable	30	$[-30,30]^D$	0	5.00E+05
f06	Step function	Separable	30	$[-100,100]^D$	0	1.50E+05
f07	Quartic function	Separable	30	$[-1.28,1.28]^D$	0	3.00E+05
f08	Generalized Schwefel's problem 2.26	Separable	30	$[-500,500]^D$	−12569.5	3.00E+05
f09	Generalized Rastrigin's function	Separable	30	$[-5.12,5.12]^D$	0	3.00E+05
f10	Ackley's function	Separable	30	$[-32,32]^D$	0	1.50E+05
f11	Generalized Griewank function	Separable	30	$[-600,600]^D$	0	2.00E+05
f12	Generalized Penalized function 1	Non-separable	30	$[-50,50]^D$	0	1.50E+05
f13	Generalized Penalized function 2	Non-separable	30	$[-50,50]^D$	0	1.50E+05
f14	Shekel's Foxholes function	Non-separable	2	$[-65.536,65.536]^D$	0.99800383779445	1.00E+04
f15	Kowalik's function	Non-separable	4	$[-5,5]^D$	0.0003075	4.00E+05
f16	Six-Hump Camel-Back function	Non-separable	2	$[-5,5]^D$	−1.03162845348988	1.00E+04
f17	Branin Function	Non-separable	2	$[-5,10]\times[0,15]$	0.397887357729738	1.00E+04
f18	Glodstein-Price function	Non-separable	2	$[-2,2]^D$	2.99999999999992	1.00E+04
f19	Hartman's function 1	Non-separable	3	$[0,1]^D$	−3.86278214782076	1.00E+04
f20	Hartman's function 2	Non-separable	6	$[0,1]^D$	−3.32199517158424	2.00E+04
f21	Shekel's Function 1	Non-separable	4	$[-0,10]^D$	−10.153199679	1.00E+04
f22	Shekel's Function 2	Non-separable	4	$[-0,10]^D$	−10.4029405667869	1.00E+04
f23	Shekel's Function 3	Non-separable	4	$[-0,10]^D$	−10.5364	1.00E+04
F01	Shifted sphere	Separable	30	$[-100,100]^D$	−450	3.00E+05
F02	Shifted Schwefel's problem 1.2	Non-separable	30	$[-100,100]^D$	−450	3.00E+05
F03	Shifted rotated high conditioned elliptic	Non-separable	30	$[-100,100]^D$	−450	3.00E+05
F04	Shifted Schwefel's problem 1.2	Non-separable	30	$[-100,100]^D$	−450	3.00E+05
F05	Schwefel's problem 2.6	Non-separable	30	$[-100,100]^D$	−310	3.00E+05
F06	Shifted Rosenbrock	Non-separable	30	$[-100,100]^D$	390	3.00E+05
F07	Shifted rotated Griewank's function	Non-separable	30	$[0,600]^D$	−180	3.00E+05
F08	Shifted rotated Ackley's function	Non-separable	30	$[-32,32]^D$	−140	3.00E+05
F09	Shifted Rastrigin	Separable	30	$[-100,100]^D$	−330	3.00E+05
F10	Shifted rotated Rastrigin	Non-separable	30	$[-5,5]^D$	−330	3.00E+05
F11	Shifted rotated weierstrass	Non-separable	30	$[-0.5,0.5]^D$	90	3.00E+05
F12	Schwefel's problem 2.13	Non-separable	30	$[-\pi,\pi]^D$	−590	3.00E+05
F13	Expanded extended F8 plus F2	Non-separable	30	$[-3,1]^D$	−130	3.00E+05
F14	Rotated expanded extended Scaffe's F6	Non-separable	30	$[-100,100]^D$	−300	3.00E+05

F01–F14, from the same as in CEC2005 [32]. Functions f01–f04 and F01–F05 are unimodal, while the rest 28 functions are multimodal. Functions f01–f04 are high-dimensional unimodal functions. Function f05 is a multimodal function when $D > 3$. Function f6 is a high-dimensional discontinuous step function with one minimum. Function f7 is a high-dimensional function with noisy perturbation. Functions f8–f13 are high-dimensional multimodal functions where the number of local minima grows exponentially with increased dimensions. Functions f14–f23 are low-dimensional functions with only a few local minima. F06–F12 are basic multimodal functions, and F13–F14 expanded multimodal functions. Among all the functions, 9 of them, i.e., f01, f06–f11, F01 and F09, are separable, while the rest 28 non-separable.

Three performance criteria are adopted from Ref. [32] as follows.

- **Error**: The error of a solution x is defined as $f(x) - f(x*)$, where $x*$ is the global minimum provided in Ref. [33]. The minimum error is recorded when the maximum number of functional evaluations (Max_FEs) is reached in 30 independent runs. The Max_FEs values for the 37 functions are set the same as in Refs. [20,32]. The mean and standard deviation of the errors are calculated for analysis.
- **SR (number of successful runs)**: The successful run of an algorithm manifests the ability of the algorithm to obtain an optimization result no worse than the required accuracy level (RAL) before the search is terminated by the Max_FEs condition. For functions f01–f06, f08–23, and F01–F14, RAL $= 10^{-8}$; for functions f07, RAL $= 10^{-2}$ as in Refs. [20,32].
- **Convergence**: The convergence shows the mean error of the best solution over the total runs, in the respective experiments.

Table 2
Parameter settings of the four existing BBO algorithms.

Algorithm	Parameters
rcBBO	$n = 100$, $I = E = 1$, $\pi_{\max} = 0.005$, elitism parameter $K = 2$
rcBBOg	$n = 100$, $I = E = 1$, $\pi_{\max} = 0.005$, $K = 2$
pBBO	$n = 100$, $I = E = 1$, $\pi_{\max} = 0.005$, $K = 2$
DE/BBO	$n = 100$, $I = E = 1$, $\pi_{\max} = 0.005$, $K = 2$, scaling factor $F = \mathrm{rand}(0.1,1)$, crossover probability CR = 0.9.

The parameter settings of the four existing BBO variants are the same as in their original literature, respectively, as presented in Table 2.

For all the four CMM-BBO algorithms, P_e is set as 0.5. This is based on the thorough sensitivity analysis we have conducted to determine a proper value for parameter P_e. For details, see Appendix A. It should be noted, though, that it is impractical to have a universal optimal setting of parameter P_e as it basically depends upon the specific problems.

3.1. Performances of CMM-BBO algorithms

Table 3 compares the errors between the existing BBO variants and the corresponding CMM-BBO algorithms on the 37 benchmark functions. For each pair of the existing BBO and its CMM-BBO, the better performance in terms of mean error is highlighted in boldface. A nonparametric statistical test, called Wilcoxon's rank-sum test, between the existing BBO and its CMM-BBO is conducted at a 5% significance level, so as to see whether the results obtained with the better performing algorithm significantly exhibit superior

Table 3
Comparison of the errors between the existing BBO variants and the corresponding CMM-BBO algorithms.

Algorithm		rcBBO			CMM-rcBBO		rcBBOg			CMM-rcBBOg	
Function		Mean	SD		Mean	SD	Mean	SD		Mean	SD
f01	Separable	2.10E+00	7.45E−01	+	**4.49E−11**	**2.53E−11**	5.26E−04	2.14E−04	+	**4.81E−15**	**1.83E−15**
f02	Non-separable	3.92E−01	5.59E−02	+	**6.90E−07**	**1.67E−07**	5.14E−02	9.81E−03	+	**7.98E−08**	**2.23E−08**
f03	Non-separable	3.74E+03	1.26E+03	+	**2.04E+00**	**2.69E+00**	2.32E+01	9.12E+00	+	**1.16E+00**	**5.95E−01**
f04	Non-separable	1.39E+00	3.10E−01	+	**6.75E−03**	**2.07E−02**	6.49E−02	2.75E−02	+	**1.12E−02**	**5.38E−03**
f05	Non-separable	1.19E+02	3.89E+01	+	**3.73E+01**	**2.43E+01**	9.31E+01	1.34E+02	=	3.44E+01	2.29E+01
f06	Separable	2.23E+00	1.38E+00	+	**0.00E+00**	**0.00E+00**	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00
f07	Separable	5.49E−03	2.84E−03	+	**2.05E−03**	**7.62E−04**	4.87E−03	1.72E−03	+	**2.08E−03**	**6.38E−04**
f08	Separable	1.52E+00	6.66E−01	+	**1.34E−02**	**3.75E−11**	**4.76E+02**	**2.24E+02**	−	3.14E+03	6.47E+02
f09	Separable	2.41E−01	1.00E−01	+	**8.41E−12**	**6.16E−12**	1.44E−02	7.82E−03	+	**1.22E−13**	**5.96E−14**
f10	Separable	6.05E−01	1.44E−01	+	**1.49E−06**	**4.64E−07**	1.53E−02	2.86E−03	+	**4.35E−08**	**1.53E−08**
f11	Separable	8.12E−01	1.02E−01	+	**2.47E−04**	**1.35E−03**	2.99E−01	2.65E−01	+	**2.05E−03**	**4.30E−03**
f12	Non-separable	1.06E−02	1.13E−02	+	**2.11E−13**	**1.33E−13**	6.26E−01	7.61E−01	+	**4.03E−17**	**3.12E−17**
f13	Non-separable	1.09E−01	2.58E−02	+	**2.80E−12**	**2.39E−12**	1.18E−04	1.23E−04	+	**1.14E−15**	**9.92E−16**
f14	Non-separable	1.43E+00	2.47E+00	+	**7.08E−01**	**1.69E+00**	3.02E+00	4.07E+00	+	**1.59E+00**	**2.55E+00**
f15	Non-separable	2.62E−03	4.87E−03	+	**8.33E−04**	**5.40E−04**	4.11E−03	6.37E−03	+	**6.88E−04**	**5.34E−04**
f16	Non-separable	2.35E−02	2.66E−02	+	**1.26E−04**	**3.00E−04**	1.13E−02	2.24E−02	+	**1.84E−04**	**6.43E−04**
f17	Non-separable	1.19E−02	1.83E−02	+	**1.57E−03**	**2.71E−03**	8.76E−03	1.44E−02	+	**1.08E−03**	**3.02E−03**
f18	Non-separable	2.12E+00	5.50E+00	+	**2.27E−03**	**1.22E−02**	1.34E+00	2.99E+00	+	**2.13E−02**	**1.05E−01**
f19	Non-separable	7.51E−03	9.87E−03	+	**1.03E−04**	**2.56E−04**	1.21E−02	1.37E−02	+	**1.17E−04**	**4.07E−04**
f20	Non-separable	4.97E−02	5.75E−02	+	**3.17E−02**	**5.35E−02**	5.02E−02	6.09E−02	+	**1.98E−02**	**4.51E−02**
f21	Non-separable	5.36E+00	2.93E+00	+	**2.26E+00**	**3.11E+00**	3.91E+00	3.33E+00	+	**1.83E+00**	**3.14E+00**
f22	Non-separable	4.50E+00	2.67E+00	+	**1.43E+00**	**2.92E+00**	3.07E+00	3.13E+00	+	**1.32E+00**	**2.71E+00**
f23	Non-separable	4.91E+00	2.53E+00	+	**1.01E+00**	**2.62E+00**	3.51E+00	3.19E+00	+	**9.73E−01**	**2.53E+00**
F01	Separable	5.71E−01	2.10E−01	+	**3.59E−12**	**2.13E−12**	8.94E−05	4.18E−05	+	**4.73E−16**	**1.94E−16**
F02	Non-separable	6.88E+03	2.59E+03	+	**3.37E+02**	**1.59E+02**	1.88E+02	8.67E+01	+	**3.67E+01**	**1.80E+01**
F03	Non-separable	1.65E+07	7.83E+06	+	**2.78E+06**	**7.70E+05**	3.44E+06	1.47E+06	+	**1.71E+06**	**5.72E+05**
F04	Non-separable	1.67E+04	6.05E+03	+	**1.94E+03**	**3.53E+02**	2.02E+04	8.00E+03	+	**5.04E+03**	**1.89E+03**
F05	Non-separable	6.23E+03	1.15E+03	+	**4.54E+03**	**4.43E+02**	6.42E+03	1.03E+03	+	**5.02E+03**	**8.49E+02**
F06	Non-separable	8.72E+02	2.55E+03	+	**5.40E+02**	**1.64E+03**	4.11E+03	5.76E+03	+	**2.82E+02**	**3.93E+02**
F07	Non-separable	5.34E+03	1.16E+02	+	**3.71E+02**	**1.01E+02**	2.24E+03	7.71E+01	+	**2.50E+02**	**1.01E+02**
F08	Non-separable	2.09E+01	1.04E−01	=	2.09E+01	6.12E−02	2.07E+01	1.29E−01	+	**2.06E+01**	**9.38E−02**
F09	Separable	2.86E−01	1.12E−01	+	**1.49E−11**	**2.63E−11**	1.76E−02	8.89E−03	+	**1.08E−13**	**6.07E−14**
F10	Non-separable	5.12E+01	1.48E+01	=	4.70E+01	1.72E+01	5.92E+01	2.03E+01	=	5.05E+01	1.64E+01
F11	Non-separable	3.23E+01	3.51E+00	+	**1.59E+01**	**2.37E+00**	3.13E+01	3.16E+00	+	**1.51E+01**	**3.20E+00**
F12	Non-separable	**1.66E+00**	**1.01E+00**	−	7.60E+03	7.30E+03	1.95E+04	1.54E+04	+	**1.03E+00**	**1.13E+00**
F13	Non-separable	1.26E+00	3.08E−01	=	1.14E+00	1.80E−01	1.24E+00	1.44E−01	+	**1.11E+00**	**2.17E−01**
F14	Non-separable	1.32E+01	3.90E−01	+	**1.26E+01**	**3.96E−01**	1.36E+01	2.59E−01	+	**1.30E+01**	**5.16E−01**
+/=/−		33/3/1					33/3/1				

Algorithm		pBBO			CMM-pBBO		DE/BBO			CMM-DE/BBO	
Function		Mean	SD		Mean	SD	Mean	SD		Mean	SD
f01	Separable	7.74E−08	3.28E−07	+	**6.31E−11**	**3.18E−11**	9.92E−21	5.31E−21	+	**2.90E−25**	**1.59E−25**
f02	Non-separable	3.32E−05	9.24E−05	+	**7.98E−06**	**2.17E−06**	**2.15E−18**	**6.99E−19**	−	7.82E−17	2.54E−17
f03	Non-separable	2.15E−01	1.11E−01	+	**2.29E−07**	**1.39E−07**	6.32E+02	2.21E+02	+	**1.53E−23**	**4.13E−23**
f04	Non-separable	7.87E−03	2.36E−03	+	**7.50E−06**	**1.40E−06**	1.71E−07	5.25E−08	+	**2.14E−15**	**1.33E−15**
f05	Non-separable	4.55E+01	9.26E+01	=	2.50E+01	1.38E+01	1.80E+01	3.73E−01	+	**2.14E−01**	**5.12E−01**
f06	Separable	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00
f07	Separable	**2.04E−03**	**7.00E−04**	−	9.15E−03	2.97E−03	6.05E−03	1.54E−03	+	**2.66E−03**	**7.46E−04**
f08	Separable	**3.63E+02**	**1.96E+02**	−	1.97E+03	6.34E+02	1.34E−02	0.00E+00	=	1.34E−02	0.00E+00
f09	Separable	4.30E−07	1.17E−06	+	**4.93E−10**	**1.73E−10**	**0.00E+00**	**0.00E+00**	−	2.27E−01	6.74E−01
f10	Separable	3.32E−05	6.85E−05	=	5.45E−06	1.33E−06	2.26E−11	6.14E−12	+	**2.17E−13**	**5.69E−14**
f11	Separable	1.45E−02	2.61E−02	+	**3.29E−04**	**1.80E−03**	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00
f12	Non-separable	4.49E−02	1.35E−01	=	3.46E−03	1.89E−02	2.60E−21	1.39E−21	+	**2.95E−25**	**2.33E−25**
f13	Non-separable	8.01E−09	2.22E−08	=	1.14E−11	5.86E−12	**1.43E−30**	**7.03E−21**	−	4.99E−25	2.86E−25
f14	Non-separable	5.93E−01	1.48E+00	+	**4.61E−01**	**1.36E+00**	5.97E−08	3.99E−16	=	5.97E−08	4.81E−12
f15	Non-separable	1.35E−03	3.56E−03	+	**3.52E−04**	**8.65E−05**	1.27E−05	2.95E−05	+	**−1.40E−08**[a]	**5.16E−20**
f16	Non-separable	2.19E−04	1.06E−03	+	**2.26E−15**	**9.03E−17**	5.16E−11	2.73E−10	=	1.64E−12	3.98E−12
f17	Non-separable	2.10E−05	7.01E−05	+	**3.22E−06**	**1.62E−05**	9.95E−16	1.46E−15	+	**1.67E−16**	**0.00E+00**
f18	Non-separable	7.50E−03	3.66E−02	+	**1.91E−15**	**1.27E−15**	2.30E−14	1.63E−14	+	**7.18E−15**	**7.86E−15**
f19	Non-separable	1.74E−06	6.24E−06	+	**2.55E−07**	**2.93E−09**	2.54E−07	3.88E−15	=	2.54E−07	4.10E−15
f20	Non-separable	4.76E−02	5.92E−02	+	**2.77E−02**	**5.11E−02**	1.59E−02	4.11E−02	=	5.03E−14	2.04E−13
f21	Non-separable	5.09E+00	3.43E+00	+	**2.75E+00**	**3.49E+00**	6.69E−01	1.61E+00	+	**4.96E−07**	**1.39E−06**
f22	Non-separable	3.34E+00	3.43E+00	+	**4.78E−01**	**1.82E+00**	3.03E−04	1.28E−03	+	**1.32E−07**	**2.22E−07**
f23	Non-separable	3.16E+00	3.69E+00	+	**4.04E−01**	**1.55E+00**	1.89E−05	1.43E−04	+	**−9.68E−06**	**1.60E−02**
F01	Separable	6.98E−10	1.52E−09	=	4.21E−12	2.09E−12	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00
F02	Non-separable	4.18E+00	2.15E+00	+	**1.79E−02**	**2.47E−02**	7.30E+02	1.90E+02	+	**3.19E−12**	**7.10E−12**
F03	Non-separable	2.09E+06	8.62E+05	+	**1.03E+06**	**4.65E+05**	1.76E+07	4.92E+06	+	**3.06E+05**	**2.68E+05**
F04	Non-separable	9.48E+02	7.58E+02	+	**3.11E−01**	**2.92E−01**	2.37E+03	6.14E+02	+	**1.02E−04**	**2.70E−04**
F05	Non-separable	4.63E+03	1.00E+03	+	**4.07E+03**	**6.22E+02**	5.01E+02	2.16E+02	+	**1.15E+02**	**2.83E+02**
F06	Non-separable	6.82E+02	2.07E+03	=	3.30E+02	4.62E+02	2.37E+01	1.03E+01	+	**8.80E+00**	**2.84E+00**
F07	Non-separable	1.46E−02	1.47E−02	+	**1.30E−02**	**7.95E−03**	6.57E−04	2.50E−03	+	**0.00E+00**	**0.00E+00**

Table 3 (*Continued*)

Algorithm		pBBO			CMM-pBBO		DE/BBO			CMM-DE/BBO	
Function		Mean	SD		Mean	SD	Mean	SD		Mean	SD
F08	Non-separable	**2.03E+01**	**9.33E−02**	−	2.06E+01	9.00E−02	2.09E+01	4.51E−02	=	2.09E+01	7.29E−02
F09	Separable	8.20E−07	3.89E−06	+	**3.55E−10**	**1.77E−10**	**0.00E+00**	**0.00E+00**	−	5.71E−06	2.87E−05
F10	Non-separable	5.74E+01	1.91E+01	=	5.35E+01	1.60E+01	8.09E+01	9.61E+00	+	**6.95E+01**	**8.78E+00**
F11	Non-separable	2.86E+01	4.22E+00	+	**1.31E+01**	**3.78E+00**	3.07E+01	1.56E+00	+	**2.93E+01**	**1.69E+00**
F12	Non-separable	1.13E+04	8.23E+03	+	**6.06E+03**	**8.52E+03**	2.47E+04	1.24E+04	+	**5.41E+03**	**1.01E+04**
F13	Non-separable	1.08E+00	2.11E−01	=	1.10E+00	1.69E−01	**2.58E+00**	**2.59E−01**	−	3.37E+00	3.44E−01
F14	Non-separable	1.35E+01	2.75E−01	+	**1.31E+01**	**3.49E−01**	1.29E+01	1.82E−01	=	1.28E+01	1.88E−01
+/=/−		25/9/3					22/10/5				

"+ ", "−", and "=" symbolize the performance of the CMM-BBO algorithm being better or worse than, and similar to that of the existing BBO variant, respectively, according to the Wilcoxon rank-sum test at the 5% significance level.

[a] A negative value means that the achieved value is better than the optimal value provided in Table 1.

performance. "+ ", "−", and "=" symbolize the performance of the CMM-BBO algorithm being better or worse than, and similar to that of the existing BBO, respectively.

Compared with rcBBO, of the 37 benchmark functions, CMM-rcBBO achieves significantly better performances on 33 (9 separable and 24 non-separable), not statistically different performances on 3, and poorer performance on only one non-separable function (F12). Therefore, it is fair to say that the CMM operator is able to enhance BBO's capability on non-separable as well as separable functions.

Our proposed CMM operator has remarkably improved the performance of rcBBOg. Compared with rcBBOg, of the 37 benchmark functions, CMM-rcBBOg exhibits significantly better performances on 33 (7 separable and 26 non-separable), not statistically different performances on 3 functions (1 separable and 2 non-separable), and poorer performance on only one separable function f8. What is more, on most of the non-separable functions, CMM-rcBBOg outperforms rcBBOg. In one word, the benefit of the CMM operator for rcBBOg is apparent.

The CMM operator improves the performance of pBBO as well. Compared with pBBO, of the 37 benchmark functions, CMM-pBBO exhibits significantly better performances on 25 functions (4 separable and 21 non-separable), not statistically different performances on 9 functions (3 separable and 6 non-separable) and poorer performances on only the rest 3 functions (2 separable and 1 non-separable).

The performance of DE/BBO is substantially improved by the CMM operator. Compared with DE/BBO, of the 37 benchmark functions, CMM-DE/BBO achieves statistically significant improvements on 22 functions (3 separable and 19 non-separable), not statistically different performances on 10 functions (4 separable and 6 non-separable), and poorer performances on only 5 functions (2 separable and 3 non-separable).

Table 4 presents the SR values of all the existing BBO variants and the corresponding CMM-BBO algorithms on the 37 benchmark functions. "+ ", "−", and "=" symbolize the SR value of the CMM-BBO algorithm being better or worse than, and similar to that of the existing BBO variant, respectively. As shown in Table 4, the CMM operator has improved the performances of all the existing four BBO variants in terms of SR values on most of the benchmark functions. This manifests that the CMM-BBO algorithms are more effective than the existing BBO variants.

In addition, Table 5 summarizes the multiple-problem Wilcoxon signed-rank test for the CMM-BBO algorithms and the existing BBO variants on all the 37 benchmark functions. The Wilcoxon signed-rank test was conducted in the KEEL software [34]. In Table 6, *R*+ is the sum of ranks for the functions in which the first algorithm outperforms the second, and *R*− the sum of ranks for the opposite. *p*-vlaue is the smallest level of significance. As shown in Table 6, all the CMM-BBO algorithms attain higher *R*+ values than *R*− values. Furthermore, the *p* values in all cases are less than 0.05, which means that the CMM-BBO algorithms are significantly better than the existing BBO variants.

To establish the rankings across all the eight BBO algorithms, i.e., the four existing BBO variants and their corresponding four CMM-BBO algorithms, on the 37 benchmark functions, the Friedman test is carried out, in which Bonferroni–Dunn's procedure was used as a post hoc procedure. As shown in Table 6, CMM-DE/BBO ranks first, followed in order by CMM-pBBO, DE/BBO, and CMM-rcBBOg.

Fig. 3 plots the convergence graphs of the eight BBO algorithms on 8 selected functions, namely f01, f03, f10, f12, F02, F07, F11, and F14. The convergence graphs depict the mean error curves of all the BBO algorithms in our numeric simulations over 30 independent runs. Overall, our developed CMM-BBO algorithms converge faster than the existing BBO variants on most of the benchmark functions.

3.2. Comparison of CMM-DE/BBO with other EAs

As shown above, across the eight BBO algorithms, CMM-DE/BBO performs the best. In this sub-section, we will further evaluate CMM-BBO by comparing CMM-DE/BBO with six other state-of-the-art EAs, namely CMAES [2], jDE [3], SaDE [4], JADE [5], CLPSO [6], and DMSPSO [7]. The Matlab source codes of CMAES, jDE, SaDE, JADE, and CLPSO were downloaded from Q. Zhang's website "http://dces.essex.ac.uk/staff/qzhang." The Matlab source code of DMSPSO was provided by P. N. Suganthan.

CMAES, proposed by Hansen and Ostermeier [2], is an evolution strategy (ES) based on completely derandomized self-adaptation. jDE, SaDE, and JADE are three representative DE algorithms. jDE, proposed by Brest et al. [3], is a DE with self-adaptive control parameter. SaDE, proposed by Qin et al. [4], gradually self-adapts both mutation strategies and their associated control parameters through learning from the previous experiences in generating promising solutions. JADE, proposed by Zhang and Sanderson [5], employs a new mutation strategy "DE/current-to-pbest" with optional external archive and updates control parameters in an adaptive manner.

CLPSO and DMSPSO are two representative PSO algorithms. CLPSO, proposed by Liang et al. [6], uses all other particles' historical best information to update a particle's velocity. DMSPSO, proposed by Liang and Suganthan [7], uses dynamic multi-swarm topology to balance the exploration and exploitation.

The parameter settings of the six EAs are the same as in their original literature, respectively, as presented in Table 7.

All the algorithms are evaluated on the 37 benchmark functions over 30 independent runs. Table 8

compares the errors of all the algorithms. The multiple-problem Wilcoxon signed-rank test is presented in Table 9. In addition, the rankings of the EAs according to the Friedman test are presented in Table 10. As shown in Table 10, performances of CMM-DE/BBO

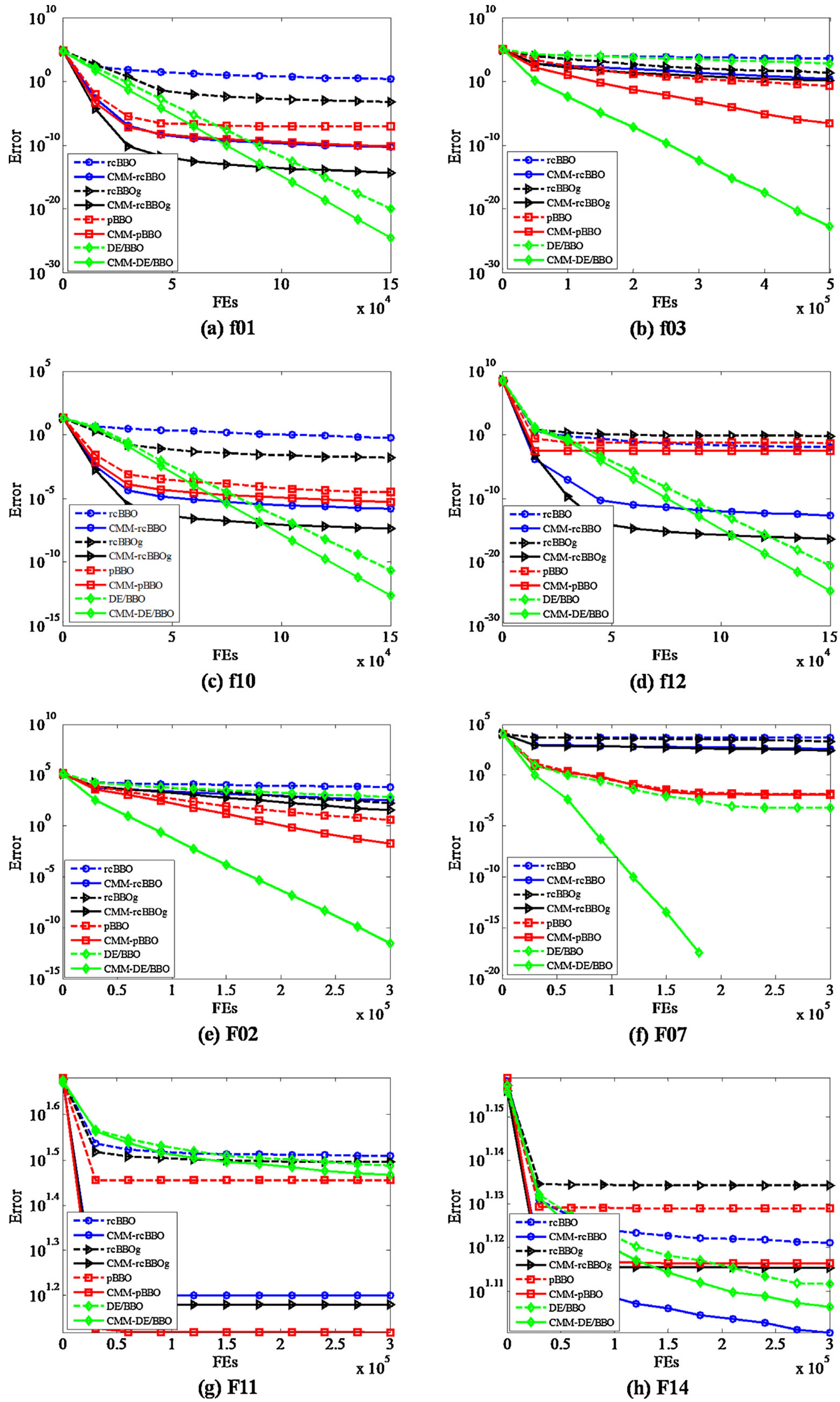

Fig. 3. Convergence graphs (mean curves) of all the eight BBO algorithms on f01, f03, f10, f12, F02, F07, F11, and F14.

Table 4
Comparison of SR values between the existing BBO variants and the corresponding CMM-BBO algorithms.

Algorithm		rcBBO	CMM-rcBBO		rcBBOg	CMM-rcBBOg		pBBO	CMM-pBBO		DE/BBO	CMM-DE/BBO	
Functions		SR	SR		SR	SR		SR	SR		SR	SR	
f01	Separable	0	30	+	0	30	+	23	30	+	30	30	=
f02	Non-separable	0	0	=	0	0	=	6	0	–	30	30	=
f03	Non-separable	0	0	=	0	0	=	0	0	=	0	30	+
f04	Non-separable	0	0	=	0	0	=	0	0	=	0	30	+
f05	Non-separable	0	0	=	0	0	=	0	0	=	0	15	+
f06	Separable	2	30	+	30	30	=	30	30	=	30	30	=
f07	Separable	28	30	+	30	30	=	30	20	–	30	30	=
f08	Separable	0	0	=	0	0	=	0	0	=	0	0	=
f09	Separable	0	30	+	0	30	+	16	30	+	30	6	–
f10	Separable	0	0	=	0	0	=	6	0	–	30	30	=
f11	Separable	0	29	+	0	24	+	18	29	+	30	30	=
f12	Non-separable	0	30	+	0	30	+	24	29	+	30	30	=
f13	Non-separable	0	30	+	0	30	+	26	30	+	30	30	=
f14	Non-separable	0	0	+	0	0	=	0	0	=	0	0	=
f15	Non-separable	0	0	=	0	0	=	0	0	=	11	30	+
f16	Non-separable	0	5	+	0	15	+	19	30	+	30	30	=
f17	Non-separable	0	1	+	0	1	+	10	23	+	30	30	=
f18	Non-separable	0	24	+	0	17	+	21	30	+	30	30	=
f19	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
f20	Non-separable	0	21	+	0	25	+	10	23	+	26	30	+
f21	Non-separable	0	19	+	0	21	+	5	18	+	0	0	=
f22	Non-separable	0	24	+	0	24	+	7	28	+	3	1	–
f23	Non-separable	0	26	+	0	26	+	13	28	+	27	30	+
F01	Separable	0	30	+	0	30	+	30	30	=	30	30	=
F02	Non-separable	0	0	=	0	0	=	0	0	=	0	30	+
F03	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F04	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F05	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F06	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F07	Non-separable	0	0	=	0	0	=	0	0	=	14	30	+
F08	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F09	Separable	0	30	+	0	30	+	17	30	+	30	22	–
F10	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F11	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F12	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F13	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
F14	Non-separable	0	0	=	0	0	=	0	0	=	0	0	=
+/=/–		17/20/0			14/23/0			13/21/3			8/26/3		

"+ ", "–", and "=" symbolize the SR value of the CMM-BBO algorithms being better or worse than, and similar to that of the exisiting BBO variants, respectively.

Table 5
Multiple-problem Wilcoxon test for the CMM-BBO algorithms and the existing BBO variants.

Algorithm	*R*+	*R*–	*p*-value	$\alpha = 0.05$	$\alpha = 0.1$
CMM-rcBBO vs rcBBO	632	34	1.08E – 07	Yes	Yes
CMM-rcBBOg vs rcBBOg	634	32	1.65E – 07	Yes	Yes
CMM-pBBO vs pBBO	579	87	3.97E – 05	Yes	Yes
CMM-DE/BBO vs DE/BBO	572.5	93.5	6.90E – 05	Yes	Yes

R+ is the sum of ranks for the functions in which the first algorithm outperforms the second, and *R*– the sum of ranks for the opposite. *p*-vlaue is the smallest level of significance. Yes means significant difference and No means no significant difference at the given significance level.

Table 6
Average ranking of existing BBO variants and the CMM-BBO algorithms according to the Friedman tests.

Algorithm	Average ranking	Final rank
rcBBO	7.0946	8
CMM-rcBBO	4.0135	5
rcBBOg	6.7297	7
CMM-rcBBOg	3.9324	4
pBBO	4.8243	6
CMM-pBBO	3.3784	2
DE/BBO	3.4865	3
CMM-DE/BBO	2.5405	1

Table 7
Parameter settings of the six EAs.

Algorithm	Parameters
CMAES	Number of offsprings $\lambda = 4$ +floor(3log(D)), number of parents μ = floor($\lambda/2$);
jDE	Population size NP = 100, $\tau_1 = \tau_2 = 0.1$, $F_l = 0.1$, $F_u = 0.9$;
SaDE	NP = 50, learning period LP = 50;
JADE	NP = 100, $c = 0.1$, $p = 0.05$;
CLPSO	Population size ps = 40, inertia weight w linearly decreasing from 0.9 to 0.2, acceleration coefficients $c = 1.494$, refreshing gap $m = 5$;
DMSPSO	ps = 40, $w = 0.729$, $c_1 = c_2 = 1.496$, population size of sub-swarm $m = 5$, regrouping period $R = 5$.

are significantly better than CMAES, jDE, SaDE, JADE, CLPSO, and DMSPSO on 24, 14, 14, 9, 28, and 21 functions, respectively; and similar to those of CMAES, jDE, SaDE, JADE, CLPSO, and DMSPSO on 3, 9, 7, 7, 4 and 6 functions, respectively. However, the performances of CMM-DE/BBO are significantly worse than those of CMAES, jDE, SaDE, JADE, CLPSO, and DMSPSO on 10, 14, 16, 21, 5, and 10 functions, respectively. According to the multiple-problem Wilcoxon signed-rank test, CMM-DE/BBO attains higher positive-ranks (*R*+)

Table 8
Comparison of CMM-DE/BBO with other evolutionary algorithms.

	CMM-DE/BBO		CMAES			jDE			SaDE		
Functions	Mean	SD	Mean	SD		Mean	SD		Mean	SD	
f01	2.90E−25	1.59E−25	5.80E−29	1.33E−29	−	1.10E−28	8.81E−29	−	1.42E−64	3.36E−64	−
f02	7.82E−17	2.54E−17	9.35E−02	4.04E−01	+	1.38E−23	8.79E−24	−	7.22E−53	7.46E−53	−
f03	1.53E−23	4.13E−23	1.57E−26	2.45E−27	−	2.87E−14	4.44E−14	+	6.34E−14	7.89E−14	+
f04	2.14E−15	1.33E−15	4.13E−15	6.01E−16	+	2.08E−01	3.96E−01	+	5.48E−21	2.89E−20	−
f05	2.14E−01	5.12E−01	1.33E−01	7.28E−01	−	1.43E−01	7.28E−01	−	2.46E+01	2.14E+01	+
f06	0.00E+00	0.00E+00	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	=
f07	2.66E−03	7.46E−04	1.97E−01	7.32E−02	+	3.64E−03	6.94E−04	+	2.33E−03	9.15E−04	=
f08	1.34E−02	0.00E+00	5.64E+03	5.31E+02	+	1.34E−02	0.00E+00	=	1.34E−02	0.00E+00	=
f09	2.27E−01	6.74E−01	2.19E+02	6.27E+01	+	0.00E+00	0.00E+00	−	6.63E−02	2.52E−01	−
f10	2.17E−13	5.69E−14	1.95E+01	1.73E−01	+	7.22E−15	6.49E−16	−	6.21E−02	2.36E−01	+
f11	0.00E+00	0.00E+00	1.31E−03	3.02E−03	+	0.00E+00	0.00E+00	=	4.11E−03	7.18E−03	+
f12	2.95E−25	2.33E−25	6.91E−03	2.63E−02	+	1.00E−29	1.36E−29	−	6.91E−03	2.63E−02	+
f13	4.99E−25	2.86E−25	3.66E−04	2.01E−03	+	6.47E−29	8.41E−29	−	1.35E−32	0.00E+00	−
f14	5.97E−08	4.81E−12	1.20E+01	6.97E+00	+	5.97E−08	2.72E−16	−	1.11E−02	6.06E−02	+
f15	−1.40E−08	5.16E−20	4.22E−03	8.06E−03	+	−1.40E−08	6.42E−20	+	−1.40E−08	7.27E−13	=
f16	1.64E−12	3.98E−12	3.99E−01	8.21E−01	+	7.49E−13	1.65E−12	=	2.24E−15	6.78E−17	−
f17	1.67E−16	0.00E+00	1.67E−16	0.00E+00	=	1.67E−16	0.00E+00	=	1.67E−16	0.00E+00	=
f18	7.18E−15	7.86E−15	1.89E+01	3.48E+01	+	4.88E−15	2.75E−15	=	2.09E−15	1.19E−15	−
f19	2.54E−07	4.10E−15	1.91E−01	7.26E−01	+	2.54E−07	1.42E−15	−	2.54E−07	2.13E−16	−
f20	5.03E−14	2.04E−13	2.77E−02	5.11E−02	+	1.59E−02	4.11E−02	+	7.93E−03	3.02E−02	+
f21	4.96E−07	1.39E−06	4.76E+00	3.30E+00	+	3.44E−04	1.18E−03	+	−5.67E−11	4.51E−12	−
f22	1.32E−07	2.22E−07	4.65E+00	3.44E+00	+	1.26E−02	6.73E−02	+	−3.12E−11	3.11E−12	−
f23	−9.68E−06	1.60E−02	5.52E+00	3.52E+00	+	6.38E−06	5.90E−05	+	−9.82E−06	4.69E−10	−
F01	0.00E+00	0.00E+00	1.84E−25	4.39E−26	+	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	=
F02	3.19E−12	7.10E−12	6.50E−25	1.86E−25	−	1.29E−06	1.75E−06	+	4.47E−06	8.77E−06	+
F03	3.06E+05	2.68E+05	5.09E−21	1.26E−21	−	1.71E+05	1.10E+05	−	4.36E+05	1.73E+05	+
F04	1.02E−04	2.70E−04	3.13E+05	9.11E+05	+	1.34E−02	1.21E−02	+	1.02E+02	1.47E+02	+
F05	1.15E+02	2.83E+02	3.13E−10	8.99E−11	−	3.00E+02	3.41E+02	+	3.18E+03	7.95E+02	−
F06	8.80E+00	2.84E+00	6.64E−01	1.51E+00	−	2.50E+01	2.72E+01	+	4.78E+01	3.91E+01	+
F07	0.00E+00	0.00E+00	3.61E−03	6.36E−03	+	4.70E+03	5.34E−13	+	1.51E−02	1.06E−02	−
F08	2.09E+01	7.29E−02	2.03E+01	5.60E−01	−	2.09E+01	5.73E−02	=	2.10E+01	4.10E−02	+
F09	5.71E−06	2.87E−05	3.74E+02	1.22E+02	+	0.00E+00	0.00E+00	−	6.63E−02	2.52E−01	+
F10	6.95E+01	8.78E+00	4.61E+01	1.17E+01	−	5.57E+01	8.83E+00	−	4.79E+01	9.64E+00	−
F11	2.93E+01	1.69E+00	6.26E+00	2.66E+00	−	2.79E+01	2.24E+00	−	1.73E+01	2.76E+00	−
F12	5.41E+03	1.01E+04	1.16E+04	1.25E+04	+	5.13E+03	6.19E+03	=	3.23E+03	3.15E+03	=
F13	3.37E+00	3.44E−01	3.42E+00	8.64E−01	=	1.72E+00	1.51E−01	−	4.03E+00	3.71E−01	+
F14	1.28E+01	1.88E−01	1.47E+01	2.95E−01	+	1.30E+01	1.71E−01	+	1.27E+01	2.35E−01	−
+=/−			24/3/10			14/9/14			14/7/16		

JADE			CLPSO			DMSPSO		
Mean	SD		Mean	SD		Mean	SD	
1.05E−56	5.51E−56	−	6.70E−11	2.93E−11	+	7.87E−47	1.05E−46	−
5.10E−25	2.75E−24	−	3.75E−10	1.15E−10	+	3.80E−36	8.78E−36	−
1.06E−85	5.50E−85	−	6.25E+01	2.00E+01	+	4.46E−09	5.71E−09	+
5.77E−66	1.48E−65	−	4.63E−01	7.06E−02	+	2.86E−13	5.21E−13	+
9.92E−31	2.91E−30	−	2.36E−01	2.25E−01	+	4.41E+00	2.70E+00	+
0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	=	0.00E+00	0.00E+00	=
6.34E−04	2.40E−04	−	3.20E−03	7.26E−04	+	2.79E−03	7.36E−04	=
1.34E−02	0.00E+00	=	1.34E−02	4.61E−13	=	3.92E+03	7.26E+02	+
0.00E+00	0.00E+00	−	0.00E+00	0.00E+00	−	2.43E+01	6.61E+00	+
3.67E−15	6.49E−16	−	5.32E−06	1.36E−06	+	6.28E−15	1.53E−15	−
0.00E+00	0.00E+00	=	4.53E−11	5.47E−11	+	5.50E−03	8.32E−03	+
1.57E−32	5.57E−48	−	2.87E−12	1.44E−12	+	3.46E−03	1.89E−02	+
1.35E−32	5.57E−48	−	7.31E−11	3.35E−11	+	2.20E−03	4.47E−03	+
5.97E−08	1.11E−14	=	5.97E−08	2.22E−11	+	1.33E−01	1.65E−01	=
−1.40E−08[a]	1.32E−19	=	2.07E−04	8.23E−05	+	−1.40E−08	1.77E−19	+
1.29E−09	3.83E−09	+	4.86E−10	1.11E−09	+	4.78E−12	2.23E−11	+
7.59E−16	1.26E−15	+	9.45E−12	3.73E−11	+	1.67E−16	0.00E+00	=
3.75E−14	1.93E−14	+	2.14E−11	7.38E−11	+	3.89E−15	1.62E−15	=
2.54E−07	5.02E−14	+	2.54E−07	4.67E−11	+	2.54E−07	0.00E+00	−
1.98E−02	4.51E−02	+	2.54E−06	4.95E−06	+	3.96E−03	2.17E−02	+
1.72E−01	9.22E−01	+	2.38E−01	4.70E−01	+	5.00E−01	1.90E+00	+
7.69E−04	3.60E−03	+	3.46E−01	7.73E−01	+	−3.07E−11	5.41E−12	−
3.32E−04	1.46E−03	+	3.69E−01	9.99E−01	+	−9.82E−06	4.30E−14	−
0.00E+00	0.00E+00	=	6.02E−26	3.94E−26	+	1.68E−30	9.22E−30	=
1.06E−28	9.88E−29	−	1.02E+03	2.31E+02	+	5.10E−04	6.98E−04	+
8.26E+03	7.17E+03	−	1.64E+07	4.58E+06	+	1.09E+06	4.51E+05	+
9.42E−15	4.55E−14	−	6.63E+03	1.69E+03	+	7.18E+01	5.89E+01	+
6.71E−08	2.59E−07	−	4.15E+03	3.93E+02	+	2.84E+03	5.01E+02	+
5.36E+00	1.98E+01	−	2.16E+00	5.08E+00	−	2.04E+01	2.46E+01	+
6.08E−03	5.28E−03	+	9.83E−01	6.04E−02	+	4.70E+03	1.53E−12	+
2.09E+01	1.84E−01	=	2.09E+01	5.88E−02	=	2.08E+01	5.82E−02	−
0.00E+00	0.00E+00	−	0.00E+00	0.00E+00	−	4.22E+01	9.79E+00	+

Table 8 (*Continued*)

JADE			CLPSO			DMSPSO		
Mean	SD		Mean	SD		Mean	SD	
2.32E+01	5.04E+00	–	1.14E+02	1.34E+01	+	5.98E+01	1.17E+01	–
2.54E+01	1.76E+00	–	2.60E+01	1.96E+00	–	1.39E+01	3.00E+00	–
5.41E+03	4.40E+03	–	1.51E+04	5.51E+03	+	6.90E+03	7.20E+03	+
1.50E+00	9.68E−02	–	1.93E+00	1.83E−01	–	4.74E+00	1.46E+00	+
1.23E+01	3.15E−01	–	1.28E+01	1.87E−01	=	1.18E+01	3.85E−01	–
9/7/21			28/4/5			21/6/10		

"+", "–", and "=" symbolize the performance of the CMM-DE/BBO algorithm being better or worse than, and similar to that of the EA, respectively, according to the Wilcoxon rank-sum test at the 5% significance level.

[a] A negative value means that the achieved value is better than the optimal value provided in Table 1.

Table 9
Multiple-problem Wilcoxon signed-rank test for CMAES, jDE, SaDE, JADE, CLPSO, DMSPSO, and CMM-DE/BBO.

Algorithm	*R*+	*R*−	*p*-value	$\alpha = 0.05$	$\alpha = 0.1$
CMM-DE/BBO vs CMAES	499.5	203.5	2.48E-2	Yes	Yes
CMM-DE/BBO vs jDE	331	335	≥0.2	No	No
CMM-DE/BBO vs SaDE	421	282	≥0.2	No	No
CMM-DE/BBO vs JADE	190	476	≥0.2	No	No
CMM-DE/BBO vs CLPSO	565.5	137.5	8.54E-04	Yes	Yes
CMM-DE/BBO vs DMSPSO	540	163	3.71E-03	Yes	Yes

R+ is the sum of ranks for the functions in which the first algorithm outperforms the second, and *R*− the sum of ranks for the opposite. *p*-vlaue is the smallest level of significance. Yes means significant difference and No means no significant difference at the given significance level.

Table 10
Average ranking of CMAES, jDE, SaDE, JADE, CLPSO, DMSPSO, and CMM-DE/BBO according to the Friedman tests.

	Average ranking	Final rank
CMM-DE/BBO	3.3784	2
CMAES	4.9324	7
jDE	3.7432	3
SaDE	4.027	4
JADE	3.0135	1
CLPSO	4.4865	6
DMSPSO	4.4189	5

than CMAES, CLPSO and DMSPSO, and there are significant differences among these algorithms when $\alpha = 0.05$ and $\alpha = 0.1$. There are no significant differences among CMM-DE/BBO, jDE, SaDE, and JADE when $\alpha = 0.05$ and $\alpha = 0.1$. As shown in Table 10, JADE ranks the first, and CMM-DE/BBO the second, followed by jDE, SaDE, DMSPSO, CLPSO, and CMAES. Therefore, it is fair to say that CMM-DE/BBO is an effective BBO variant, thanks to the CMM operator.

4. Discussions, conclusions and future work

BBO is a new bio-inspired EA which has proven its quality and versatility on a wide range of optimization problems. However, the single-feature-migration of BBO leaves it with heavy dependence upon the coordinate system, and poor performance when applied to non-separable problems.

To address this drawback of BBO, in this paper we have proposed the covariance matrix based migration (CMM) to relieve BBO's dependence upon the coordinate system so that BBO's rotational invariance is enhanced. By use of our proposed CMM operator, the original coordinate system is rotated into an eigenvector-based one, in which habitants can share their information more efficiently.

By embedding the CMM into BBO, we have put forward a new BBO approach, namely biogeography-based optimization with covariance matrix based migration, called CMM-BBO. Specifically, four CMM-BBO algorithms, namely, CMM-rcBBO, CMM-rcBBOg, CMM-pBBO, and CMM-DE/BBO, have been developed by the CMM operator being randomly combined with the original migration in the four selected existing BBO variants.

While our proposed CMM operator looks like the eigenvector-based crossover operator in DE [29,30], there are important differences between the two. Firstly, the eigenvector-based crossover operator is designed for DE, while our CMM operator is for BBO. Secondly, the eigenvector-based crossover operator only utilizes the information of two individuals, i.e., one rotated parent individual and its corresponding child individual; but in our CMM operator, the generated individual can obtain the information from all the rotated habitants based on BBO migration. Thirdly, the core operator in DE is the mutation, not the crossover; while the core operator in BBO is the migration. Therefore, the CMM operator would impact upon BBO substantially more profoundly than the eigenvector-based crossover operator would upon DE.

The comprehensive numeric results[1] we have carried out have shown that our proposed CMM-BBO approach significantly improves the performances of the existing BBO algorithms on most of the benchmark functions.

By now we can rightly draw the following conclusions.

i. The covariance matrix based migration (CMM) significantly enhances the rotational invariance of BBO.
ii. The CMM operator can be easily applied to existing BBO variants. It can be embedded into any BBO variants without the framework of CMM-BBO having necessarily to be changed. Therefore, CMM-BBO approach serves as a unified framework for all kinds of BBO variants.
iii. The CMM-BBO approach effectively improves the performances of the existing BBO variants on non-separable as well as separable problems. Overall, the CMM-BBO algorithms have better performances than the existing BBO variants in terms of the performance criteria of error, SR, and convergence.
iv. According to the Friedman test, the CMM-DE/BBO algorithm ranks the first across all the eight BBO algorithms, i.e., rcBBO, rcBBOg, pBBO, DE/BBO, and CMM-rcBBO, CMM-rcBBOg, CMM-pBBO, CMM-DE/BBO.
v. Compared with six other representative EAs, including CMAES, jDE, SaDE, JADE, CLPSO and DMSPSO, CMM-DE/BBO achieves

[1] The source codes of the CMM-BBO algorithms are available from the first author upon request.

highly competitive results on the 37 benchmark functions. According to the Friedman test, CMM-DE/BBO ranks the second, only after JADE.

Several aspects may be worth exploring in the future work. Firstly, adaptive or self-adaptive adjusting mechanism may be designed for control parameter *Pe* and its impact on performance may be studied. Secondly, Markov theory may be applied for analysis of CMM-BBO algorithms. Thirdly, computation of the covariance matrix is time-consuming in large-scale problems. How to increase the computational efficiency of CMM-BBO algorithms would be of significance for large-scale optimization problems. Finally, it is also interesting to apply CMM-BBO to challenging real-world problems.

Acknowledgments

The first author would like to acknowledge the supports by the Research Talents Startup Foundation of Jiangsu University (Grant No. 15JDG139), and the Priority Academic Program Development of Jiangsu Higher Education Institutions.

The authors would like to thank Dr. N. Hansen, Dr. J. Brest, and Dr. J. Zhang for providing the source codes of CMAES, jDE, and JADE, and Dr. P. N. Suganthan for providing the source codes of SaDE, CLPSO, and DMSPSO.

Appendix A. Sensitivity analysis of parameter P_e.

In order to analyze the sensitivity of parameter P_e, we have evaluated the four CMM-BBO algorithms with different P_e: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0. The sensitivity analysis of parameterP_e is detailed in Tables A1–A4.

Table A5 presents the rankings of the four CMM-BBO algorithms with different P_e according to the Friedman test on all the 37 benchmark functions. As shown in Table A5, for CMM-rcBBO, the best ranking is achieved when P_e is 0.7, followed by 0.5 and 0.4. For CMM-rcBBOg, the best ranking is achieved when P_e is 0.5, followed by 0.4 and 0.7. For CMM-pBBO, the best ranking occurs when $P_e = 0.5$, followed by $P_e = 0.7$ and $P_e = 0.4$. For CMM-DE/BBO, the best ranking occurs when P_e is 0.5, followed by 0.7 and 0.4.

Considering both the benchmark functions and the four CMM-BBO algorithms, $P_e = 0.5$ seems to be the common appropriate value.

Table A1
Mean errors of CMM-rcBBO with different P_e on the 37 benchmark functions.

P_e	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f02	3.92E−01	7.06E−08	5.02E−12	3.59E−09	1.10E−07	6.90E−07	3.26E−06	7.67E−06	1.66E−05	2.74E−05	4.23E−05
f03	3.74E+03	6.62E+02	1.07E+02	3.03E+01	8.79E+00	2.04E+00	5.01E−01	1.36E−01	7.42E−03	2.70E−04	**1.15E−04**
f04	1.39E+00	6.07E−01	3.70E−01	9.96E−02	3.69E−02	6.75E−03	7.07E−03	**9.90E−05**	3.44E−04	2.08E−04	9.61E−04
f05	1.19E+02	6.31E+01	6.11E+01	4.73E+01	4.52E+01	3.73E+01	3.87E+01	2.73E+01	2.88E+01	3.28E+01	**2.98E+01**
f06	2.23E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f07	5.49E−03	**9.42E−04**	1.26E−03	1.58E−03	2.08E−03	2.05E−03	2.19E−03	2.29E−03	2.54E−03	2.71E−03	2.77E−03
f08	1.52E+00	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**
f09	2.41E−01	6.44E−12	**0.00E+00**	**0.00E+00**	1.15E−13	8.41E−12	1.32E−10	1.12E−09	3.83E−09	1.91E−08	2.52E−08
f10	6.05E−01	4.68E−05	**8.26E−11**	1.52E−08	2.24E−07	1.49E−06	4.79E−06	1.06E−05	1.92E−05	3.17E−05	5.43E−05
f11	8.12E−01	1.16E−02	9.04E−04	3.29E−04	6.57E−04	**2.47E−04**	**2.47E−04**	2.70E−03	1.15E−03	8.76E−03	7.47E−03
f12	1.06E−02	5.36E−09	**2.51E−20**	2.78E−17	6.62E−15	2.11E−13	1.61E−12	1.65E−11	5.43E−11	1.23E−10	4.82E−10
f13	1.09E−01	7.56E−04	**1.01E−18**	1.10E−14	1.31E−13	2.80E−12	3.66E−11	2.16E−10	1.24E−09	1.96E−09	5.43E−09
f14	1.43E+00	1.84E+00	1.22E+00	9.66E−01	**5.96E−01**	7.08E−01	7.36E−01	8.51E−01	1.19E+00	1.34E+00	1.97E+00
f15	2.62E−03	2.04E−03	1.39E−03	1.45E−03	8.45E−04	8.33E−04	**6.24E−04**	6.31E−04	**6.24E−04**	1.19E−03	9.47E−04
f16	2.35E−02	7.68E−03	1.60E−03	4.56E−04	1.57E−04	1.26E−04	2.05E−04	3.01E−05	**2.14E−05**	5.36E−05	4.27E−04
f17	1.19E−02	3.29E−03	3.75E−03	2.40E−03	1.19E−03	1.57E−03	8.64E−04	**8.04E−04**	1.20E−03	1.43E−03	1.99E−03
f18	2.12E+00	3.06E−01	6.95E−02	2.09E−02	1.02E−01	2.27E−03	1.34E−03	**4.14E−05**	3.67E−04	8.27E−05	5.28E−02
f19	7.51E−03	9.95E−04	2.23E−04	1.39E−04	6.28E−05	1.03E−04	7.60E−05	**5.68E−05**	1.17E−04	7.46E−05	1.99E−04
f20	4.97E−02	2.44E−02	3.97E−02	2.38E−02	5.55E−02	3.17E−02	3.96E−02	2.38E−02	2.77E−02	**1.59E−02**	2.01E−02
f21	5.36E+00	3.93E+00	3.09E+00	2.76E+00	1.75E+00	2.26E+00	1.58E+00	1.50E+00	7.58E−01	**2.62E−01**	5.44E−01
f22	4.50E+00	3.01E+00	1.33E+00	2.04E+00	8.92E−01	1.43E+00	1.67E+00	7.00E−01	8.32E−01	**4.65E−01**	1.03E+00
f23	4.91E+00	2.66E+00	2.11E+00	4.80E−01	9.73E−01	1.01E+00	1.15E+00	2.70E−01	6.59E−01	**2.55E−01**	1.01E+00
F01	5.71E−01	**2.39E−18**	8.21E−22	1.48E−16	8.41E−14	3.59E−12	4.07E−11	2.36E−10	8.71E−10	2.63E−09	6.14E−09
F02	6.88E+03	2.87E+03	1.12E+03	8.22E+02	5.38E+02	3.37E+02	2.41E+02	1.36E+02	1.27E+02	7.94E+01	**7.09E+01**
F03	1.65E+07	6.22E+06	4.81E+06	3.54E+06	3.09E+06	2.78E+06	2.69E+06	2.72E+06	2.37E+06	**2.18E+06**	3.26E+06
F04	1.67E+04	7.39E+03	4.44E+03	3.03E+03	2.40E+03	1.94E+03	1.50E+03	9.78E+02	9.37E+02	8.13E+02	**7.08E+02**
F05	6.23E+03	4.20E+03	**3.91E+03**	4.14E+03	4.56E+03	4.54E+03	4.97E+03	5.35E+03	6.00E+03	7.94E+03	1.06E+04
F06	8.72E+02	1.66E+03	5.85E+02	6.04E+02	4.61E+02	5.40E+02	3.97E+02	3.45E+02	**2.02E+02**	3.74E+02	3.45E+02
F07	5.34E+03	8.76E−01	**3.54E−01**	4.55E+00	7.61E+01	3.71E+02	6.06E+02	9.35E+02	1.26E+03	1.58E+03	2.53E+03
F08	2.09E+01	**2.08E+01**	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01	2.09E+01
F09	2.86E−01	5.67E−11	**0.00E+00**	**0.00E+00**	1.34E−13	1.49E−11	9.71E−11	1.08E−09	3.58E−09	1.47E−08	1.32E−07
F10	5.12E+01	4.49E+01	4.49E+01	**4.09E+01**	4.62E+01	4.70E+01	5.53E+01	5.50E+01	6.01E+01	7.57E+01	1.54E+02
F11	3.23E+01	2.08E+01	1.73E+01	1.57E+01	1.53E+01	1.59E+01	**1.50E+01**	1.51E+01	1.55E+01	1.69E+01	1.77E+01
F12	**1.66E+00**	1.15E+04	1.16E+04	9.67E+03	7.33E+03	7.60E+03	8.37E+03	7.15E+03	9.86E+03	8.23E+03	1.01E+04
F13	1.26E+00	1.24E+00	1.15E+00	1.19E+00	**1.14E+00**	**1.14E+00**	1.19E+00	1.18E+00	1.27E+00	1.24E+00	1.30E+00
F14	1.32E+01	1.28E+01	1.27E+01	1.26E+01	1.27E+01	1.26E+01	1.27E+01	1.26E+01	**1.24E+01**	1.26E+01	1.27E+01

The mean error is recorded based on 30 independent runs, and the best results are highlighted in boldface.

Table A2
Mean error of CMM-rcBBOg with different P_e on the 37 functions.

P_e	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f01	5.26E−04	3.53E−11	4.05E−23	8.83E−19	1.85E−16	4.81E−15	6.20E−14	2.49E−13	1.06E−12	2.41E−12	8.60E−12
f02	5.14E−02	8.29E−08	4.75E−13	3.15E−10	1.08E−08	7.98E−08	2.84E−07	7.89E−07	1.55E−06	3.12E−06	5.49E−06
f03	2.32E+01	7.74E+00	3.70E+00	2.38E+00	1.48E+00	1.16E+00	6.67E−01	3.40E−01	2.03E−01	1.83E−01	**1.25E−01**
f04	6.49E−02	2.43E−02	1.40E−02	1.23E−02	**9.74E−03**	1.12E−02	1.17E−02	1.41E−02	1.90E−02	1.87E−02	3.35E−02
f05	9.31E+01	1.04E+02	5.23E+01	4.40E+01	3.98E+01	3.44E+01	**2.99E+01**	3.07E+01	3.27E+01	3.19E+01	4.34E+01
f06	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f07	4.87E−03	**1.06E−03**	1.35E−03	1.75E−03	1.73E−03	2.08E−03	2.36E−03	2.42E−03	2.53E−03	2.90E−03	2.97E−03
f08	4.76E+02	**4.72E+02**	6.30E+02	1.16E+03	2.26E+03	3.14E+03	4.19E+03	4.46E+03	4.80E+03	4.99E+03	4.82E+03
f09	1.44E−02	4.09E−15	**0.00E+00**	**0.00E+00**	2.72E−15	1.22E−13	1.72E−12	1.18E−11	4.97E−11	1.85E−10	4.71E−10
f10	1.53E−02	2.27E−06	**2.20E−12**	4.28E−10	7.74E−09	4.35E−08	1.45E−07	3.34E−07	6.79E−07	1.13E−06	2.14E−06
f11	2.99E−01	6.06E−03	1.89E−03	**5.75E−04**	2.05E−03	2.05E−03	1.88E−03	7.87E−03	1.90E−02	2.35E−02	9.28E−03
f12	6.26E−01	5.19E−02	9.17E−25	3.46E−03	6.91E−03	**4.03E−17**	1.73E−02	6.91E−03	1.38E−02	1.04E−02	7.95E−02
f13	1.18E−04	1.12E−10	9.35E−23	**1.59E−19**	4.99E−17	1.14E−15	1.38E−14	6.70E−14	2.66E−13	1.27E−12	3.56E−12
f14	3.02E+00	2.44E+00	1.43E+00	1.12E+00	1.07E+00	1.59E+00	1.39E+00	**1.04E+00**	1.49E+00	1.38E+00	2.99E+00
f15	4.11E−03	1.21E−03	7.82E−04	7.96E−04	**5.21E−04**	6.88E−04	5.85E−04	5.64E−04	5.43E−04	6.04E−04	7.03E−04
f16	1.13E−02	2.73E−03	1.28E−03	9.06E−04	7.60E−04	1.84E−04	1.31E−04	2.88E−05	**1.82E−05**	3.54E−05	1.17E−04
f17	8.76E−03	3.56E−03	2.58E−03	1.39E−03	**5.75E−04**	1.08E−03	1.62E−03	1.71E−03	1.15E−03	9.94E−04	1.67E−03
f18	1.34E+00	1.72E−01	4.74E−02	1.47E−02	4.05E−02	2.13E−02	2.83E−01	1.64E−03	**4.27E−06**	1.15E−05	3.16E−02
f19	1.21E−02	1.28E−03	1.99E−04	8.37E−05	**6.88E−05**	1.17E−04	1.08E−04	1.35E−04	1.07E−04	5.68E−04	8.30E−05
f20	5.02E−02	2.83E−02	3.57E−02	2.38E−02	2.78E−02	1.98E−02	3.57E−02	1.98E−02	1.98E−02	2.38E−02	**1.60E−02**
f21	3.91E+00	3.20E+00	4.33E+00	2.51E+00	3.08E+00	1.83E+00	1.58E+00	**4.41E−01**	1.08E+00	7.49E−01	5.76E−01
f22	3.07E+00	2.93E+00	1.46E+00	1.91E+00	2.07E+00	1.32E+00	1.81E+00	6.21E−01	4.78E−01	4.64E−01	**2.84E−01**
f23	3.51E+00	1.21E+00	1.39E+00	1.63E+00	4.79E−01	9.73E−01	5.26E−01	4.79E−01	2.84E−01	2.55E−01	**1.80E−01**
F01	8.94E−05	**5.49E−22**	4.24E−26	2.53E−20	1.57E−17	4.73E−16	5.78E−15	3.86E−14	1.65E−13	6.04E−13	6.26E−12
F02	1.88E+02	7.78E+01	4.91E+01	3.74E+01	3.71E+01	3.67E+01	**3.26E+01**	3.69E+01	4.47E+01	8.61E+01	5.35E+02
F03	3.44E+06	2.56E+06	**2.31E+00**	1.90E+06	1.70E+06	1.71E+06	1.60E+06	1.62E+06	1.98E+06	1.93E+06	3.26E+06
F04	2.02E+04	7.00E+03	7.76E+03	5.38E+03	5.50E+03	**5.04E+03**	5.89E+03	5.41E+03	5.54E+03	6.41E+03	1.16E+04
F05	6.42E+03	4.37E+03	4.49E+03	4.61E+03	4.77E+03	5.02E+03	5.70E+03	6.58E+03	7.50E+03	9.75E+03	1.52E+04
F06	4.11E+03	1.30E+03	4.47E+02	4.59E+02	6.27E+02	**2.82E+02**	1.03E+03	5.99E+02	6.30E+02	1.03E+03	8.43E+02
F07	2.24E+03	5.98E+02	3.04E+02	2.19E+02	**2.18E+02**	2.50E+02	3.01E+02	4.23E+02	7.28E+02	1.52E+03	6.60E+03
F08	2.07E+01	**2.06E+01**	**2.06E+01**	**2.06E+01**	**2.06E+01**	**2.06E+01**	2.07E+01	**2.06E+01**	2.07E+01	**2.06E+01**	2.07E+01
F09	1.76E−02	1.15E−14	**0.00E+00**	**0.00E+00**	3.02E−15	1.08E−13	1.91E−12	1.27E−11	5.37E−11	1.65E−10	5.42E−10
F10	5.92E+01	4.85E+01	4.69E+01	4.69E+01	**4.52E+01**	5.05E+01	4.99E+01	6.71E+01	6.85E+01	9.32E+01	1.83E+02
F11	3.13E+01	1.93E+01	1.68E+01	1.57E+01	1.53E+01	1.51E+01	1.52E+01	**1.46E+01**	1.49E+01	1.64E+01	1.89E+01
F12	1.95E+04	1.57E+04	1.08E+04	1.29E+04	1.09E+04	**1.03E+00**	7.94E+03	9.72E+03	1.12E+04	1.28E+04	1.42E+04
F13	1.24E+00	**1.06E+00**	1.10E+00	1.06E+00	1.07E+00	1.11E+00	1.10E+00	1.18E+00	1.07E+00	1.21E+00	1.31E+00
F14	1.36E+01	1.32E+01	1.31E+01	1.32E+01	1.32E+01	**1.30E+01**	1.31E+01	**1.30E+01**	**1.30E+01**	**1.30E+01**	1.31E+01

The mean error is recorded based on 30 independent runs, and the best results are highlighted in boldface.

Table A3
Mean errors of CMM-pBBO with different P_e on the 37 functions.

P_e	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f01	7.74E−08	1.34E−24	3.95E−16	2.31E−13	8.04E−12	6.31E−11	2.84E−10	8.87E−10	1.93E−09	3.94E−09	8.19E−09
f02	3.32E−05	3.33E−14	7.64E−09	3.29E−07	2.30E−06	7.98E−06	2.06E−05	4.14E−05	6.95E−05	1.08E−04	1.49E−04
f03	2.15E−01	2.87E−03	3.00E−04	1.63E−05	7.67E−07	**2.29E−07**	3.03E−07	5.80E−07	8.63E−07	1.51E−06	2.00E−06
f04	7.87E−03	1.32E−06	4.45E−08	**6.53E−07**	2.88E−06	7.50E−06	1.36E−05	2.40E−05	3.59E−05	4.91E−05	6.54E−05
f05	4.55E+01	3.94E+01	4.83E+01	3.65E+01	3.77E+01	2.50E+01	2.87E+01	2.95E+01	2.25E+01	2.57E+01	3.21E+01
f06	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f07	2.04E−03	5.13E−03	6.54E−03	7.48E−03	7.92E−03	9.15E−03	9.40E−03	8.86E−03	8.89E−03	8.30E−03	6.89E−03
f08	**3.63E+02**	4.48E+02	6.19E+02	8.47E+02	1.45E+03	1.97E+03	2.62E+03	3.11E+03	3.62E+03	4.27E+03	4.88E+03
f09	4.30E−07	**0.00E+00**	4.74E−16	7.42E−13	3.08E−11	4.93E−10	3.05E−09	1.15E−08	2.87E−08	6.92E−08	1.28E−07
f10	3.32E−05	**5.90E−13**	1.21E−08	3.27E−07	1.90E−06	5.45E−06	1.14E−05	1.95E−05	2.84E−05	4.04E−05	6.73E−05
f11	1.45E−02	1.56E−03	6.57E−04	**2.60E−15**	4.93E−04	3.29E−04	1.40E−03	3.37E−03	5.50E−03	2.35E−02	9.60E−03
f12	4.49E−02	1.04E−02	2.07E−02	**1.81E−15**	1.04E−02	3.46E−03	1.73E−02	3.46E−03	2.07E−02	3.11E−02	3.80E−02
f13	8.01E−09	1.95E−25	**4.99E−17**	4.47E−14	1.17E−12	1.14E−11	6.95E−11	2.07E−10	5.22E−10	1.20E−09	2.60E−09
f14	5.93E−01	9.21E−01	1.48E+00	7.02E−01	8.77E−01	**4.61E−01**	8.00E−01	1.22E+00	8.34E−01	1.50E+00	2.27E+00
f15	1.35E−03	1.15E−03	4.33E−04	3.94E−04	1.07E−03	3.52E−04	4.45E−04	3.64E−04	3.64E−04	2.87E−04	**2.82E−04**
f16	2.19E−04	2.38E−15	1.54E−14	2.35E−15	2.26E−15	2.26E−15	7.46E−09	**2.23E−15**	**2.22E−15**	2.23E−15	6.37E−08
f17	2.10E−05	1.19E−05	3.95E−06	6.14E−06	4.07E−06	3.22E−06	3.25E−07	1.86E−07	**1.45E−09**	7.69E−08	5.56E−06
f18	7.50E−03	7.49E−06	3.14E−15	2.53E−09	7.74E−14	1.91E−15	1.44E−15	2.83E−15	**1.04E−15**	1.38E−15	3.32E−10
f19	1.74E−06	1.62E−06	3.17E−07	2.75E−07	2.72E−07	2.55E−07	2.56E−07	7.83E−07	**2.54E−07**	2.55E−07	2.54E−05
f20	4.76E−02	3.57E−02	3.17E−02	2.77E−02	2.38E−02	2.77E−02	2.77E−02	**1.19E−02**	2.77E−02	1.98E−02	1.59E−02
f21	5.09E+00	3.83E+00	2.84E+00	2.51E+00	1.42E+00	2.75E+00	1.42E+00	1.33E+00	1.34E+00	7.58E−01	**4.19E−01**
f22	3.34E+00	2.77E+00	1.31E+00	2.07E+00	7.50E−01	4.78E−01	4.78E−01	4.45E−01	**−3.18E−11**[a]	4.77E−01	4.77E−01
f23	3.16E+00	1.92E+00	2.12E+00	1.44E+00	1.00E+00	4.04E−01	1.33E+00	8.38E−01	4.02E−01	**−9.82E−06**	2.23E−01
F01	6.98E−10	**2.82E−27**	9.06E−18	6.73E−15	4.22E−13	4.21E−12	2.23E−11	7.87E−11	1.99E−10	4.36E−10	2.88E−09
F02	4.18E+00	4.07E−01	1.55E−01	6.93E−02	3.25E−02	1.79E−02	7.89E−03	**3.67E−03**	7.47E−03	1.73E−02	3.73E+01
F03	2.09E+06	1.61E+06	1.24E+06	1.15E+06	1.18E+06	1.03E+06	**9.92E+05**	1.09E+06	1.09E+06	1.16E+06	1.35E+06
F04	9.48E+02	7.37E+01	1.35E+01	4.19E+00	7.84E−01	3.11E−01	9.07E−02	6.08E−02	**5.00E−02**	5.76E−02	2.17E+02

Table A3 (*Continued*)

F05	4.63E+03	**3.65E+03**	3.82E+03	3.74E+03	3.75E+03	4.07E+03	4.15E+03	4.70E+03	5.49E+03	6.41E+03	1.01E+04
F06	6.82E+02	6.72E+02	3.04E+02	3.06E+02	2.30E+02	3.30E+02	3.65E+02	**1.63E+02**	5.52E+02	4.18E+02	8.26E+02
F07	1.46E−02	1.28E−02	**1.17E−02**	1.47E−02	1.20E−02	1.30E−02	1.26E−02	2.23E−02	2.76E−02	1.50E+00	5.15E+03
F08	2.03E+01	**2.02E+01**	2.06E+01	2.07E+01	2.07E+01	2.06E+01	2.07E+01	2.07E+01	2.06E+01	2.06E+01	2.06E+01
F09	8.20E−07	**0.00E+00**	2.37E−16	6.12E−13	2.74E−11	3.55E−10	2.77E−09	1.09E−08	3.43E−08	8.39E−08	1.85E−07
F10	5.74E+01	5.52E+01	5.14E+01	4.82E+01	**4.59E+01**	5.35E+01	5.28E+01	5.17E+01	5.93E+01	7.51E+01	1.39E+02
F11	2.86E+01	1.80E+01	1.86E+01	1.60E+01	1.36E+01	1.31E+01	1.34E+01	1.30E+01	**1.23E+01**	1.39E+01	1.54E+01
F12	1.13E+04	7.65E+03	9.20E+03	6.16E+03	5.04E+03	6.06E+03	**4.99E+03**	5.39E+03	6.16E+03	1.15E+04	8.54E+03
F13	**1.08E+00**	**1.08E+00**	1.10E+00	1.13E+00	1.11E+00	1.10E+00	1.08E+00	1.13E+00	1.25E+00	1.45E+00	1.43E+00
F14	1.35E+01	1.32E+01	1.31E+01	1.32E+01	1.31E+01	1.31E+01	**1.30E+01**	**1.30E+01**	**1.30E+01**	**1.30E+01**	1.31E+01

The mean error is recorded based on 30 independent runs, and the best results are highlighted in boldface.
[a] A negative value means that the achieved value is better than the optimal value provided in Table 1.

Table A4
Mean errors of CMM-DE/BBO with different P_e on the 37 functions.

P_e	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f01	9.92E−21	2.54E−21	4.14E−22	5.06E−23	4.74E−24	2.90E−25	9.43E−27	1.21E−28	8.48E−31	1.18E−32	5.57E+01
f02	2.15E−18	7.80E−18	2.31E−17	4.78E−17	7.03E−17	7.82E−17	4.10E−17	1.17E−17	1.85E−18	8.89E−20	2.13E+00
f03	6.32E+02	7.09E−10	7.13E−18	1.03E−22	**1.09E−24**	1.53E−23	1.32E−20	6.73E−16	1.42E−09	2.00E−06	5.47E+01
f04	1.71E−07	8.25E−09	3.11E−10	8.83E−12	2.03E−13	2.14E−15	**1.68E−17**	4.60E−13	7.45E−06	1.61E−02	2.11E+00
f05	1.80E+01	1.62E+01	1.21E+01	2.46E+00	6.75E−01	**2.14E−01**	1.35E+00	6.81E+00	1.86E+01	2.36E+01	5.45E+02
f06	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.63E+01
f07	6.05E−03	5.29E−03	4.28E−03	3.48E−03	3.23E−03	2.66E−03	2.29E−03	2.20E−03	1.87E−03	**1.71E−03**	2.50E−03
f08	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	**1.34E−02**	3.89E+03
f09	**0.00E+00**	**0.00E+00**	**0.00E+00**	5.39E−15	2.03E−09	2.27E−01	4.65E+00	1.05E+01	1.75E+01	3.00E+01	4.66E+01
f10	2.26E−11	1.19E−11	5.85E−12	2.24E−12	7.54E−13	2.17E−13	4.00E−14	1.01E−14	6.16E−15	**5.57E−15**	1.98E+00
f11	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.47E−04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.43E+00
f12	2.60E−21	6.57E−22	1.70E−22	2.35E−23	3.77E−24	2.95E−25	8.45E−27	1.04E−28	2.59E−30	9.04E−31	1.84E−01
f13	**1.43E−30**	3.59E−21	6.21E−22	9.21E−23	7.47E−24	4.99E−25	2.04E−26	4.86E−28	2.22E−29	1.04E−29	1.48E+00
f14	**5.97E−08**	**5.97E−08**	**5.97E−08**	**5.97E−08**	1.11E−02	**5.97E−08**	5.98E−08	1.18E−07	4.14E−04	1.24E−02	1.18E−01
f15	1.27E−05	−9.20E−10	**−1.40E−08**[a]	**−1.40E−08**	−1.23E−08	**−1.40E−08**	2.39E−06	**−1.40E−08**	9.82E−07	6.70E−06	1.45E−05
f16	5.16E−11	9.94E−12	5.53E−12	6.21E−12	5.54E−12	1.64E−12	2.26E−12	2.31E−13	2.81E−13	6.67E−13	**1.32E−13**
f17	9.95E−16	2.85E−16	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**	**1.67E−16**
f18	2.30E−14	1.40E−14	1.35E−14	1.26E−14	9.56E−15	7.18E−15	6.50E−15	5.88E−15	3.57E−15	**2.98E−15**	3.46E−15
f19	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**	**2.54E−07**
f20	1.59E−02	3.96E−03	3.96E−03	7.73E−13	1.46E−13	5.03E−14	**3.30E−14**	3.96E−03	3.80E−14	3.96E−03	3.13E−13
f21	6.69E−01	1.74E−01	3.27E−05	7.01E−06	1.27E−06	4.96E−07	3.39E−07	1.56E−07	7.87E−08	3.97E−08	**1.65E−08**
f22	3.03E−04	1.36E−05	1.07E−05	4.79E−07	2.73E−07	1.32E−07	8.55E−08	3.61E−08	3.43E−08	**1.31E−08**	2.11E−08
f23	1.89E−05	8.78E−06	−8.50E−06	−9.37E−06	−9.64E−06	−9.68E−06	−9.30E−06	−9.74E−06	−9.75E−06	**−9.80E−06**	**−9.80E−06**
F01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	9.14E−29	7.51E+03
F02	7.30E+02	3.87E−04	1.13E−08	1.06E−11	1.84E−12	**3.19E−12**	1.58E−10	1.22E−08	1.37E−05	4.63E−02	4.68E+03
F03	1.76E+07	1.56E+05	**1.53E+05**	2.14E+05	2.55E+05	3.06E+05	4.19E+05	4.44E+05	4.45E+05	6.48E+05	3.61E+06
F04	2.37E+03	3.37E−01	6.39E−04	**4.61E−05**	6.58E−05	1.02E−04	3.71E−04	1.84E−03	1.24E−01	5.71E+00	3.57E+03
F05	5.01E+02	2.64E+01	2.43E+00	**6.64E−01**	7.68E+00	1.15E+02	1.44E+02	2.16E+02	2.99E+02	9.36E+02	7.55E+03
F06	2.37E+01	2.08E+01	1.82E+01	1.37E+01	9.18E+00	**8.80E+00**	1.05E+01	1.80E+01	2.30E+01	3.56E+01	2.11E+08
F07	6.57E−04	5.22E−11	1.48E−17	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.47E−04	1.73E−03	1.05E−02	5.76E+03
F08	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**	**2.09E+01**
F09	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.42E−15	5.71E−06	1.76E+00	7.74E+00	1.51E+01	2.74E+01	6.79E+01
F10	8.09E+01	7.63E+01	7.66E+01	7.11E+01	7.26E+01	6.95E+01	6.89E+01	6.42E+01	**6.09E+01**	6.24E+01	9.70E+01
F11	3.07E+01	2.97E+01	2.98E+01	2.97E+01	2.95E+01	2.93E+01	2.84E+01	2.91E+01	**2.86E+01**	2.87E+01	2.88E+01
F12	2.47E+04	2.58E+04	2.11E+04	2.26E+04	8.80E+03	5.41E+03	1.92E+03	**1.68E+03**	2.31E+03	2.33E+03	1.48E+04
F13	**2.58E+00**	2.70E+00	2.83E+00	2.98E+00	3.19E+00	3.37E+00	3.52E+00	3.87E+00	4.58E+00	5.55E+00	7.32E+00
F14	1.29E+01	1.29E+01	1.29E+01	1.29E+01	1.28E+01	1.28E+01	1.27E+01	1.27E+01	1.27E+01	1.26E+01	**1.25E+01**

The mean error is recorded based on 30 independent runs, and the best results are highlighted in boldface.
[a] A negative value means that the achieved value is better than the optimal value provided in Table 1.

Table A5
Average ranking of CMM-BBO algorithms with different P_e according to the Friedman test.

	P_e	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
CMM-rcBBO	Average ranking	10.26	7.69	6.20	5.43	4.93	4.80	4.96	**4.16**	5.11	5.46	7.00
	Final rank	11	10	8	6	3	2	4	**1**	5	7	9
CMM-rcBBOg	Average ranking	10.23	7.27	5.58	4.95	4.50	**4.49**	5.34	4.70	5.46	6.08	7.41
	Final rank	11	9	7	4	2	**1**	5	3	6	8	10
CMM-pBBO	Average ranking	8.92	6.14	5.61	5.34	4.81	**4.46**	5.11	5.12	5.66	6.62	8.22
	Final rank	11	8	6	5	2	**1**	3	4	7	9	10
CMM-DE/BBO	Average ranking	7.54	6.15	5.72	5.19	5.22	**4.96**	5.01	5.34	5.82	6.45	8.61
	Final rank	10	8	6	3	4	**1**	2	5	7	9	11
Sum	Average ranking	36.95	27.24	23.11	20.91	19.46	**18.70**	20.42	19.32	22.05	24.61	31.23
	Final rank	11	9	7	5	3	**1**	4	2	6	8	10

References

[1] T. Park, K.R. Ryu, A dual-population genetic algorithm for adaptive diversity control, IEEE Trans. Evol. Comput. 14 (2010) 865–884.
[2] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2001) 159–195.
[3] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (2006) 646–657.
[4] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.
[5] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (2009) 945–958.
[6] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (2006) 281–295.
[7] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, IEEE (2005) 124–129.
[8] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (2008) 702–713.
[9] Y.-J. Zheng, H.-F. Ling, H.-H. Shi, H.-S. Chen, S.-Y. Chen, Emergency railway wagon scheduling by hybrid biogeography-based optimization, Comput. Oper. Res. 43 (2014) 1–8.
[10] Y.-J. Zheng, H.-F. Ling, S.-Y. Chen, J.-Y. Xue, A hybrid neuro-fuzzy network based on differential biogeography-based optimization for online population classification in earthquakes, IEEE Trans. Fuzzy Syst. 23 (2015) 1070–1083.
[11] M. Tamjidy, S. Paslar, B.H.T. Baharudin, T.S. Hong, M. Ariffin, Biogeography based optimization (BBO) algorithm to minimise non-productive time during hole-making process, Int. J. Prod. Res. 53 (2015) 1880–1894.
[12] Y.-J. Zheng, H.-F. Ling, J.-Y. Xue, Ecogeography-based optimization: enhancing biogeography-based optimization with ecogeographic barriers and differentiations, Comput. Oper. Res. 50 (2014) 115–127.
[13] W. Gong, Z. Cai, C.X. Ling, H. Li, A real-coded biogeography-based optimization with mutation, Appl. Math. Comput. 216 (2010) 2749–2758.
[14] X. Li, M. Yin, Multi-operator based biogeography based optimization with mutation for global numerical optimization, Comput. Math. Appl. 64 (2012) 2833–2844.
[15] G. Xiong, D. Shi, X. Duan, Enhancing the performance of biogeography-based optimization using polyphyletic migration operator and orthogonal learning, Comput. Oper. Res. 41 (2014) 125–139.
[16] X. Li, J. Wang, J. Zhou, M. Yin, A perturb biogeography based optimization with mutation for global numerical optimization, Appl. Math. Comput. 218 (2011) 598–609.
[17] H. Ma, D. Simon, Blended biogeography-based optimization for constrained optimization, Eng. Appl. Artif. Intell. 24 (2011) 517–525.
[18] D. Simon, M.G. Omran, M. Clerc, Linearized biogeography-based optimization with re-initialization and local search, Inf. Sci. 267 (2014) 140–157.
[19] D. Du, D. Simon, M. Ergezer, Biogeography-based optimization combined with evolutionary strategy and immigration refusal, Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on, IEEE (2009) 997–1002.
[20] W. Gong, Z. Cai, C.X. Ling, DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization, Soft Comput. 15 (2010) 645–665.
[21] I. Boussaïd, A. Chatterjee, P. Siarry, M. Ahmed-Nacer, Two-stage update biogeography-based optimization using differential evolution algorithm (DBBO), Comput. Oper. Res. 38 (2011) 1188–1198.
[22] H. Kundra, M. Sood, Cross-country path finding using hybrid approach of PSO and BBO, Int. J. Comput. Appl. 7 (2010) 15–19.
[23] P. Savsani, R. Jhala, V. Savsani, Effect of hybridizing biogeography-based optimization (BBO) technique with artificial immune algorithm (AIA) and ant colony optimization (ACO), Appl. Soft Comput. 21 (2014) 542–553.
[24] Y.-J. Zheng, H.-F. Ling, X.-B. Wu, J.-Y. Xue, Localized biogeography-based optimization, Soft Comput. 18 (2014) 2323–2334.
[25] X.-w. Zheng, D.-j. Lu, X.-g. Wang, H. Liu, A cooperative coevolutionary biogeography-based optimizer, Appl. Intell. 43 (2015) 95–111.
[26] H. Ma, M. Fei, Z. Ding, J. Jin, Biogeography-based optimization with ensemble of migration models for global numerical optimization, Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE (2012) 1–8.
[27] M. Ergezer, D. Simon, D. Du, Oppositional biogeography-based optimization, Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on, IEEE (2009) 1009–1014.
[28] S. Saremi, S. Mirjalili, A. Lewis, Biogeography-based optimisation with chaos, Neural Comput. Appl. 25 (2014) 1077–1097.
[29] Y. Wang, H.-X. Li, T. Huang, L. Li, Differential evolution based on covariance matrix learning and bimodal distribution parameter setting, Appl. Soft Comput. 18 (2014) 232–247.
[30] S.-M. Guo, C.-C. Yang, Enhancing differential evolution utilizing eigenvector-based crossover operator, IEEE Trans. Evol. Comput. 19 (2015) 31–49.
[31] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (1999) 82–102.
[32] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, KanGAL report, 2005005 (2005).
[33] Q. Feng, S. Liu, J. Zhang, G. Yang, L. Yong, Biogeography-based optimization with improved migration operator and self-adaptive clear duplicate operator, Appl. Intell. 41 (2014) 563–581.
[34] J. Alcala-Fdez, L. Sanchez, S. Garcia, M.J. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Comput. 13 (2009) 307–318.