Applied Soft Computing Journal 74 (2019) 385–410

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Mixed grey wolf optimizer for the joint denoising and unmixing of multispectral images

Benoit Martin [a], Julien Marot [b,*], Salah Bourennane [b]

[a] IntuiSense Technologies, 13420 Gemenos, France
[b] Aix Marseille Univ, CNRS, Ecole Centrale Marseille, Institut Fresnel, Domaine Universitaire de Saint Jerome, Av. Escadrille Normandie Niemen, 13397, Marseille, France

HIGHLIGHTS

- A novel swarm intelligence algorithm is proposed: a mixed version of the grey wolf optimizer.
- The proposed method outperforms other versions of grey wolf optimizer on benchmark functions.
- For the first time, an image processing issue is faced with a mixed grey wolf optimizer.
- Rank values for denoising, and coefficients for unmixing, are jointly estimated with our method.
- A comparative study is made with particle swarm optimization and modified grey wolf optimizer.

ARTICLE INFO

Article history:
Received 9 January 2018
Received in revised form 4 October 2018
Accepted 11 October 2018
Available online 24 October 2018

Keywords:
Bio-inspired optimization
Grey wolf optimizer
Multispectral image
Denoising
Unmixing

ABSTRACT

Grey wolf optimizer (GWO) is a bio-inspired iterative optimization algorithm which simulates the hunting behaviour of a pack of wolves. Their position is updated across iterations in a search space under the leadership of the α, β, and δ wolves. In this work, a novel mixed GWO is proposed, which, for the first time, searches for parameter values in both continuous and discrete spaces. In the proposed approach, the update rules are as follows: the leaders guide the hunt, assisted by two random wolves during the first half of the iterations; for the continuous parameters, a weighted combination of the leaders' contribution is calculated; for the discrete parameters, a random selection is performed instead, and the probability for the α to be selected increases at the detriment of the other leaders across iterations. The exploration and exploitation phases are distinguished for continuous and discrete parameters. The proposed mixed GWO is compared against other bio-inspired optimization methods, using several test problems which are either continuous, discrete or mixed: the proposed algorithm can significantly improve the performance metrics. Moreover, our method is adapted to simultaneously denoise and unmix real-world multispectral images.

1. Introduction

An optimization problem consists of maximizing or minimizing a real fitness function by systematically choosing input values from within an allowed set and computing the value of the function. To solve an optimization problem, an interesting field of applied mathematics has attracted much interest during the past few years: Meta-heuristics. These methods exhibit a good capacity in solving computationally expensive numerical problems, with a limited number of function evaluations [1]. Meta-heuristics may be classified into three main classes: physics-based, evolutionary, and swarm intelligence (SI) algorithms. Physics-based algorithms include for instance gravitational search algorithm [2–4], where the search agents are provided with a mass which depends on their fitness, and wind driven optimization (WDO) [5]. Simulated annealing [6] can also be seen as a physics-based algorithm. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored. Evolutionary algorithms [7], together with swarm intelligence algorithms [8–11], compose the bio-inspired optimization algorithms. They differ for instance in their way to encode the agents which search the space composed by the allowed sets of values. The convergence success of a bio-inspired optimization algorithm depends on directing and balancing its so-called 'exploration' and 'exploitation' abilities [12,13]. Among bio-inspired swarm intelligence algorithms,

* Corresponding author.
E-mail address: julien.marot@fresnel.fr (J. Marot).

https://doi.org/10.1016/j.asoc.2018.10.019
1568-4946/

Nomenclature

Notations used

a or A	Scalars
$\mathbf{a}$	Vectors
$\mathbf{A}$	Matrices
$\mathbb{A}$	Manifolds

Parameters used

N	Number of expected parameters
i	Index of the expected parameter with $i = 1, \ldots, N$
K_i	ith expected parameter
$f(\cdot)$	Function to minimize, also called criterion
a	Phase distinguisher
$iter$	Current iteration
T_{max}	Maximum number of iterations
η	Phase emphasizer
Q	Number of wolves in the Herd
$\mathbf{x}_q^{\mathbf{k}}(\text{iter})$	Position of the qth wolf, with $q = 1, \ldots, Q$
$\mathbf{x}_\alpha^{\mathbf{k}}, \mathbf{x}_\beta^{\mathbf{k}}, \mathbf{x}_\delta^{\mathbf{k}}$	Positions of the leaders α, β and δ
$\mathbf{x}_{\rho 1}^{\mathbf{k}}, \mathbf{x}_{\rho 2}^{\mathbf{k}}$	Positions of the random wolves $\rho 1$ and $\rho 2$
$\hat{K}_i$	ith estimated parameter

Parameters solely used for Improved Discrete GWO

H_i	Number of candidate value for the parameter K_i
$\mathbf{d}_i^{val} = \left[K_i^1, \ldots, K_i^{h_i}, \ldots, K_i^{H_i}\right]^T$	Candidates values for the parameter K_i
$\mathbf{d}_i^{ind} = [1, \ldots, h_i, \ldots, H_i]^T$	Indexes of each candidate value for the parameter K_i
$\mathbf{h}_q(\text{iter})$	Vector of indexes associated with the wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$
$\mathbf{h}_\alpha, \mathbf{h}_\beta, \mathbf{h}_\delta$	Vector of indexes associated with the leaders α, β and δ
l	Refer to the selected leader, either α, β, δ, $\rho 1$ or $\rho 2$
Δ	Displacement factor

Parameters solely used for Global Continuous GWO

y_i^l	Contribution of a leader l for the parameter K_i

the grey wolf optimizer (GWO) [10,11,14,15] has recently attracted much attention due to its good performances, simplicity of use, and good capacity to distinguish between an exploration phase and an exploitation phase. That is why, further in this work, we present a detailed state-of-the-art about bio-inspired optimization in general and GWO in particular.

The main purpose of this paper is to develop an automatic algorithm which estimates the best parameter values in a mixed search space, with an adequate variant of the grey wolf optimizer. For this, we propose the mixed grey wolf optimization algorithm (mixedGWO). This novel version of GWO handles the simultaneous estimation of optimal parameters in search spaces which can be continuous as well as discrete.

This new research has been driven by an image processing application: the simultaneous denoising and unmixing of multispectral images. Denoising of multispectral images is an issue which can be tackled by the tensor signal processing paradigm: tensor signal denoising methods have been proposed which involve so-called subspace ranks, which are integer-valued. Unmixing of multispectral images consists in estimating mixing coefficients, which are positive, real-valued, and less than or equal to 1. Therefore, to solve the issue of simultaneous denoising and unmixing of multispectral images, a mixed optimization method is required.

Our main contributions are listed as follows:

- for the first time, we propose a mixed grey wolf optimizer, which allows searching parameter values in both discrete and continuous spaces;
- update rules are proposed for wolves which permit to distinguish clearly between an exploration phase, and an exploitation phase, respecting thereby the primal philosophy of the grey wolf optimizer, and taking advantage of its properties;
- to the best of authors' knowledge, the identification of the best parameters for tensor signal processing with a mixed-integer optimization problem has not been reported yet, in the previous research.

The remainder of this paper is organized as follows.

In Section 2, a state-of-the-art about bio-inspired optimization methods, in particular swarm intelligence, and discrete algorithms, is provided. We present their applications in the image processing field. Most of the discrete swarm intelligence methods are restricted to binary versions of the algorithms, and a comparative study permits to justify the choice of GWO as a basis to create a mixed bio-inspired optimization algorithm. In Section 3, the proposed algorithms mixedGWO and its adaptive version amixedGWO and their computational aspects are described. In Section 4, a performance evaluation of our proposed methods is done compared to the state-of-the-art on several benchmark functions. Finally, in Section 5, the usefulness of the amixedGWO algorithm on a real-world application such as the denoising and unmixing of multispectral images is studied.

The notations and the parameters used throughout this paper are gathered in the following nomenclature along with their description.

2. Background and state-of-the-art

This section provides an overview of bio-inspired optimization methods and focuses on their image processing applications.

In Section 2.1, a background about bio-inspired optimization methods is proposed to the reader. These methods distinguish for instance on the properties of the search agents they encode. We focus on swarm intelligence (SI) algorithms, which includes the most recent publications about bio-inspired optimization methods. In Section 2.2, we provide a short state-of-the-art about discrete bio-inspired optimization methods, which are essentially binary optimization methods. In Section 2.3, we summarize the image processing applications which have been tackled with bio-inspired optimization methods, and we focus on the applications of discrete bio-inspired methods.

2.1. Bio-inspired optimization

Among evolutionary algorithms, the most popular are the genetic algorithms (GA). This algorithm was proposed by Holland in 1992 [7] and simulated Darwinian evolution concepts. An often cited, now well-known reference [16] introduces genetic algorithms in the context of evolutionary computation which implies the evolution of a population which is inspired by Darwin's natural selection theory. Another largely cited reference presents basics about genetics, the hierarchical genetic algorithm, and applications to H∞ control, neural network, and speech recognition [17]. The seminal work about swarm intelligence is particle swarm optimization (PSO) [8,9], proposed by Kennedy and Eberhart, who got inspired by [18] where the term 'particle swarm' was chosen to define the members of a population or test set. In the PSO paradigm, the population members are mass-less and volumeless. Their evolution is described through position, speed, and acceleration parameters. The concept of swarm got first inspired by the behaviour of birds, where one leader guides the flock. In [10], a method involving three leaders was proposed, which gets inspired by the behaviour of wolves, namely grey wolf optimization: the GWO algorithm mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. Four types of grey wolves such as alpha, beta, delta, and omega are employed to simulate the leadership hierarchy. In addition, the three main steps of hunting, searching for prey, encircling prey, and attacking prey, are implemented.

The computational rules of PSO and GWO are described in Algorithms 5 and 6, and available in the Appendix .

A number of variants are also proposed to improve the performance of vanilla GWO that include a hybrid version of GWO with PSO [19], and a 'modified GWO' (mGWO) [11] which includes a slightly modified update rule for the wolves. In [14] grey wolf optimization is adapted for multi-objective optimization problems. In [15] a 'chaotic' GWO is proposed which aims at accelerating the global convergence speed of GWO. Compared to PSO and other prevailing techniques, GWO has the ability to converge to a better quality near-optimal solution and possesses better convergence characteristics [19]. Also, GWO has a good balance between exploration and exploitation that results in high local optima avoidance [20]. It has been successfully applied, as mentioned in [11], for solving economic dispatch problems, optimal design of double layer grids, time forecasting, flow shop scheduling problem, optimal power flow problem, and optimizing key values in the cryptography algorithms. Some other variants of the GWO algorithm can be found in [13,21,22] and in the references inside.

Within bio-inspired optimization algorithms, Artificial Bee Colony (ABC) [23] and Tree Seed Algorithm (TSA) [24] exhibit a common property: they rule the displacement of the search agents through the choice of a random leader, selected among the whole population of search agents.

In the ABC algorithm [23], the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. A bee waiting on the dance area for making decision to choose a food source is called an onlooker and a bee going to the food source visited by itself previously is named an employed bee. A bee carrying out random search is called a scout. In the original ABC algorithm, one half of the bees are employed bees, and the other half constitutes the onlookers. In order to update the position of the current employed bee, one of the other employed bees is selected at random. As the difference between the location of the current and the random bee decreases, the displacement of the current bee decreases too. Thus, as the search approaches to the optimum solution in the search space, the displacement magnitude is adaptively reduced: exploration is preferred at the beginning of the algorithm, and exploitation is preferred at the end of the algorithm.

In TSA [24], a new seed location is produced through one of the two principal equations of the algorithm (see [24,25] for details). In the first equation, the global best position obtained so far rules the creation of a new seed. In the second equation, a tree selected at random among all but the current tree rules the creation of a new seed. The decision of choosing either the first equation or the second one is made through the 'search tendency' parameter (ST). The higher ST, the more probable is the choice for the first equation. The smaller ST, the more probable is the choice for the second equation. A high value of ST yields a powerful local search and a high convergence speed, whereas a low value of ST causes slow convergence but powerful global search. In other words, the exploration and exploitation capabilities of the TSA are controlled by ST parameter. We notice that this capability is set *a priori* for an entire run of the algorithm.

A common and interesting property of ABC and TSA is that the evolution of the population of search agents is based on the selection of a random agent. However, they can only handle continuous search spaces.

Mainly, the rest of the swarm intelligence techniques which mimic the behaviour of groups of animals are as follows (see [10] and references inside): Marriage in Honey Bees Optimization Algorithm (MBO) in 2001, Artificial Fish-Swarm Algorithm (AFSA) in 2003, Termite Algorithm in 2005, Wasp Swarm Algorithm in 2007, Monkey Search in 2007, Bee Collecting Pollen Algorithm (BCPA) in 2008, Dolphin Partner Optimization (DPO) and Cuckoo search (CS) [26] in 2009, Firefly Algorithm (FA) in 2010, Bird Mating Optimizer (BMO) in 2012, Krill Herd (KH) in 2012, Fruit fly Optimization Algorithm (FOA) in 2012, Glowworm Swarm Optimization [27] in 2016, artificial algae algorithm (AAA) [28] in 2018.

We now focus on methods which are based on the division of the search agents into 'groups' also called 'tribes'. Tribes PSO has first been proposed in [29] and further developed in [30]. Its performance has been analysed in [31]. The principles of tribes PSO is to reduce the number of parameters in PSO, through an adaptive process which modifies autonomously the number of particles. In [32] a grouped grey wolf optimizer is proposed which is a 'tribe version' of GWO: the grey wolves are divided into two groups, including a cooperative hunting group with four types of wolves and a random scout group with two types of wolves. Contrary to the tribes in [30], these two tribes of wolves are not defined symmetrically. The first group (hunters) is meant for both 'exploration' in a first phase of the algorithm, and then 'exploitation', that is, concentration around the global minimum when it is assumed to be found. The second group is meant for exploration, to randomly look for a potential prey.

In summary, as evoked by the "No Free Lunch" theorem [33], the diversity of meta-heuristic algorithms comes from the fact that general applicability comes at the cost of domain specific performance, and that there is no one best approach for all domains. Further, there are no guarantees of finding a globally optimal solution, even within a specified tolerance.

2.2. Discrete bio-inspired optimization

The methods which aim at estimating the best integer values within an allowed set by minimizing some linear criterion are called integer linear programming (ILP). When only part of the values are enforced to be integer, one refers to mixed-integer linear programming (MILP).

Discrete bio-inspired optimization methods are meant to minimize some non-linear criteria. A major discrete bio-optimization method is ant colony optimization (ACO) [34]. ACO is motivated by the natural collective behaviour of real-world ant colonies. Artificial ants used in ACO are procedures of solution construction that probabilistically build a solution by iteratively adding solution components to partial solutions [34] by taking into account heuristic information about the problem instance being solved, if available, and (artificial) pheromone trails which change dynamically at runtime to reflect the agents' acquired search experience.

The other swarm intelligence methods, which are originally dedicated to continuous search spaces, have been firstly adapted to binary problems. Binary optimization is a subfield of discrete optimization, in the sense that the research space is restricted to the set $\{0, 1\}$ for all parameters. In [35] a binary version of PSO (BPSO) is proposed. In [36] a mimetic binary particle swarm optimization scheme is introduced based on hybrid local and global searches in BPSO. In [37], the first binary version of Cuckoo search was proposed.

After this, in [38], a binary version of the TSA method has been created.

In [39] a binary GWO is proposed and used to select an optimal feature subset for classification purposes. In [40], the first discrete version of GWO which is not restricted to binary search spaces is proposed, to address a combinatorial problem. A multi-objective mixed integer programming model is formulated. These advances are driven by an application of job scheduling for an optimal assignment of machines. One of the contributions in [40] consists in turning the computation of the influence of the leaders into a decision process: one leader is selected at random, between the three leaders, at each iteration. Job permutation is performed with a probability $\frac{1}{3}$ for each leader at any iteration. Machine assignment is performed with a probability which depends on the iteration index.

In [41], to further enhance the exploration, the authors embed a genetic operator, namely permutation, into their discrete GWO algorithm. The proposed method is called hybrid multi-objective grey wolf optimizer because GWO and GA are mixed together. In [42], an improved discrete cuckoo optimization algorithm is proposed for a flowshop scheduling problem.

We notice that, to the best of our knowledge, no mixed GWO has been proposed yet. Despite, there may be numerous applications which require the estimation of interdependent parameters taking their values in both discrete and continuous spaces. The proposed mixed GWO is driven by an image processing application.

2.3. Application of bio-inspired optimization to image processing

The physics-based WDO method has been applied to solve an issue of satellite image thresholding [5]. In [6], a genetic algorithm is compared to a simulated annealing algorithm with a view to reconstruct an image through thin scattering media.

As concerns swarm intelligence algorithms, they have been used essentially for classification issues, and more precisely for the estimation of the parameters of SVM (support vector machines) classifiers [43–45]. In [46] a 'firefly-SVM' algorithm is proposed to train all parameters of the SVM simultaneously. In [47], GWO estimates the parameters of an SVM classifier: 3-fold validation is performed for colour image classification. GWO has also been applied to the training of Multi-Layer Perceptron [20]. Bio-inspired optimization is undoubtedly of great interest for the processing of the images provided by optical remote sensing instruments [48,49]. Indeed, their huge dimensionality and complex data structure yield nonlinear optimization problems [48]. For instance, in [50], GWO is adapted to band selection in hyperspectral data. Binary versions of GWO have been developed for image processing applications: feature selection [39], and classification of cervix lesion images [51].

Discrete bio-inspired methods (not restricted to binary) have been firstly driven by engineering applications. A multi-objective discrete grey wolf optimizer (MODGWO) has been derived for a job scheduling purpose in [40,41]. This MODGWO algorithm is detailed in the Appendix , in Algorithm 7.

To the best of our knowledge, and due to their novelty, the discrete versions of GWO which have been applied to solve an image processing issue are restricted to binary. Moreover, no mixed bio-inspired optimization method has been developed to solve any image processing issue.

2.4. Comparative discussion about optimization methods

In this subsection, we aim at showing that GWO is an appropriate paradigm to create a mixed bio-inspired optimization method, and that other algorithms such as ABC or TSA can inspire us to ensure good convergence capabilities to the proposed method.

Ant Colony Optimization (ACO) [34] is dedicated to optimization problems with only discrete parameters: combinatorial problems such as the travelling salesman [52]. The specificity of this problem is that the same value cannot be present several times in a set of solutions. This is not the case for the considered application, where two unknown parameters may bear the same value.

In the considered application of denoising and unmixing of multispectral images, all parameters should be estimated jointly, within their whole range of possible values, which should not change across the iterations. Hence, methods which belong to the family of ACO do not seem to be appropriate for this application. As concerns ABC and TSA, they have been originally dedicated to continuous optimization problems. A valuable effort has been performed to adapt TSA into a binary optimization problem [38], but a discrete version, not restricted to binary, can still be considered, to the best of our knowledge, as future work for [38]. A careful study of the existing literature shows that the most recent advances on discrete optimization methods which are not restricted to binary problems concern GWO [14,40,41]. For all these reasons, it seems appropriate to choose GWO as a paradigm to develop a mixed bio-inspired optimization method.

The discrete version of GWO proposed in [40] has much merit in the sense that, for the first time, a variant of GWO is proposed to search discrete spaces. For this, the computation of a centroid which accounts simultaneously for the location of the three leaders is replaced by a selection process, where, at each iteration, one leader is selected at random. The update rules proposed in [40] are certainly valid for the application considered therein. Though, one can notice that some key properties of the seminal GWO [10] are lost in this discrete version [40]:

- for each of its components, a wolf is not displaced *towards* the selected leader. It actually takes the value of the selected leader;
- in the second update rule, in Eq. A.3 of Algorithm 7 in the Appendix , there seem to be firstly an 'exploitation' phase where importance is given to the leaders at the detriment of the other wolves, and thereafter an exploration phase where importance is given to the wolf selected at random at the detriment of the leaders.

In [41], the authors modify their proposed discrete grey wolf optimizer as a hybrid grey wolf optimizer. This hybrid optimizer includes a genetic search operator: crossover and then mutation are applied to enhance the exploration phase. However, we wish to better distinguish the exploration phase from the exploitation phase. We also notice that the methods proposed in [40] and [41] are only valid when the search spaces are strictly discrete.

Though ABC and TSA are still essentially dedicated to continuous optimization, some of their key features inspire us to build an improved version of GWO with well-balanced exploration and exploitation capacities: what is original in ABC and TSA is that not only the agent with the best score obtained so far but also some random agents are used in the update rules.

For the first time in this paper, we propose a new bio-inspired optimization method which is able to handle mixed problems, that is, problems with both discrete and continuous variables. In some applications, some parameters take their values in a discrete space, while other parameters take their values in a continuous space. Also, all parameters may be interdependent. An example in the image processing field will be presented further in the paper. In Section 3, we explain the theoretical aspects of the proposed mixed grey wolf optimizer.

3. Proposed mixed grey wolf optimizer

Our general goal is to propose a mixed GWO which can handle a problem with both discrete and continuous variables. In other words, the research space can be discrete for some of the parameters to be estimated, and continuous for other parameters. Moreover, it can also be used for problems solely composed of discrete variable or solely composed of continuous variables. The discrete process is applied to a subset of the expected parameters, taking their values in discrete search spaces, while the continuous process is applied independently to the other parameters, taking their values in continuous search spaces.

In this section, we firstly propose, in Section 3.1 a discrete bio-inspired optimization algorithm, that we will call 'improved discrete GWO'. Secondly, we explain in Section 3.2 what is the exact continuous counterpart of the proposed discrete GWO. We combine these two algorithms in a mixed grey wolf optimizer presented in Section 3.3, which handles both discrete and continuous parameters.

For sake of coherence with the original GWO [10], we still use a parameter, also denoted by a, to distinguish an exploration phase to an exploitation phase. The scalar a is decreased from 2 to 0, but in opposite to the seminal version of GWO [10], a does not decrease regularly. A parameter η is added in the model so that:

$$a = 2(1 - \frac{\text{iter}^{\eta}}{{T_{max}}^{\eta}}) \tag{1}$$

We notice that the modified GWO implemented in [11] is a particular case where $\eta = 2$.

With this parameter η, the amount of iterations spent on the exploration phase $t_{exploration}$ becomes :

$$t_{exploration} = \frac{T_{max}}{2^{\frac{1}{\eta}}} \tag{2}$$

Thus, it is possible to freely decide if the GWO algorithm must emphasize the exploration phase ($\eta > 1$), emphasize the exploitation phase ($\eta < 1$) or spend an equal amount of iterations on both phases ($\eta = 1$).

3.1. Improved discrete GWO

The idea behind the discrete grey wolf optimizer is that search spaces are discrete, and that not only vectors with values are defined for each wolf but also vectors with indexes. Still, a search agent or 'wolf' at iteration iter is denoted by $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ and contains N components which are candidate values of expected parameters.

Algorithm 1 details the proposed improved discrete GWO.

We notice that a value located at the component with index h_i in $\mathbf{d}_i^{ind}$ is denoted by $K_i^{h_i}$. For instance, for the vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter}) = \left[K_1^2, \ldots, K_N^5\right]$, the associated vector of indexes is $\mathbf{h}_q(\text{iter}) = [2, \ldots, 5]$.

We predict that the performance of our improved discrete GWO could be better than the performances of the MODGWO proposed in [40] which is suited for a specific combinatorial multi-objective optimization problem.

In Algorithm 2, we detail the wolf update rule for one wolf index $q \in [1, \ldots, Q]^T$, at any iteration index iter.

Here are some details about the proposed improved discrete GWO algorithm:

Algorithm 1 Pseudo-code: Improved Discrete Grey Wolf Optimization for multiple parameter estimation

Inputs: fitness function, number N of expected parameters, small factor ϵ set by the user, to stop the algorithm, maximum number of iterations T_{max}.
For each parameter indexed by $i = 1, \ldots, N$: the search space $\mathbf{d}_i^{ind}$ with H_i possible values.

1. Set iteration number iter = 1.
   Create an initial set of index vectors $\mathbf{h}_q(\text{iter})$, $q = 1, \ldots, Q$. For each index i between 1 and N, a component $h_i(\text{iter})$ of $\mathbf{h}_q(\text{iter})$ is an integer value between 1 and H_i.
   Create an initial herd composed of Q wolves $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$ with the N required parameter values. This initial population takes the form of a matrix with Q rows and N columns. For each index q, the components of $x_i(\text{iter})$ are $\mathbf{d}_i^{ind}(h_i(\text{iter}))$.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Sort the wolves through their fitness value and update the wolves which hold the first, second and third best fitness value: store the corresponding vectors of indexes $\mathbf{h}_\alpha, \mathbf{h}_\beta, \mathbf{h}_\delta$ and vectors of values $\mathbf{x}_\alpha^{\mathbf{k}}, \mathbf{x}_\beta^{\mathbf{k}}, \mathbf{x}_\delta^{\mathbf{k}}$.
4. If $a > 1$, select two wolves $\rho 1$ and $\rho 2$, randomly among the herd of Q wolves, with $\rho 1 \neq \rho 2$. Store the corresponding vectors of indexes $\mathbf{h}_{\rho 1}, \mathbf{h}_{\rho 2}$ and vectors of values $\mathbf{x}_{\rho 1}^{\mathbf{k}}, \mathbf{x}_{\rho 2}^{\mathbf{k}}$.
   Else if $a \leq 1$, go to step 5.
5. Repeat steps for each wolf q, $q = 1, \ldots, Q$, with vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ and vector of indexes $\mathbf{h}_q(\text{iter})$:
   Apply Algorithm 2.
6. Exchange the current population with the new one, obtained at step 5
7. If iter $< \mathrm{T_{max}}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, increase iter, and go to step 2.

Output: estimated parameter values $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$ contained in $\mathbf{x}_\alpha^{\mathbf{k}}$.

At step 1, if $a > 1$, the leader is selected randomly among the α, β, δ, $\rho 1$, and $\rho 2$ wolves:

$$\mathbf{x}_l^{\mathbf{k}} = \begin{cases} \mathbf{x}_\alpha^{\mathbf{k}} & \text{if} \quad r \leq \frac{a}{10} \\ \mathbf{x}_\beta^{\mathbf{k}} & \text{if} \quad r > \frac{a}{10} \quad \text{and} \quad r \leq \frac{2a}{10} \\ \mathbf{x}_\delta^{\mathbf{k}} & \text{if} \quad r > \frac{2a}{10} \quad \text{and} \quad r \leq \frac{3a}{10} \\ \mathbf{x}_{\rho 1}^{\mathbf{k}} & \text{if} \quad r > \frac{3a}{10} \quad \text{and} \quad r \leq \frac{4a}{10} \\ \mathbf{x}_{\rho 2}^{\mathbf{k}} & \text{if} \quad r > \frac{4a}{10} \quad \text{and} \quad r \leq \frac{5a}{10} \\ \mathbf{x}_\alpha^{\mathbf{k}} & \text{if} \quad r > \frac{5a}{10} \end{cases} \tag{3}$$

where r is a random value in $\mathbb{R}$, between 0 and 1.

if $a \leq 1$, the leader is selected randomly among the α, β, and δ wolves:

$$\mathbf{x}_l^{\mathbf{k}} = \begin{cases} \mathbf{x}_\alpha^{\mathbf{k}} & \text{if} \quad r \leq \frac{a}{6} \\ \mathbf{x}_\beta^{\mathbf{k}} & \text{if} \quad r > \frac{a}{6} \quad \text{and} \quad r \leq \frac{2a}{6} \\ \mathbf{x}_\delta^{\mathbf{k}} & \text{if} \quad r > \frac{2a}{6} \quad \text{and} \quad r \leq \frac{3a}{6} \\ \mathbf{x}_\alpha^{\mathbf{k}} & \text{if} \quad r > \frac{3a}{6} \end{cases} \tag{4}$$

where r is a random value in $\mathbb{R}$, between 0 and 1.

As the parameter a is decreasing from 2 to 0 across the iterations, it is more and more probable for α to be chosen as the leader. The random wolves may be selected during the first part of the process, when $a > 1$, and cannot be selected during the second

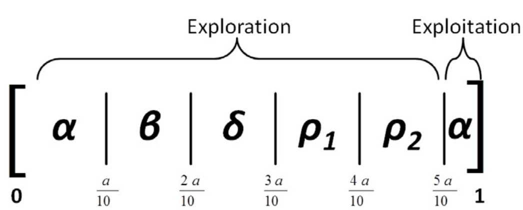
Algorithm 2 Pseudo-code: Update rule for Improved Discrete Grey Wolf Optimization for multiple parameter estimation

Inputs: Vector of indexes $\mathbf{h}_q(\text{iter})$, vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$.

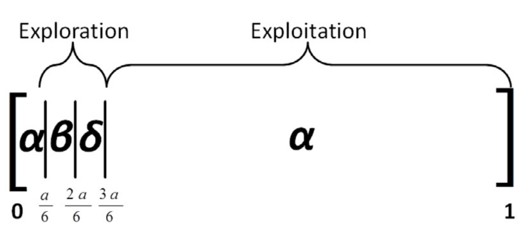
1. Select leader:
   (a) Let $\mathbf{x}_l^{\mathbf{k}}$ denote the vector of values for the selected leader, chosen randomly among $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, $\mathbf{x}_\delta^{\mathbf{k}}$, $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, and $\mathbf{x}_{\rho 2}^{\mathbf{k}}$. This selection process is detailed in Eqs. (3) to (4).
   (b) Store the vector of indexes corresponding to $\mathbf{x}_l^{\mathbf{k}}$ in a vector denoted by $\mathbf{h}_{\mathrm{l}}$.
2. Update wolf q:
   for each index $i = 1, \dots, N$
   (a) Compute the updated component $h_i(\text{iter}+1)$.
   (b) Compute the updated component $x_i(\text{iter}+1)$.

   This update process is detailed in Eqs. (5), (7), (8)
3. Store the N components obtained at steps 2a and 2b:
   (a) Get the updated vector of indexes: $\mathbf{h}_q(\text{iter}+1) = [h_1(\text{iter}+1), \dots, h_i(\text{iter}+1), \dots, h_N(\text{iter}+1)]^T$.
   (b) Get the updated vector of values: $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1) = [x_1(\text{iter}+1), \dots, x_i(\text{iter}+1), \dots, x_N(\text{iter}+1)]^T$.

Output: $\mathbf{h}_q(\text{iter}+1)$, $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$



(a) Interval of possible values of r: possibilities of leader selection with $a \simeq 2$



(b) Interval of possible values of r: possibilities of leader selection with $a \simeq 0$

Fig. 1. Evolution of the possibilities of leader selection according to the decrease of a.

part of the process, when $a \leq 1$. This is coherent with the paradigm of the original Grey Wolf Optimizer [10], where exploration is emphasized when $a > 1$, and exploitation is emphasized when $a \leq 1$.

The Fig. 1 illustrates how the separation between the exploration phase and the exploitation phase is done in the proposed algorithm. At the beginning of the process, when $a \simeq 2$, each leader may be chosen with the same probability, which permits to 'explore' the search space. When a decreases towards 0, the last possibility in Eqs. (3) and (4) may be selected with a higher probability, which permits to 'exploit' the corresponding promising location in the search space. Indeed, a wolf will then move according to the leader α, which is the leader with the best fitness.

The leader $\mathbf{x}_l^{\mathbf{k}}$ will rule the displacement of wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ as explained below.

At step 2a, each component of the vector of indexes $\mathbf{h}_q(\text{iter})$ is updated as follows:

For each index i, $i = 1, \dots, N$:

$$h_i(\text{iter}+1) = (h_i(\text{iter}) + \Delta\ sgn(h_i^{\mathrm{l}} - h_i(\text{iter})))\ mod\ H_i \tag{5}$$

where:

- $sgn(\cdot)$ denotes the sign function, which is such that $sgn(z) = -1$ if $z < 0$, $sgn(z) = 0$ if $z = 0$, and $sgn(z) = 1$ if $z > 0$ for any real value z;
- mod denotes the 'Modulo' operator, defined as follows: whatever the real values $u \in \mathbb{R}_+$ and $v \in \mathbb{R}_+^*$:

$$u\ mod\ v = \begin{cases} u - v\lfloor u/v \rfloor & \text{if} \quad u \neq v \\ v & \text{if} \quad u = v, \ \text{or } u = 0 \end{cases} \tag{6}$$

where $\lfloor \cdot \rfloor$ denotes integer part.
- Δ is computed as follows:

$$\Delta = \begin{cases} 1 & \text{if} \quad \phi \leq \frac{a}{6} \\ 2 & \text{if} \quad \phi > \frac{a}{6} \quad \text{and} \quad \phi \leq \frac{2a}{6} \\ 4 & \text{if} \quad \phi > \frac{2a}{6} \quad \text{and} \quad \phi \leq \frac{3a}{6} \\ 1 & \text{if} \quad \phi > \frac{3a}{6} \end{cases} \tag{7}$$

where ϕ is a random value in $\mathbb{R}$, between 0 and 1.

As the parameter a is decreasing from 2 to 0 across the iterations, it is more and more probable for the value 1 to be chosen, and less and less probable for the larger values such as 2 and 4 to be chosen. This means that, at the beginning of the process, the wolf which is currently modified may be displaced by 2 or 4 components in the direction of the leader with a rather high probability, and that, at the end of the process, it is highly probable that it will be displaced by only 1 component in the direction of the leader.

This permits to explore the research space with large displacement values at the beginning of the optimization process, and to perform exploitation, with rather small displacements towards the selected leader at the end of the optimization process.

Then, at step 3a the updated values $h_i(\text{iter}+1)$, $i = 1, \dots, N$, computed in Eq. (8) are stored in vector $\mathbf{h}_q(\text{iter}+1)$.

At step 2b:

each component of the updated vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$ is computed as follows:

$$x_i(\text{iter}+1) = \mathbf{d}_i^{val}(h_i(\text{iter}+1)) \tag{8}$$

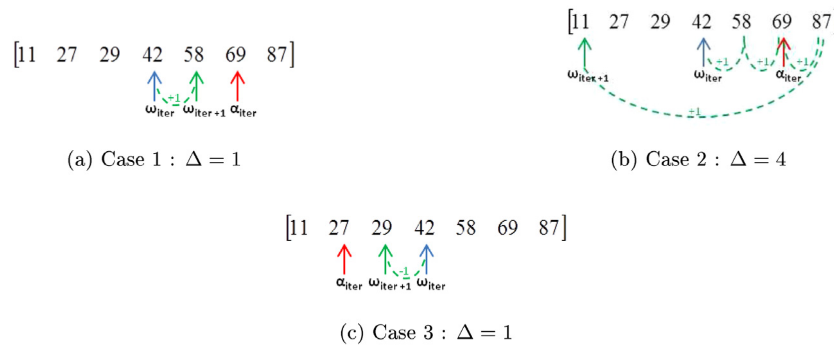
Then, at step 3b the updated values $x_i(\text{iter}+1)$, $i = 1, \dots, N$, computed in Eq. (8) are stored in vector $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$.

When each vector, or 'search agent' $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$, $q = 1, \dots, Q$, has been computed at step 5, the whole population of wolves is updated at step 6, and one may increase the iteration index or terminate the algorithm. At step 7, vector $\mathbf{x}_\alpha^{\mathbf{k}}$ contains the estimated parameters $\hat{K}_1, \hat{K}_2, \dots, \hat{K}_N$.

The Fig. 2 illustrates a simple example with 3 possible updates for a wolf ω. In this example, the α leader has been selected as the leader that ω must follow. Moreover, there is only one parameter to search with $\mathbf{d}^{val} = [11, 27, 29, 42, 58, 69, 87]^T$ and $H = 7$.

Table A.1 presents, in cases 2a, 2b, and 2c, the index $h(iter)$, the updated index $h(iter+1)$ and the updated value $x(iter+1) = \mathbf{d}^{val}(h(iter+1))$.

This improved discrete GWO is the most innovative part of the mixed GWO proposed in this paper. In Section 3.2, we present its continuous counterpart.



(a) Case 1 : $\Delta = 1$

(b) Case 2 : $\Delta = 4$

(c) Case 3 : $\Delta = 1$

Fig. 2. Examplification of a wolf update with the proposed discrete GWO.

3.2. Global continuous grey wolf optimizer

In Algorithm 3, we detail the continuous version of wolf leader selection and wolf update for one parameter index $i \in [1, \ldots, N]^T$ and one wolf index $q \in [1, \ldots, Q]^T$, at any iteration index iter. To preserve the coherence with the proposed improved discrete grey wolf optimizer, we introduce two random wolves which may also contribute to the displacement of any wolf. As these random wolves help the algorithm to explore the search space and avoid local minima, we call this version of GWO the Global Continuous Grey Wolf Optimizer.

Algorithm 3 Pseudo-code: Global Continuous Grey Wolf Optimization for multiple parameter estimation

---

Inputs: $x_i(\text{iter})$, ith component of a given wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ at iteration iter; leaders α, β, δ.

1. Compute the contributions y_i^{α}, y_i^{β}, and y_i^{δ} of wolves α, β, and δ respectively to the displacement of the q^{th} wolf.
   This computation is detailed in Eqs. (11) and (12) below.
2. if $a > 1$ go to step 3, else if $a \leq 1$ go to step 4
3. Compute the contributions $y_i^{\rho 1}$ and $y_i^{\rho 2}$ of wolves $\rho 1$ and $\rho 2$, respectively to the displacement of the q^{th} wolf.
   This computation is detailed in Eqs. (11) and (12) below.
4. Compute the update position at the ith of the q^{th} wolf:
   if $a > 1$:
   $$x_i(\text{iter}+1) = \frac{1}{5}(y_i^{\alpha} + y_i^{\beta} + y_i^{\delta} + y_i^{\rho 1} + y_i^{\rho 2}) \tag{9}$$
   else if $a \leq 1$:
   $$x_i(\text{iter}+1) = \frac{1}{3}(y_i^{\alpha} + y_i^{\beta} + y_i^{\delta}) \tag{10}$$

Output: $x_i(\text{iter}+1)$

This updated position mentioned at step 4 of Algorithm 3 is computed as the equal contribution of the leaders α, β and δ. If $a > 1$, then the contributions of two random wolves $\rho 1$ and $\rho 2$ are also used. These contributions are computed as follows:

$$\begin{cases} y_i^{\alpha} = x_i^{\alpha} - b_1 \cdot d_i^{\alpha}, \\ y_i^{\beta} = x_i^{\beta} - b_2 \cdot d_i^{\beta}, \\ y_i^{\delta} = x_i^{\delta} - b_3 \cdot d_i^{\delta}, \\ y_i^{\rho 1} = x_i^{\rho 1} - b_4 \cdot d_i^{\rho 1}, \\ y_i^{\rho 2} = x_i^{\rho 2} - b_5 \cdot d_i^{\rho 2} \end{cases} \tag{11}$$

with:

$$\begin{cases} d_i^{\alpha} = |c_1 \cdot x_i^{\alpha} - x_i(\text{iter})|, \\ d_i^{\beta} = |c_2 \cdot x_i^{\beta} - x_i(\text{iter})|, \\ d_i^{\delta} = |c_3 \cdot x_i^{\delta} - x_i(\text{iter})|, \\ d_i^{\rho 1} = |c_4 \cdot x_i^{\rho 1} - x_i(\text{iter})|, \\ d_i^{\rho 2} = |c_5 \cdot x_i^{\rho 2} - x_i(\text{iter})| \end{cases} \tag{12}$$

where the scalars b and c are calculated as in the vanilla GWO: $b = 2ar_1 - a$ and $c = 2r_2$. In these expressions, r_1 and r_2 are random scalars between 0 and 1.

3.3. Extension to a mixed grey wolf optimizer

In a same problem, the expected parameters do not necessarily belong to the same type of searching space. Some of these parameters will be continuous while the others will be discrete and though, they may be interdependent and should be estimated simultaneously. We will refer to such problems as mixed problems. Combining the improved discrete and global continuous GWO methods proposed in Sections 3.1 and 3.2 respectively, we propose a mixed GWO method. The following notations will be specifically used for the mixed GWO, in addition to the notations presented in Section 3.1:

- Assuming that the ith parameter K_i takes its values in a continuous search space, its minimum acceptable value is denoted by K_i^{min} and its maximum acceptable value is denoted by K_i^{max};
- the interval of acceptable values for the parameter K_i in a continuous search space is denoted by $\mathbf{d}_i^{val} = \left[K_i^{min}; K_i^{max}\right]^T$.

Algorithm 4 describes the proposed mixed grey wolf optimization while Figs. A.14 and A.15, available in the Appendix , describe its flowchart.

Here are some remarks about Algorithm 4, the mixed GWO algorithm:

The components of each search agent may be updated with continuous update rules (see step 5b) or with discrete update rules (see step 5d), whether they belong to continuous or discrete search spaces. However, at a given iteration iter, a given wolf q characterized by the vector of values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$ where continuous and discrete parameters are mixed yields one common fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ (see step 2).

Further in the paper, we will distinguish between two versions of our mixed grey wolf optimizer, depending on the expression of the parameter a: in the mixed GWO (denoted by mixedGWO) the parameter a is expressed as follows:

$$a = 2(1 - \frac{\text{iter}^2}{{T_{max}}^2}) \tag{13}$$

Algorithm 4 Pseudo-code: Mixed Grey Wolf Optimization for multiple parameter estimation

Inputs: fitness function, small factor ϵ set by the user, to stop the algorithm.

1. Set iteration number iter $=$ 1, create an initial herd composed of Q wolves with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\text{iter}), q = 1, \ldots, Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter}), q = 1, \ldots, Q$.
3. Sort the wolves through their fitness value and update the α, β, and δ wolves which hold respectively the first, second and third best fitness value. Store their position in vectors $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, and $\mathbf{x}_\delta^{\mathbf{k}}$ respectively. For the discrete parameters, store the corresponding vectors of indexes of their discrete components $\mathbf{h}_\alpha$, $\mathbf{h}_\beta$ and $\mathbf{h}_\delta$.
4. If $a > 1$, select two wolves $\rho 1$ and $\rho 2$, randomly among the herd of Q wolves, with $\rho 1 \neq \rho 2$. Store the vectors of values $\mathbf{x}_{\rho 1}^{\mathbf{k}}$, $\mathbf{x}_{\rho 2}^{\mathbf{k}}$ and, for the discrete parameters, the corresponding vectors of indexes $\mathbf{h}_{\rho 1}$, $\mathbf{h}_{\rho 2}$.
   Else if $a \leq 1$, go to step 5.
5. Repeat steps for each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter}), q = 1, \ldots, Q$:
   For each component $x_i(\text{iter})$ with $i = 1, \ldots, N$:
   (a) if the ith parameter K_i takes its values in a continuous search space then go to step 5b,
   else if K_i takes its values in a discrete search space then go to step 5d.
   (b) Apply the continuous versions of wolf update and displacement, proposed in Algorithm 3.
   Skip steps 5d to 5e.
   (c) select the leader $\mathbf{x}_l^{\mathbf{k}}$, and the displacement magnitude Δ which will be used for the discrete parameters. For this, refer to Eqs. (3), (4), (5), and (7).
   (d) apply step 2a of algorithm 1 to get $h_i(\text{iter}+1)$ as in Eq. (5)
   (e) apply step 2b of algorithm 1 to get $x_i(\text{iter}+1)$ as in Eq. (8)
6. Exchange the current population with the new one, obtained at step 5
7. If iter $< \mathrm{T_{max}}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, increase iter, and go to step 2.

Output: estimated parameter values $\hat{K_1}, \hat{K_2}, \ldots, \hat{K_N}$

In a second version, that we call adaptive mixed GWO and we denote by amixedGWO, parameter a is such that:

$$a = \begin{cases} 2\left(1 - \frac{\text{iter}^\eta}{(\mathrm{T}_{max}/2)^\eta}\right) & \text{if} \quad \text{iter} \leq \mathrm{T}_{max}/2 \\ 2\left(1 - \frac{(\text{iter}-\mathrm{T}_{max}/2)^{\frac{1}{\eta}}}{(\mathrm{T}_{max}/2)^{\frac{1}{\eta}}}\right) & \text{if} \quad \text{iter} > \mathrm{T}_{max}/2 \end{cases} \tag{14}$$

An elevated value of η encourages exploration during the first phase, from iter $= 1$ to iter $= \mathrm{T}_{max}/2$; and exploitation during the second phase, from iter $= \mathrm{T}_{max}/2 + 1$ to iter $= \mathrm{T}_{max}$.

The term 'adaptive' means that the expression of a is 'adapted' depending on the iteration index.

4. Performance evaluation on synthetic data

In this section, we evaluate the performances of the proposed methods and comparative bio-inspired optimization methods, when applied to the minimization of various unimodal or multimodal test functions. Unimodal functions exhibit only one global minimum and no relative minima, whereas multimodal functions exhibit several relative minima.

4.1. Experimental conditions

In this subsection, we present the experimental conditions which, unless specified, are common to the whole Section 4. Unless specified, the results are computed over $M = 30$ independent runs, $Q = 30$ search agents and $\mathrm{T}_{max} = 3000$ iterations for each algorithm. The value for η in amixedGWO is empirically set to $\eta = 3$, after performing a comparative evaluation with several values between 1 and 10.

These experimental conditions are the same as in [11].

We have implemented the GWO, mGWO, and MODGWO (see [10] and Algorithm 6, and [40] and Algorithm 7) comparative methods so that they can be compared with the proposed mixedGWO and amixedGWO (adaptive mixed GWO) in the same conditions.

In Section 4, programs were written in $C++$, and executed on a PC running Windows, with a 3 GHz double core and 3GB RAM.

The performance metrics are the following:

We consider, for the mth run, $f(\mathbf{x}_\alpha^{\mathbf{k}})_m$, the fitness value obtained at iteration T_{max} for the best wolf, namely α:

• The statistical mean is the average (Avg.) of fitness values acquired while running an optimization algorithm for different M runs. The average performance of a given stochastic optimizer is formulated in Eq. (15):

$$Avg = \frac{1}{M}\sum_{m=1}^{M} f(\mathbf{x}_\alpha^{\mathbf{k}})_m \tag{15}$$

• The standard deviation (Std.) is a representation for the variation of the obtained best solutions found while running a stochastic optimizer for M different runs. Std is used as an indicator for optimizer stability and robustness. If Std. is small, this means that the optimizer converges always towards the same solution. Conversely, if Std. is large, the results obtained are much more random and the optimizer is less reliable. The standard deviation is formulated in Eq. (16):

$$Std = \sqrt{\frac{1}{M}\sum_{m=1}^{M}(f(\mathbf{x}_\alpha^{\mathbf{k}})_m - Avg)^2} \tag{16}$$

• The median (Med.) is the value separating the $\frac{M}{2}$ higher half from the $\frac{M}{2}$ lower half of values obtained for $f(\mathbf{x}_\alpha^{\mathbf{k}})$.

4.2. Benchmark functions

In this subsection, the benchmark functions used for the rest of the section are presented.

Tests have been run on 20 benchmark functions in order to compare the performance on a fully continuous problem of the proposed mixedGWO algorithm and its adaptive version amixedGWO to the vanilla GWO method and its modified version mGWO proposed by Mittal et al. in [11].

The benchmark functions have been taken from [11] and can be divided in 3 categories : the Unimodal functions, the Multimodal functions and the fixed dimension Multi-modal functions. The functions' expressions are presented on Tables A.2–A.4. The unimodal functions exhibit only one minimum, which is the expected global minimum. The multimodal functions exhibit several minima, among them the expected global minimum.

Doubting from the expressions of the functions F12, F13 and F19 presented in [11], we do not present results related to these functions. For instance, we do not know the value of the constant used in function F19.

4.3. Numerical results on synthetic data

In this subsection, the numerical results of our methods and the state-of-the-art are presented. The Sections 4.3.1–4.3.3 are respectively the results on the continuous functions, the discrete functions and the mixed functions.

4.3.1. Results on continuous functions

The results we have computed are available in Table A.5 for the Unimodal functions, Table A.6 for the Multi-modal functions and Table A.7 for the fixed dimension multi-modal functions. Although the average found minimum are close to each other, it can be seen in Tables A.5 and A.6 that the vanilla GWO method and the mGWO method remain slightly better than our method for the Unimodal functions and the Multi-modal functions. However, it can be seen in Table A.7 that the proposed mixedGWO and amixedGWO methods outperform the state-of-the-art methods for the fixed dimension multi-modal functions.

The Fig. 3 showcases the convergence plot of the four studied methods on some of the fixed dimension multi-modal functions, *e.g.* $F20$ to $F23$. Starting from the 1500th iteration, the curves of the proposed amixedGWO remains constant for the functions $F21$, $F22$ and $F23$, because the variations are on a scale inferior to 10^{-5}, therefore unseeable on the curves.

In all of the cases, either the mixedGWO or the amixedGWO converges faster than the state-of-the-art methods.

We compare the proposed method to the state of the art with a higher number of search agents $Q = 48$, but a smaller number of iterations $T_{max} = 20$. The comments that can be inferred by the results obtained are similar to the case where $Q = 30$ and $T_{max} = 3000$ and are presented in Table A.8.

To statistically validate the results obtained on Table A.7 with the fixed-dimension multi-modal benchmark functions, the Wilcoxon test is used. The obtained p-value for the continuous functions and the discrete functions are displayed on Table A.9. The level of significance is set as $p = 0.05$. A $(+)$ means that the first compared method is better than the second, a $(-)$ means that the second compared method is better than the first and a $(=)$ means that there is no significant difference between the two compared methods.

According to the p-values obtained with the Wilcoxon test, it appears that the proposed method is usually better than the GWO and the mGWO methods on the fixed dimension multi-modal benchmark functions. Indeed it is significantly better in 19 cases out of 36 and only significantly worse in 2 cases.

The proposed method amixedGWO has also been tested on the CEC2014 benchmark functions [53], except the hybrid functions, and compared to state-of-the-art methods such as PSO, GWO, ABC, TSA, a genetic algorithm (GA) and simulated annealing (SA) [6]. The average residual values obtained are available on Table A.10. The tests have been done in the same conditions as previously, that is $Q = 30$ and $T_{max} = 3000$. When SA is used, we set the number of 'control points' to 30, that is, the same value as the number of agents Q used in the other methods.

These results follow the logic of what has been observed previously: the proposed mixedGWO is not the best method on strictly continuous unimodal and multi-modal problems. However, the proposed method is not meant for these kinds of problem but rather for the fixed-dimension multi-modal ones and more particularly for the mixed problems. Moreover, it has been noted that, in these conditions, the proposed method is at least twice faster than the compared methods, for the same number of iterations and for the same number of search agents.

4.3.2. Results on discrete functions

The unimodal functions from F_1 to F_6 will be used to test the proposed method on discrete functions. A low dimension problem, with only three variables, is considered, to be close to the multidimensional image processing issue considered further.

The searching space is as follows: there will be a step of 1 in between each possible value for the functions F_1, F_3 and F_4 while the step will be of 0.5 for the functions F_2, F_5 and F_6.

The two proposed methods mixedGWO and amixedGWO are compared with the MODGWO method, which is presented in [40]. The results obtained for each algorithm are presented in Table A.11. For each method, in addition to the average result and the standard deviation, the medium value obtained is also indicated.

The results shown in Table A.11 confirm that the proposed methods (mixedGWO and amixedGWO) outperform the state-of-the-art method (MODGWO) on discrete problems. Moreover, as indicated by the medium value, the optimal minimum has been found on all the benchmark functions on at least 50% of the runs, even on the function F_5, which was not the case on the continuous functions.

The Table A.12, available in the Appendix , showcases the results obtained with the MODGWO methods with much more iterations. From these results, it appears that the MODGWO method needs an elevated number of iterations in order to converge and approach the optimal minimum still without reaching it.

To statistically validate the results obtained on Table A.11 with the discrete benchmark functions, the Wilcoxon test is used. The obtained p-values for the discrete benchmarks functions are displayed on Table A.13. The level of significance is set as $p = 0.05$. A $(+)$ means that the first compared method is better than the second, a $(-)$ means that the second compared method is better than the first and a $(=)$ means that there is no significant difference between the two compared methods.

According to the p-values obtained with the Wilcoxon test, it appears that the differences in the average values between the MODGWO method and the proposed methods are significant. However, the two proposed methods mixedGWO and amixedGWO seem not to hold significant differences between them.

4.3.3. Results on mixed functions

The Unimodal functions from F_1 to F_6 will be used to test the mixedGWO method on mixed functions.

To be as close as possible to the considered image processing application, there will only be 6 variables instead of 30. Out of those 6 variables, 4 will be discrete and 2 continuous. The searching space is always in the same range as in the continuous case. For the discrete values, there will be a step of 1 in between each possible value for the functions F_1, F_3 and F_4 while the step will be of 0.5 for the functions F_2, F_5 and F_6. To tackle mixed problems, the comparative methods PSO, GWO, mGWO, ABC, TSA, GA and SA should include a rounding process for the discrete parameters, what we wish to avoid here. Only the two proposed methods mixedGWO and amixedGWO will be tested here. The results obtained for each algorithm are presented in Table A.14. For each method, in addition to the average result and the standard deviation, the medium value obtained is also indicated. The medium values obtained indicate that the proposed methods have found the optimal minimum value in at least 50% of the runs on the benchmark functions F_1 to F_4. However, the proposed methods have difficulties to converge correctly on the function F_5, which is in accordance with the results obtained on the continuous benchmark functions. But the algorithms also converged correctly on the function F_6.

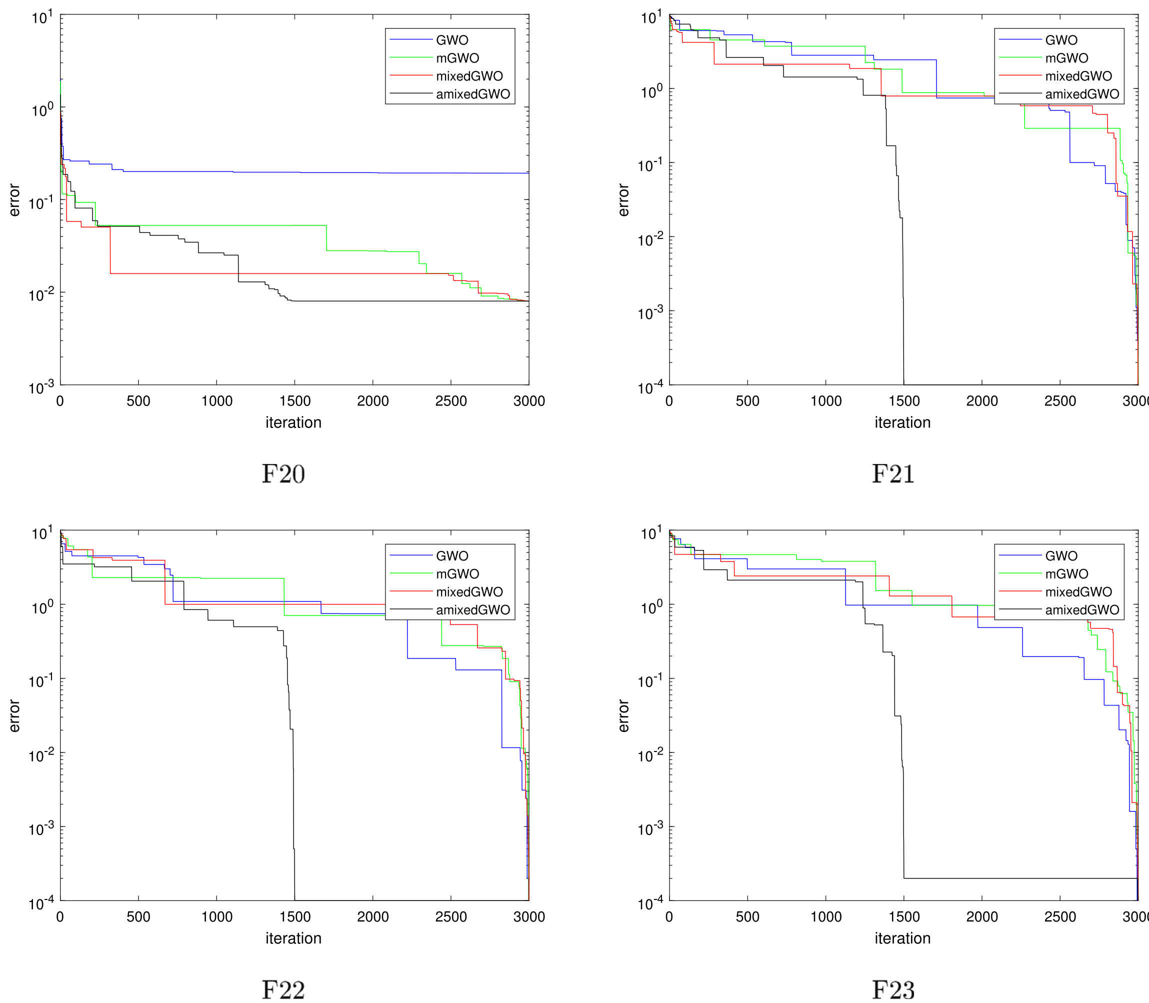


Fig. 3. Convergence plot of the functions 'F20', 'F21', 'F22' and 'F23'.

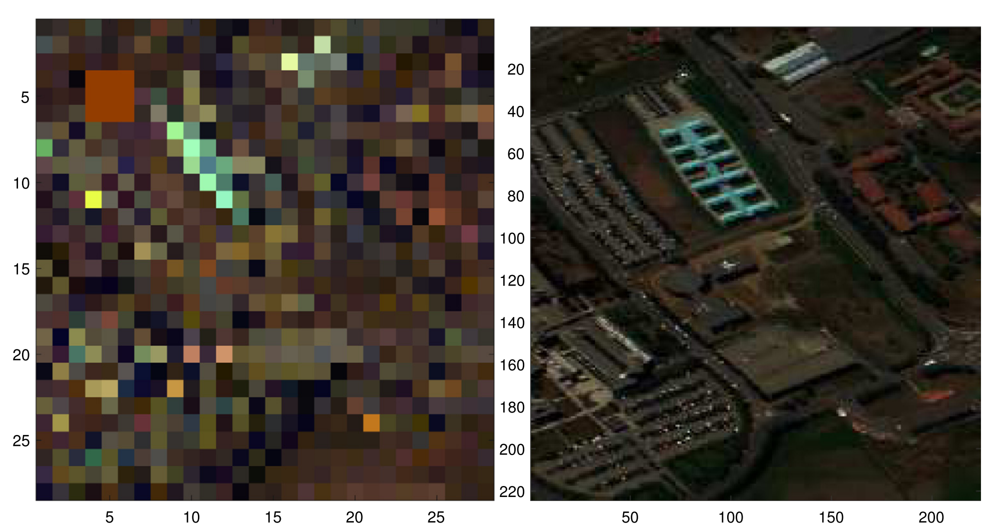


Fig. 4. PaviaU 32 × 32 × 4 and PaviaU 256 × 256 × 103: Noise-free images.

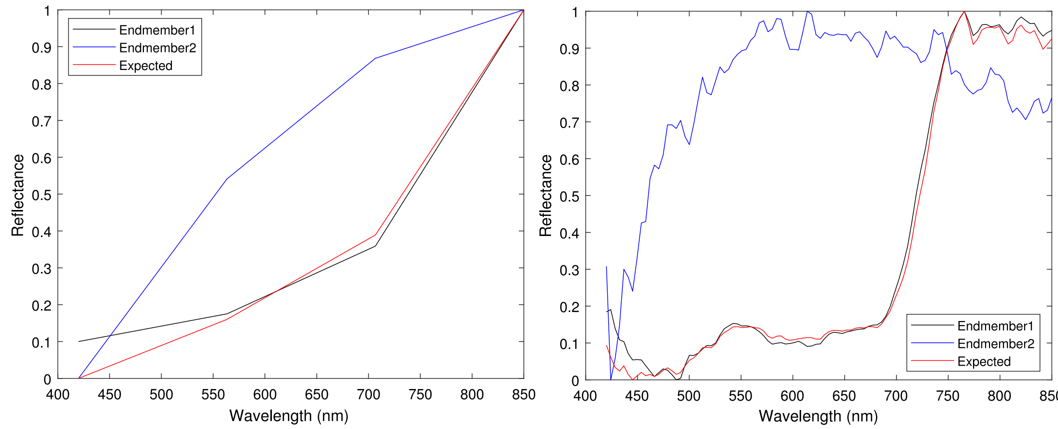


Fig. 5. PaviaU 32 × 32 × 4 and PaviaU 256 × 256 × 103: endmembers **s1** (black) and **s2** (blue), and expected spectrum **y** (red).

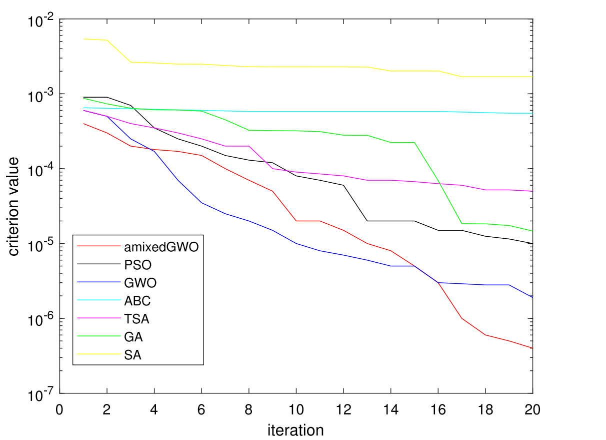


Fig. 6. Mean convergence plot with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$.

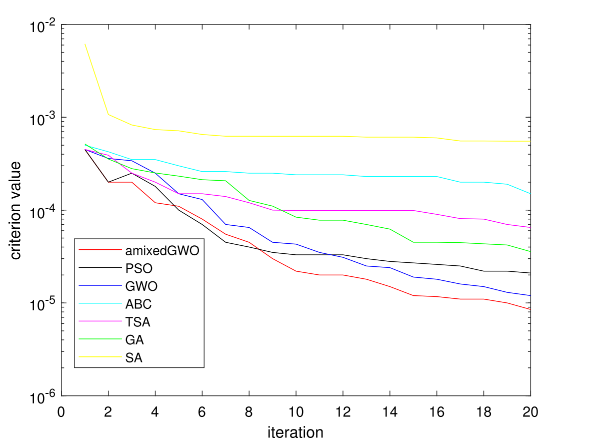


Fig. 7. Mean convergence plot with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$.

5. Application to multidimensional image processing

5.1. Short state-of-the-art about multispectral image processing with bio-inspired optimization methods

Generally speaking, an image is a multidimensional array, whose values are accessed *via* indices: we need two indices to access the values of a 2-dimensional (2-D) image, and 3 indices to access the values of a 3-dimensional (3-D) image. Such a multidimensional array is called 'tensor' [54].

A seminal work dedicated to tensor denoising consisted in adapting Wiener filtering in a tensor framework, yielding the Multiway Wiener Filtering (MWF) [55], a subspace-based method requiring the estimation of the dimension of the signal subspace, also called rank, along each mode. The following notations are adopted: $\mathcal{X}$ is the noise-free tensor, $\mathcal{R}$ is the noised tensor and $\hat{\mathcal{X}}$ is the estimated tensor. In this paper, modes are indexed by i, and third-order tensors are considered: multispectral images of size $I_1 \times I_2 \times I_3$. For $i = 1, 2$ or 3, I_i is the size of the multispectral image along the ith mode: I_1 is the number of rows, I_2 the number of columns, and I_3 the number of bands. Impaired multispectral images are expressed as: $\mathcal{R} = \mathcal{X} + \mathcal{N}$, where tensor $\mathcal{N}$ stands for additive independent and identically distributed zero-mean Gaussian noise.

A signal subspace value for mode i is denoted by K_i. We denote by $\hat{\mathcal{X}}(K_1, K_2, K_3)$ the estimate provided by Multiway Wiener Filtering applied to $\mathcal{R}$ with rank values K_1, K_2, K_3. Details about MWF can be found in [54,55].

In [56], MWF has been inserted into a wavelet framework to denoise images while preserving details. The advantage of this method is to preserve small features while denoising efficiently, but requires the knowledge of numerous rank values. In [57], a least squares (LS) criterion is minimized with PSO to estimate these rank values.

In [50], GWO is adapted for hyperspectral band selection: the authors propose to reduce the dimensionality of hyperspectral images to improve classification results while removing irrelevant spectral bands.

5.2. Simultaneous denoising and unmixing issue

Our purpose in this section is to show that the proposed mixed grey wolf optimizer is adequate to perform a simultaneous denoising and unmixing of multispectral images.

5.2.1. Description of the proposed application

Denoising multispectral images is generally a preliminary step before higher level image processing operations such as target detection [58] or spectra unmixing [59]. One issue in this context consists in finding a good compromise between the efficiency of the image denoising process, and the accuracy of the forthcoming unmixing process. Let $\mathbf{y} \in \mathbb{R}^{I_3}$ be one spectrum of the multispectral image $\mathcal{X}$. For this spectrum $\mathbf{y}$, supervised unmixing aims at estimating the contributions of the spectral signature of materials in the scene (called endmembers) for part of or all pixels in the scene.

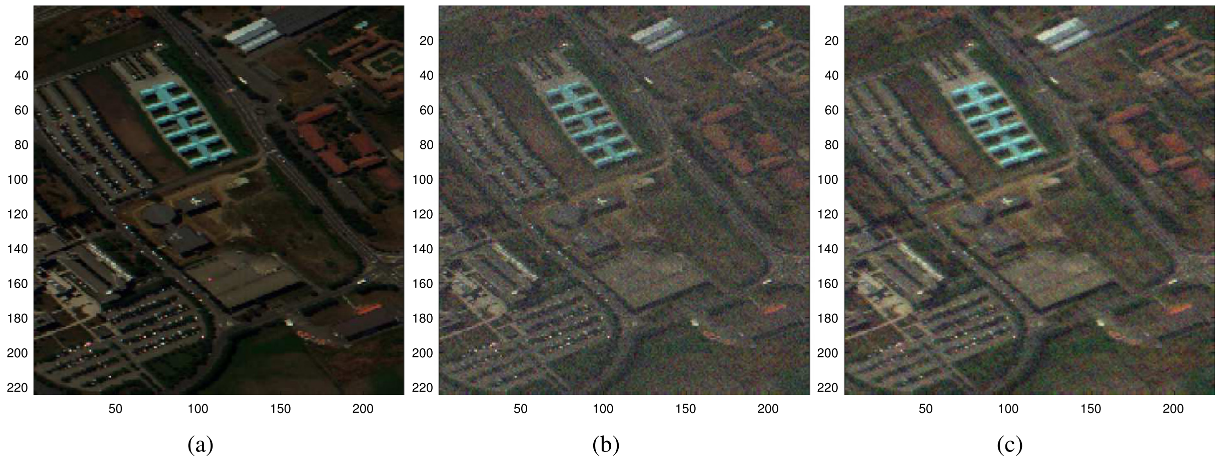


Fig. 8. PaviaU 256 × 256 × 103: (a) Noise-free, (b) impaired 10 dB, and (c) reference images.

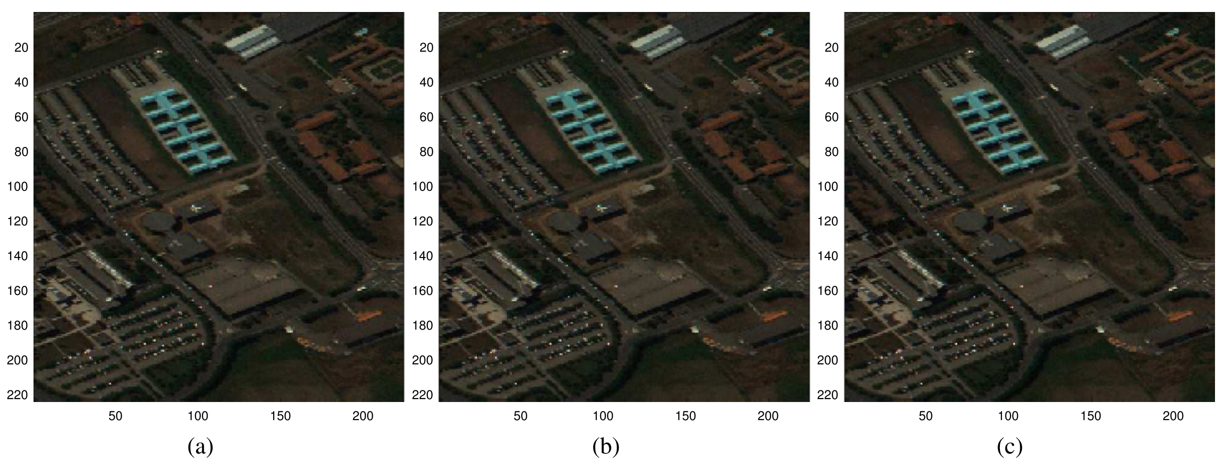


Fig. 9. PaviaU 256 × 256 × 103. Denoised images obtained with (a) amixedGWO, (b) PSO, (c) GWO.

Note that 'supervised unmixing' means that the endmembers contained in the image have been estimated by an endmember extraction algorithm such as vertex component analysis [60].

An example of linear mixing model involving two endmembers is as follows:

$$\mathbf{y}(\lambda) = (1-\lambda)\mathbf{s1} + \lambda\mathbf{s2} + \mathbf{n} \tag{17}$$

where λ is a mixing coefficient, and **s1** and **s2** both $\in \mathbb{R}^{I_3}$ are the two endmembers. **n** is a noise vector, which follows a zero-mean Gaussian distribution.

The unmixing issue in the case of a linear mixing model is well-known and tackled by non-negative matrix factorization [61] in both supervised and unsupervised cases. We wish to exemplify the ability of our mixed GWO optimization method with non-linear mixing models. We assume that the endmember spectra are known, and we wish to determine, among two possible types of nonlinear mixing models, which one is the more suitable for the data.

The considered model for spectral mixture is denoted by $\mathbf{y}(f, \lambda_1, \lambda_2)$, where f is the mixing model, either f_0 or f_1, successively defined in the following.

The first model is the polynomial post-nonlinear mixing model [62]:

$$\mathbf{y}(f_0^{mix}, \lambda_1, \lambda_2) = f_0^{mix}(\lambda_1, \lambda_2) = \mathbf{g^{mix}}(\mathbf{s}(\lambda_1), \lambda_2) + \mathbf{n} \tag{18}$$

where $\mathbf{s} = \left[s_1, \ldots, s_{I_3}\right]^T$ follows a linear mixing model of the two endmembers **s1** and **s2**:

$$\mathbf{s}(\lambda_1) = (1-\lambda_1)\mathbf{s1} + \lambda_1\mathbf{s2} \tag{19}$$

and $\mathbf{g^{mix}}$ is a second-order polynomial non linearity:

$$\mathbf{g^{mix}} : [0;\,1]^{I_3} \rightarrow \mathbb{R}^{I_3}$$

$$\mathbf{s} \mapsto \left[s_1 + \lambda_2 s_1^2, \ldots, s_{I_3} + \lambda_2 s_{I_3}^2\right]^T \tag{20}$$

The second model is the generalized bilinear model [63]:

$$\begin{aligned}\mathbf{y}(f_1^{mix}, \lambda_1, \lambda_2) &= f_1^{mix}(\lambda_1, \lambda_2)\\ &= (1-\lambda_1-\lambda_2)\mathbf{s1} + \lambda_1\mathbf{s2} + \lambda_2\mathbf{s1s2} + \mathbf{n}\end{aligned} \tag{21}$$

In Eqs. (18) and (21), λ_1 and λ_2 are in [0; 1]. Moreover, **n** is an additive independent and identically distributed zero-mean Gaussian noise sequence. It is important to notice that both models reduce to a linear model if $\lambda_2 = 0$, such that they may be similar, particularly for small values of λ_2. In the polynomial post-nonlinear mixing model of Eq. (18), the non-linear terms come from second-order reflections. This model is also a simplified but reliable model for many types of non-linearities [62]. In the generalized bilinear model of Eq. (21), the non-linear terms come from multiple scattering of photons between the two components **s1** and **s2** [63].

We propose, for the first time in this paper, to apply jointly the denoising and the supervised unmixing process to a multispectral

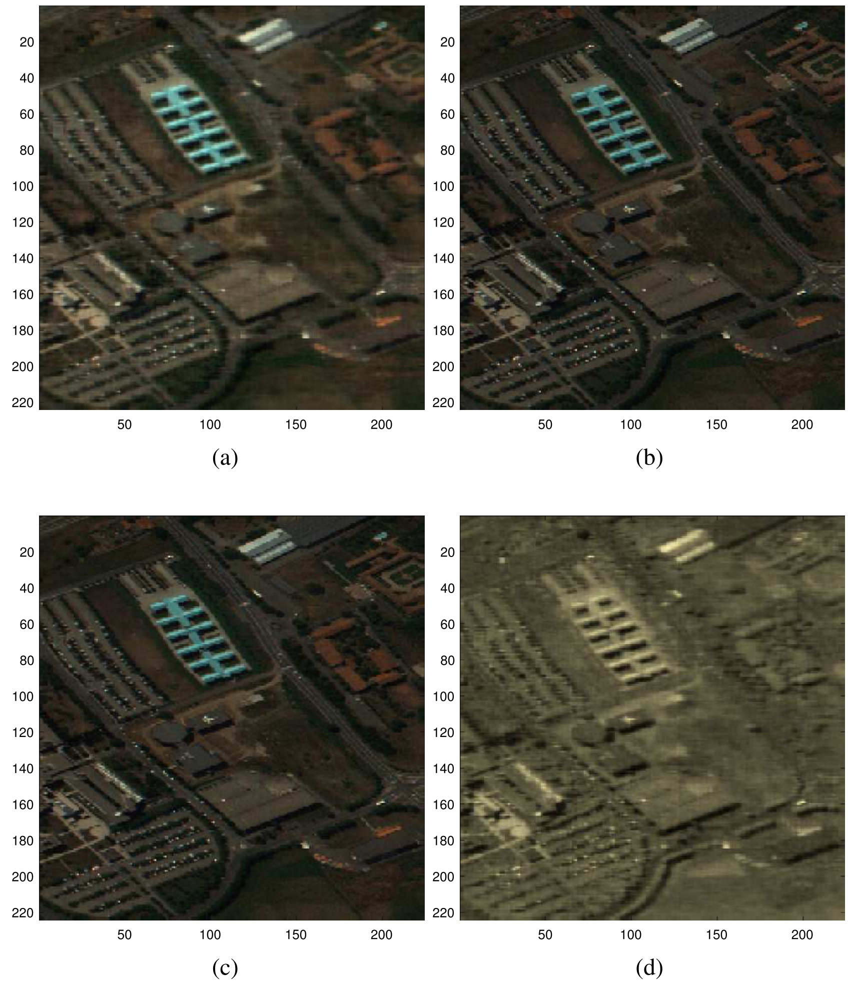


Fig. 10. PaviaU 256 × 256 × 103. Denoised images obtained with (a) ABC, (b) TSA, (c) GA, (d) SA.

image: the idea underneath is that the images could be specifically denoised, in such a manner that the best possible unmixing results are attained. For this, it is of great importance to choose adequately a criterion to minimize.

5.2.2. Proposed criterion

With our amixedGWO method, we propose to minimize the following criterion:

$$\begin{aligned} J^{LS}&(K_1, K_2, K_3, f, \lambda_1, \lambda_2) \\ &= \frac{1}{I_1 I_2 I_3}\|\mathcal{X}_1 - \hat{\mathcal{X}}(K_1, K_2, K_3)\|^2 \\ &\quad + \frac{1}{I_3}\|\mathbf{y}(f^{mix}, \lambda_1, \lambda_2) - \hat{\mathbf{y}}(K_1, K_2, K_3)\|^2 \end{aligned} \tag{22}$$

where tensor $\mathcal{X}_1$ is a gross estimate of $\mathcal{X}$, and $\hat{\mathcal{X}}(K_1, K_2, K_3)$ is the estimate provided by Multiway Wiener Filtering applied to $\mathcal{R}$ with rank values K_1, K_2, K_3. Vector $\mathbf{y}(f, \lambda_1, \lambda_2)$ is the spectrum model, where f, λ_1, and λ_2 should be estimated, and vector $\hat{\mathbf{y}}(K_1, K_2, K_3)$ is a spectrum, whose location is known, extracted from the tensor estimate $\hat{\mathcal{X}}(K_1, K_2, K_3)$. Generally, an anomaly detector such as RX [64] locates some spectra of interest before they can be unmixed.

5.3. Data description and evaluation criteria

In this subsection, we present the experimental conditions which, unless specified, are common to the whole Section 5.

Hyperspectral and multispectral images are now currently used in remote sensing applications. In this context, devoted sensors have been developed, such as AVIRIS sensor (Airborne Visible/Infrared Imaging Spectrometer) [65], or ROSIS sensor (Reflective Optics System Imaging Spectrometer) [66]. A multispectral image can be obtained by selecting some important bands from the hyperspectral images obtained by these airborne sensors. Most of the multispectral aerial images are impaired by noise [58,67] from solar radiation, or atmospheric scattering [68] for instance.

5.3.1. Image setup

We evaluate the performances of the proposed methods on multispectral images. These images are extracted from the PaviaU scene. These data were collected by the ROSIS sensor over the urban area of Pavia University [66]. The spatial size of this image is 610 × 340 pixels. The number of spectral bands is 103 in the wavelength range from about 420 to 850 nm.

In the experiments, the values of the expected image $\mathcal{X}$ and spectral mixture $\mathbf{y}$ are normalized between 0 and 1. The spectral mixture is generated with the reflectance of vegetation for **s1** and the reflectance of soil for **s2**. The resulting spectrum is placed at a specific location in the image, this location being known to extract the spectrum from the denoised image. The expected mixing parameters are set to: $f^{mix} = f_0^{mix} = 0, \lambda_1 = 0.15, \lambda_2 = 0.41$.

The denoising results obtained are evaluated in terms of *SNR*. We remind that $SNR = 10\, log_{10}(\frac{||\mathcal{X}||^2}{||\mathcal{X}-\hat{\mathcal{X}}||^2})$.

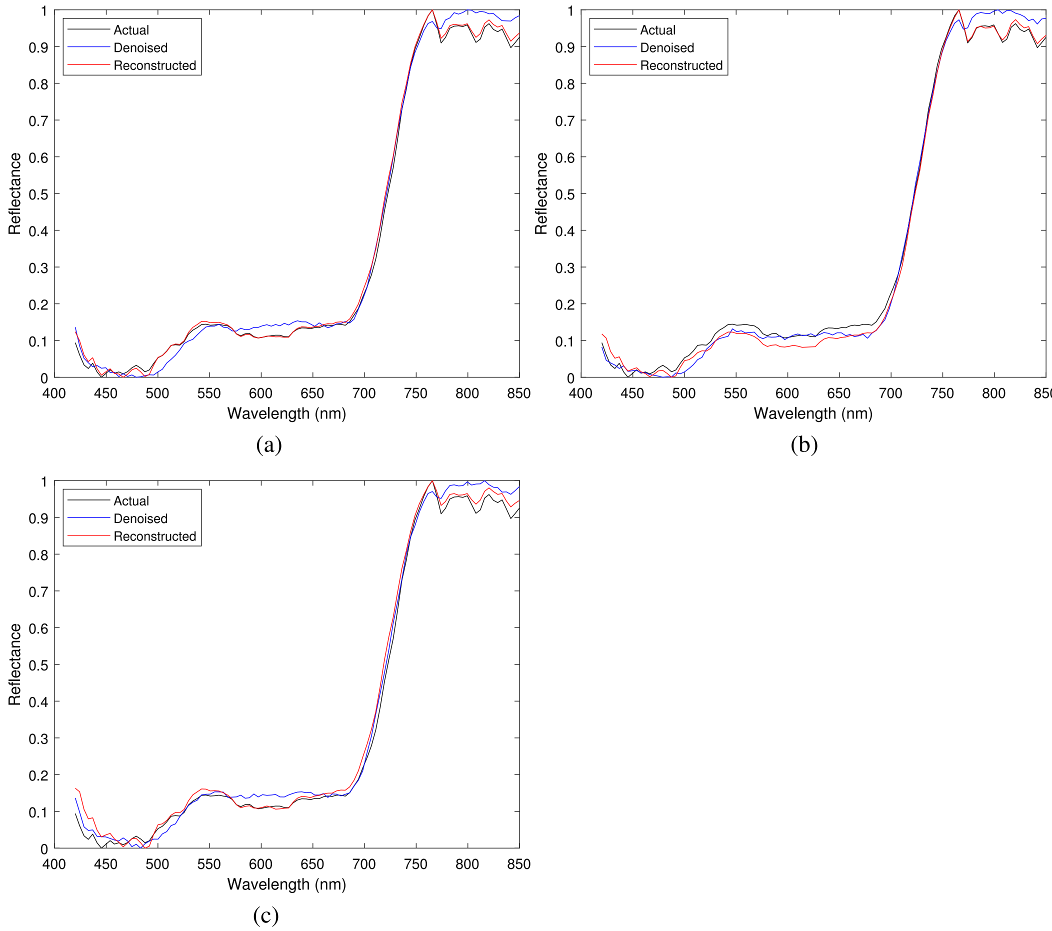


Fig. 11. PaviaU 256 × 256 × 103. Actual, denoised, and reconstructed spectra obtained with: (a) amixedGWO, (b) PSO, (c) GWO.

The images are artificially impaired with white, identically distributed random noise. The input SNR is denoted by SNR_{in} and the value for each experiment is presented in dB. The results are evaluated in terms of output SNR denoted by SNR_{out}. The unmixing results are evaluated in terms of reconstruction error RE between $\mathbf{y}$ and $\hat{\mathbf{y}}$: $RE = \sqrt{\frac{1}{I_3}\|\mathbf{y} - \hat{\mathbf{y}}\|^2}$.

We present statistical results obtained with images of size 32 × 32 × 4, from $M = 10$ runs, each with a different random noise realization. We present also visual results with images of size 256 × 256 × 103. For the RGB display of the multispectral images, 3 representative bands are selected. They are in the red (690 nm), green (550 nm), and blue (450 nm) wavelength domains respectively. In Section 5, programs were written in *Matlab*®, and executed on a PC running Windows, with a 3 GHz double core and 3 GB RAM.

5.4. Parameter setting for the proposed and comparative optimization methods

We solve a problem of multispectral image denoising and unmixing, involving four discrete parameters and two continuous parameters.

To evaluate the performance of the proposed adaptive mixed GWO (amixedGWO), we compare it with famous meta-heuristic stochastic optimization methods: particle swarm optimization (PSO) [8], grey wolf optimization (GWO) [11], artificial bee colony (ABC) [23], tree seed algorithm (TSA) [24], genetic algorithm (GA) [7], and simulated annealing (SA) [69]. PSO, GWO, ABC, TSA, GA, and SA search continuous spaces. So, when integer values are expected, we round the result they provide.

The tested algorithms require a few parameters which are set once for all processed images: all methods are run with $T_{max} = 20$ iterations. All methods including TSA are run in such a way that the computational load they require when the image size is 32 × 32 × 4 is the same, that is, 1.5 s for all methods except ABC, which requires 2.4 s, and SA which requires 2.0 s. For this, the number of agents is $Q = 12$ for amixedGWO, PSO, GWO, and GA; and $Q = 6$ for TSA and ABC.

We choose this relatively low number of agents and iterations, compared to the experiments in previous sections, because for this application the computational load required to compute the criterion value is much higher, in particular when the image contains more than 100 rows, columns or bands.

We remind that the first three expected parameters are the ranks K_1, K_2, K_3, the fourth is the type of mixing model f^{mix}, the fifth and sixth are the mixing coefficients λ_1 and λ_2. Table A.15 presents the parameters used to define the search spaces for each unknown.

The number H_i of values in the search spaces for the ranks K_i, $i = 1, \ldots, 3$ is either 8, or the size of the image I_i, $i = 1, \ldots, 3$ if $I_i \leq 16$. Note that there are only two possible values for the mixing model f^{mix}, and that the acceptable values for the mixing coefficients λ_1 and λ_2 are between 0 and 1.

For PSO, GWO, ABC, TSA, GA and SA, the search spaces are continuous, with bounds which, $\forall\, i = 1, \ldots, 6$, are the first and last values of $\mathbf{d}_i^{val}$, unless for $i = 4$, for which the lower bound is 0, and the upper bound is 1. For the first four parameters, that is, the rank values and the type of mixing model, all test

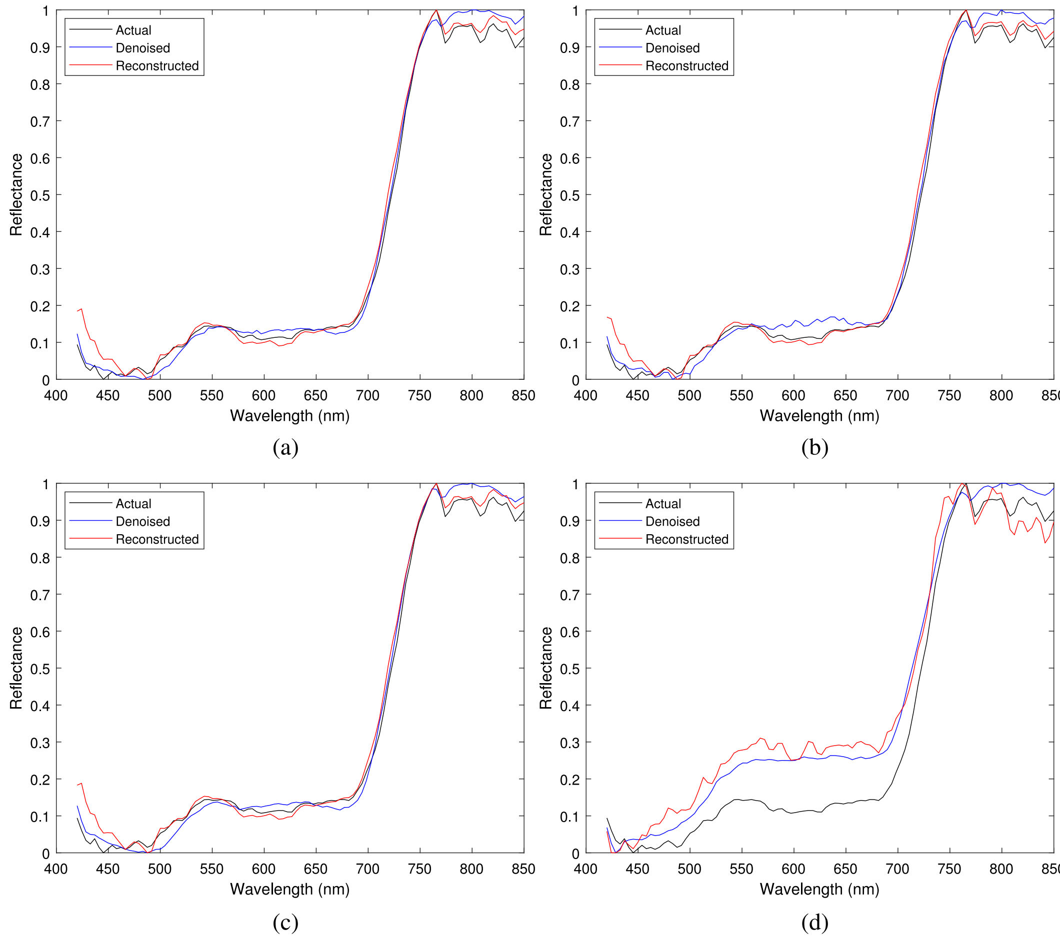


Fig. 12. PaviaU 256 × 256 × 103. Actual, denoised, and reconstructed spectra obtained with:(a) ABC, (b) TSA, (c) GA, (d) SA.

values are rounded when the criterion is evaluated. Each run is performed with a random initialization of the search agents in the optimization algorithms. The ABC method is run with 4 onlooker bees. In PSO, the acceleration constants γ_1 and γ_2 [8] are set to 2 and 3 respectively. The TSA algorithm is run with $ST = 0.9$: a value close to 1 is recommended for lower dimensional optimization problems [24] and TSA should provide a fast convergence.

For GA the crossover probability is 0.9, and the mutation probability is 0.1. For SA we set a number of control points equal to 12, that is, the same value as the one used for Q in the other methods.

5.5. Experimental results

In this subsection we provide results obtained from a multispectral image extracted from PaviaU scene: statistical results are computed on a small image, and visual results are computed on a larger image.

Fig. 4 shows the noise-free multispectral images which are used for the tests; and Fig. 5 shows the two endmembers, and the expected spectrum, obtained with the parameters $f^{mix} = f_0^{mix} = 0$, $\lambda_1 = 0.15, \lambda_2 = 0.41$.

In Section 5.5.1, we propose a school-case study in the ideal situation where $SNR_{in} = \infty$, in two cases, with two different versions of the reference $\mathcal{X}_1$. In this noise-free case we expect the optimization methods to yield a result tensor which is exactly the reference $\mathcal{X}_1$.

In Section 5.5.2, we consider a realistic case: the image $\mathcal{R}$ is impaired with some finite input SNR value, and the reference image is obtained with a Wiener filtering in Fourier domain, which is a basic parameter-free method. This denoising method is applied band-by-band, for each spectral band of the processed image.

5.5.1. Convergence study on a school case

In this subsubsection, the input SNR is $SNR_{in} = \infty$, and the reference tensor $\mathcal{X}_1$ is either the noise-free image or the output of MWF when applied to $\mathcal{X}$ with *a priori* known rank values.

First case: the reference is the noise-free tensor

The reference tensor $\mathcal{X}_1$ is here the noise-free tensor, and the expected rank values are then $K_1 = I_1 = 32$, $K_2 = I_2 = 32$, and $K_3 = I_3 = 4$. Fig. 6 presents the mean convergence plot for this experiment, obtained with $M = 10$ runs. Table A.16 presents statistical results obtained on the expected parameters, and Table A.17 presents the mean reconstruction error RE over the M runs.

The convergence plot in Fig. 6, and the numerical results in Table A.16 show that the proposed amixedGWO, PSO, and GWO behave best in this experiment. For instance, no mean estimated rank value differs from the expected one by more than 1. TSA, ABC, GA and SA underestimate rank values, which may have an influence on the estimation of the mixing parameters. It also can be noticed that the mean estimated values of mixing model f are elevated: we may face a multimodal optimization problem with two relative minima which are very close to each other. Still, our method surpasses the comparative methods in terms of convergence as shown in Fig. 6. Also, we can see from Table A.17 that the reconstruction errors are the smallest when the proposed amixedGWO or the comparative GWO methods are used.

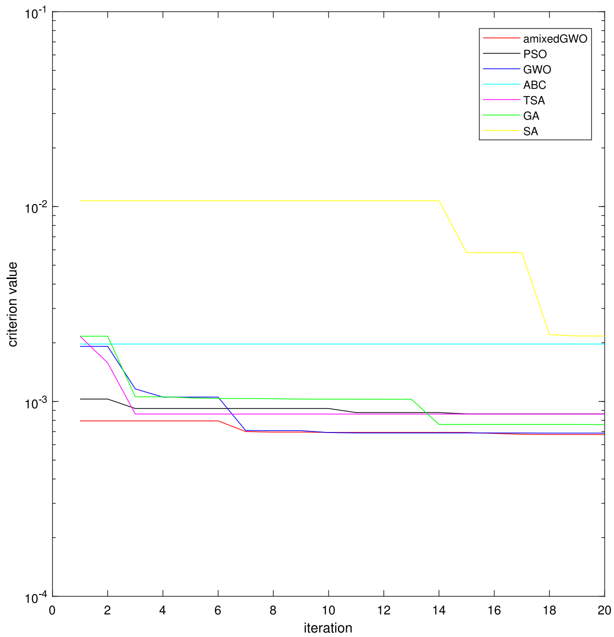


Fig. 13. Mean convergence plot with $SNR_{in} = 10$ dB and Fourier Wiener as reference $\mathcal{X}_1$.

Second case: the reference is the output of MWF

In this paragraph, the input SNR is still infinite, but the expected rank values are less than the image size. Indeed, the reference is the output of MWF applied to the noise-free tensor $\mathcal{X}$ with rank values which are as follows: $K_1 = 0.5I_1 = 16$, $K_2 = 0.5I_2 = 16$, and $K_3 = I_3 = 4$. We expect from the optimization methods that they retrieve these rank values.

Fig. 7 presents the mean convergence plot for this experiment, obtained with $M = 10$ runs. Table A.18 presents statistical results obtained on the expected parameters, and Table A.19 presents the mean reconstruction error RE over the M runs. We notice there exists a bias on the mixing model for all optimization methods, possibly due to two local minima with a very similar score. Indeed the RE values in Table A.19 are similar for amixedGWO, GWO, TSA, and PSO, although, as we can see in Table A.18, PSO yields an estimated value for the mixing model which is significantly different.

In Table A.19, we can see that the reconstruction error is more elevated than in the previous case: through filtering with low rank values, the denoised spectrum is significantly different from the expected spectrum. As the optimization methods aim for the denoised spectrum, the RE value increases. The convergence plots in Fig. 7, and show that our method surpasses the comparative methods in terms of convergence.

5.5.2. *Realistic case*

In this subsection we consider a realistic case where the reference tensor is obtained with a parameter-free, simple method. This method is a Wiener filtering process applied band-by-band in Fourier domain. In this realistic case, the input SNR is less than ∞. We exemplify our method on a large image, with five different values of input SNR: $SNR_{in} = 0, 5, 10, 15$, and 20 dB. For $SNR_{in} = 10$ dB we provide visual results for one run (see Figs. 8, 9, and 10), as well as the mean convergence curves obtained on $M = 3$ runs (in Fig. 13).

Tables A.20–A.22 present respectively the output SNR, the reconstruction error values and the global best for all input SNR values. The overall rank for amixedGWO is 1 in all of these aspects, though the output *SNR* is smaller than for at least one comparative method, on input SNR values 5 and 15 dB. This means that, overall, amixedGWO may not yield the best rank values in terms of the sole denoising. However, a good behaviour in terms of exploitation can explain the values of reconstruction error, which are the smallest for all input SNR value except 0 dB (see Table A.21). Table A.22 shows that the global best values are the smallest for amixedGWO except on input SNR values 5 and 15 dB. In these cases amixedGWO may yield a denoised spectrum which is the closest to the model spectrum, but at the expense of the first term of the criterion in Eq. (22).

From these results we can infer the following comments: thanks to the improved discrete optimization process used to estimate the rank values, the proposed amixedGWO seems to converge quickly towards rank values which minimize both terms of the criterion in Eq. (22), and has time to refine the estimation of the continuous parameters (the mixing coefficients). A compromise must be found in some cases: the spectrum of interest in the denoised image may be the closest possible to the model spectrum, and not to the corresponding spectra in the reference image.

Here are details about one run, which yielded the images in Figs. 9 and 10, and the spectra in Figs. 11 and 12:

In this case, all methods except ABC and SA yield approximately the same global best after convergence: 6.75 10^{-4} for amixedGWO, 8.63 10^{-4} for PSO, 6.87 10^{-4} for GWO, 1.97 10^{-3} for ABC, 8.61 10^{-4} for TSA, 7.60 10^{-4} for GA, 2.16 10^{-3} for SA.

The output SNR for the reference image is 12.79 dB. The output SNR values (in dB) are respectively 18.35 for amixedGWO, 14.88 for PSO, 17.83 for GWO, 8.64 for ABC, 16.26 for TSA, 18.91 for GA, and 8.54 for SA. So the denoised image with the best output SNR value is provided by GA, but as can be seen in Figs. 9 and 10, a significant difference only exists between the result obtained by ABC and SA and the other methods, but the other images are very similar.

The reconstruction error values are respectively 1.20 10^{-3} for amixedGWO, 1.94 10^{-3} for PSO, 2.47 10^{-3} for GWO, 2.92 10^{-3} for ABC, and 2.69 10^{-3} for TSA, 2.90 10^{-3} for GA, and 1.08 10^{-2} for SA.

The RE value obtained with GWO or TSA is higher than for amixedGWO, because one the one hand the RE is computed between the actual spectrum and the reconstructed one; on the other hand the criterion which is minimized involves the difference between the denoised spectrum and the reconstructed spectrum.

This is confirmed by the spectra displayed in Fig. 11: when amixedGWO is used, the reconstructed spectrum (in red) is the closest to the actual spectrum (in black), because the denoised spectrum (in blue) is the closest to the actual spectrum. Somehow, in this case, the balance between denoising and unmixing is better when amixedGWO is used. Consequently, the denoised spectrum is the closest possible to the model. As concerns the estimated parameters, we could notice that the spatial ranks are elevated (between 255 or 256 for all methods except ABC and SA), and that the spectral rank is small (between 60 and 97 for all methods except ABC and SA); it can also be noticed that TSA and GA provided f_1 as spectrum model whereas f_0 was expected, yielding though small RE values. From this we infer that there may exist at least two close relative minima in the minimized criterion. That is, we faced a multimodal optimization problem.

6. Discussion

We have evaluated the behaviour of the proposed mixed GWO and its adaptive version in the cases where the search spaces are either continuous, discrete or mixed. From the results obtained in Section 4, it can be concluded that the proposed methods mixedGWO and amixedGWO exhibit a good behaviour, with respect to the algorithms of the GWO family (the original GWO and mGWO), on the continuous case, reaching almost the same performance as the state-of-the-art methods in most of the functions of a first benchmark [11], and outperforming them on the fixed dimension multi-modal benchmark functions.

The proposed amixedGWO has also been compared to the state-of-the-art methods GWO, PSO, ABC, TSA, GA and SA using a second benchmark, namely the CEC2014 functions. The obtained results show the limits of the proposed method on strictly continuous unimodal and multi-modal functions of the CEC2014 benchmark. Indeed, the proposed method is outperformed by the state-of-the-art methods, at the exception of GWO, and especially by TSA. However, the amixedGWO is at least twice faster than the comparative methods.

On the discrete case, our mixedGWO and amixedGWO outperform the comparative MODGWO for all the functions we have tested. Our mixedGWO and amixedGWO methods seem to be applicable to a wider range of functions, and thereby possibly to a wider range of concrete problems, which makes them promising algorithms. In the mixed case, we have reached promising results with mixedGWO, which are even better with amixedGWO. In Section 5, we have considered an application to denoising and unmixing of multispectral images. This is a typical case where some of the parameters, the rank values and the type of mixing model, take their values in discrete search spaces, and some other parameters, the mixing coefficients, on continuous search spaces. We infer from the results obtained that the criterion we have chosen is a multi-modal function of the parameters, because there exists a bias on the estimated mixing model, for some of the tested algorithms and not for the others, with a small difference on the global best value which is reached. The limitation of the proposed strategy relies on the choice of the reference tensor: the convergence performance of amixedGWO is good, but the criterion is minimized with respect to a reference which may not be reliable.

7. Conclusion

In this paper, a short review of existing meta-heuristic, and particularly bio-inspired methods such as PSO, ABC, TSA, GA, SA and GWO has been firstly made. A novel method based on GWO which is able to tackle continuous problems as well as discrete or mixed problems has been proposed and named mixed GWO. This method and its adaptive variant have been compared to the already existing versions of the GWO algorithm on 20 continuous benchmark functions. From these comparison, it has been proven that the amixedGWO is able to tackle correctly continuous problems with performances comparable to the other versions of GWO, and outperforming them in the resolution of multi-modal functions. Though outperformed by TSA in continuous functions such as the CEC2014, the proposed amixedGWO behaves correctly, and exhibits a smaller computational load.

The proposed methods have also been tested on discrete benchmark functions and compared to the only discrete version of GWO which is not restricted to binary problems, the MODGWO. In this case, the amixedGWO method clearly outperforms the MODGWO method and finds the optimal minimum, with a null standard deviation, in 5 out of the 6 discrete benchmark functions. The amixedGWO method has then been tested on mixed benchmark functions. In these cases, the optimal minimum has been found in 4 out of the 6 benchmark functions, while the amixedGWO has correctly converged on all of the functions. These tests run on several benchmark functions show that the proposed amixedGWO method is able to tackle correctly various kinds of problems, disregarding the shape of their search spaces, whether they are continuous, discrete, or mixed.

The robustness of the proposed approach has been tested on a real-world application: for the first time, simultaneous denoising and unmixing of multispectral images has been performed with a bio-inspired optimization method. The goal of this application being to find the best balance between the quality of the denoising and the quality of the spectrum unmixing.

However, this application requires a reference image, involved in the criterion which is minimized. Therefore, it could be interesting to implement an iterative process, taking as an input reference the image obtained from the amixedGWO method at each iteration, in order to compensate the potential problem of a bad reference. However, this study in out of the scope of the paper, which aimed at studying the comparative performances of bio-inspired optimization methods on simultaneous denoising and unimixing of multispectral images.

In the future, it could be interesting to study the comparative performances of various versions of GWO when the number of parameters to estimate is changing, and, for the discrete version of GWO, when the number of values in the search spaces is changing. An adaptive search space combined with the graph theory could be a solution. Moreover, it seems interesting to study the performance of the proposed amixedGWO algorithm on other applications, and to create its multi-objective version. Finally, concerning bio-inspired optimization in general, it could be interesting to create a discrete version of the comparative methods we have used such as ABC and TSA, which is not restricted to binary, and getting inspired by the formalism we propose for our mixed GWO.

Acknowledgements

The authors would like to express their thanks to the anonymous reviewers for their careful reading and their helpful remarks, which have contributed in improving the clarity of the paper.

This work has been partially financed by The firm IntuiSense Technologies SAS and the ANRT through the Grant "CIFRE convention 2015/0833".

Appendix

See Figs. A.14 and A.15 and Tables A.1–A.22.

Table A.1
Updated index values for cases illustrated in Fig. 2.

Case	Wolf update					
	h^l	Δ	$sgn(h^l - h(iter))$	$h(iter)$	$h(iter+1)$	$x(iter+1)$
2(a)	6	1	+1	4	$(4+1)\ mod\ 7 = 5$	58
2(b)	6	4	+1	4	$(4+4)\ mod\ 7 = 1$	11
2(c)	2	1	−1	4	$(4-1)\ mod\ 7 = 3$	29

Table A.2
Unimodal benchmark functions.

Function	Dim	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[−100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	[−10, 10]	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	30	[−100, 100]	0
$F_4(x) = max\{\lvert x_i\rvert,\ 1 \le i \le n\}$	30	[−100, 100]	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	30	[−30, 30]	0
$F_6(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	[−100, 100]	0
$F_7(x) = \sum_{i=1}^{n} ix_i^4 + random(0, 1)$	30	[−1.28, 1.28]	0

Algorithm 5 Pseudo-code: Particle Swarm Optimization for multiple parameter estimation

Inputs: fitness function, small factor ϵ set by the user, to stop the algorithm, lower and upper bounds for each parameter.

1. Set iteration number iter $=1$, create an initial population composed of Q random particles with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate the fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each particle $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Update the local best particles $\mathbf{p}_q^{\mathbf{k}}$ ($q = 1, \ldots, Q$), and the global best particle $\mathbf{g}^{\mathbf{k}}$,
4. Repeat steps for each particle q, $q = 1, \ldots, Q$:
   (a) Compute displacement also called velocity $\mathbf{v}_q^{\mathbf{k}}(\text{iter}+1)$
   (b) Compute position $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$
5. Exchange the current population with the new one, obtained at step 4.
6. If iter $<$ T$_{\max}$ or $||\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1) - \mathbf{x}_q^{\mathbf{k}}(\text{iter})|| > \epsilon$, increase iter, and go to step 2.

Output: estimated parameter values $\hat{K}_1, \hat{K}_2, \ldots, \hat{K}_N$ contained in $\mathbf{g}^{\mathbf{k}}$.

Algorithm 6 Pseudo-code: Grey Wolf Optimization for multiple parameter estimation

Inputs: fitness function, maximum number of iterations T$_{\max}$, small factor ϵ set by the user, to stop the algorithm, lower and upper bounds for each parameter.

1. Set iteration number iter $=1$, create an initial herd composed of Q wolves with all required parameter values $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$. This initial population takes the form of a matrix with Q rows and N columns.
2. Evaluate fitness function value $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter}))$ of each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$.
3. Sort the wolves through their fitness value and update the α, β, and δ wolves which hold respectively the first, second and third best fitness value. Store their position in vectors $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, and $\mathbf{x}_\delta^{\mathbf{k}}$ respectively.
4. Repeat steps for each wolf $\mathbf{x}_q^{\mathbf{k}}(\text{iter})$, $q = 1, \ldots, Q$:
   (a) Compute the contributions $\mathbf{y}^\alpha$, $\mathbf{y}^\beta$, and $\mathbf{y}^\delta$ of wolves α, β, and δ respectively to the displacement of the q^{th} wolf.
   (b) Compute the updated position $\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1)$ of the q^{th} wolf:
   $$\mathbf{x}_q^{\mathbf{k}}(\text{iter}+1) = \frac{1}{3}(\mathbf{y}^\alpha + \mathbf{y}^\beta + \mathbf{y}^\delta) \quad \text{(A.1)}$$
5. Exchange the current population with the new one.
6. If iter $<$ T$_{\max}$ or $f(\mathbf{x}_q^{\mathbf{k}}(\text{iter})) > \epsilon$, increase iter, and go to step 2.

Output: estimated parameter values $\hat{K}_1, \hat{K}_2, \ldots, \hat{K}_N$ contained in $\mathbf{x}_\alpha^{\mathbf{k}}$.

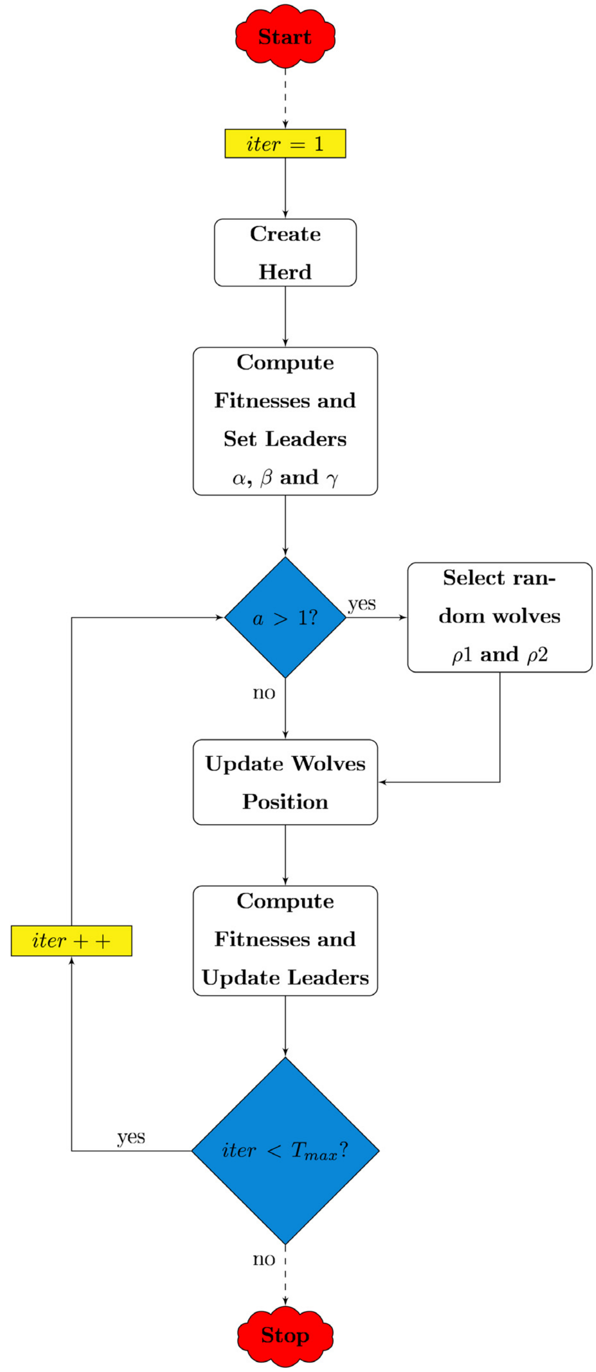


Fig. A.14. The mixedGWO's main flowchart.

Algorithm 7 Update rules for the Multi-Objective Discrete Grey Wolf Optimizer

K_1^α, K_1^β, K_1^δ, K_2^α, K_2^β, K_2^δ hold for the first and second components respectively of the leader vectors $\mathbf{x}_\alpha^{\mathbf{k}}$, $\mathbf{x}_\beta^{\mathbf{k}}$, $\mathbf{x}_\delta^{\mathbf{k}}$. Let *rand* be a random real number between 0 and 1.

$$K_1 = \begin{cases} K_1^\alpha & \text{if} \quad rand \le \frac{1}{3} \\ K_1^\beta & \text{if} \quad rand > \frac{1}{3} \quad \text{and} \quad rand \le \frac{2}{3} \\ K_1^\delta & \text{if} \quad rand > \frac{2}{3} \quad \text{and} \quad rand \le 1 \end{cases} \quad \text{(A.2)}$$

$$K_2 = \begin{cases} K_2^\alpha,\ K_2^\beta,\ \text{or}\ K_2^\delta & \text{if} \quad rand \le a \\ K_2^{'} & \text{if} \quad rand > a \quad \text{and} \quad rand \le 1 \end{cases} \quad \text{(A.3)}$$

Table A.3
Multi-modal benchmark functions.

Function	Dim	Range	f_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	[−500, 500]	$-418.9829 \times n$
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	[−5.12, 5.12]	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^n}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[−32, 32]	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600, 600]	0

Table A.4
Fixed dimension multi-modal benchmark functions.

Function	Dim	Range	f_{min}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}\left(x_i - a_{ij}\right)^6}\right)^{-1}$	2	[−65, 65]	1
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x1\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5, 5]	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + x_2^4$	2	[−5, 5]	−1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[−5, 5]	0.397887
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \left[30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	[−2, 2]	3
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6	[0, 1]	−3.32
$F_{21}(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	−10.1532
$F_{22}(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	−10.4028
$F_{23}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	−10.5363

Table A.5
Results of unimodal continuous benchmark functions with $T_{max} = 3000$.

F		GWO	mGWO	mixedGWO	amixedGWO
F_1	*Avg.*	$1.08e-205$	$\mathbf{3.03e-263}$	$1.17e-177$	$2.50e-115$
	Std.	0	0	0	$6.53e-115$
	Rank	2	1	3	4
F_2	*Avg.*	$1.15e-118$	$\mathbf{1.30e-152}$	$2.80e-104$	$6.55e-66$
	Std.	$3.62e-118$	$2.79e-152$	$7.11e-104$	$6.78e-66$
	Rank	2	1	3	4
F_3	*Avg.*	$6.23e-41$	$\mathbf{8.60e-53}$	$2.83e-41$	$4.24e-31$
	Std.	$3.24e-40$	$4.69e-52$	$1.52e-40$	$2.31e-30$
	Rank	3	1	2	4
F_4	*Avg.*	$1.69e-40$	$\mathbf{2.97e-58}$	$3.38e-38$	$3.17e-22$
	Std.	$7.18e-40$	$7.15e-58$	$8.43e-38$	$5.92e-22$
	Rank	2	1	3	4
F_5	*Avg.*	**26.336**	26.375	27.248	26.556
	Std.	0.843	0.743	0.959	0.674
	Rank	1	2	4	3
F_6	*Avg.*	**0.487**	0.495	1.916	0.834
	Std.	0.282	0.238	0.524	0.382
	Rank	1	2	4	3
F_7	*Avg.*	$3.06e-04$	$\mathbf{2.09e-04}$	$3.10e-04$	$2.27e-03$
	Std.	$1.8e-04$	$1.33e-04$	$1.75e-04$	$8.78e-04$
	Rank	2	1	3	4
Average rank		1.86	1.29	3.43	3.72
Overall rank		2	1	3	4

Table A.6
Results of multi-modal continuous benchmark functions with $T_{max} = 3000$.

F		GWO	mGWO	mixedGWO	amixedGWO
F_8	*Avg.*	−6311.745	**−6378.536**	−5970.654	−5783.265
	Std.	1053.778	603.356	835.292	555.451
	Rank	2	1	3	4
F_9	*Avg.*	**0**	**0**	0	0
	Std.	0	0	0	0
	Rank	1	1	1	1
F_{10}	*Avg.*	$5.77e-15$	$\mathbf{4.47e-15}$	$6.84e-15$	$7.31e-15$
	Std.	$1.81e-15$	$1.23e-15$	$1.45e-05$	$9.01e-16$
	Rank	2	1	3	4
F_{11}	*Avg.*	$1.12e-03$	**0**	$4.51e-03$	$7.01e-03$
	Std.	$4.58e-03$	0	$8.52e-03$	$7.95e-03$
	Rank	2	1	3	4
Average rank		1.75	1	2.5	3.25
Overall rank		2	1	3	4

Table A.7
Results of fixed dimension multi-modal continuous benchmark functions $T_{max} = 3000$.

F		GWO	mGWO	mixedGWO	amixedGWO
F_{14}	*Avg.*	4.1333240	2.9999936	4.9999841	**2.8362586**
	Std.	4.6068296	3.8506485	5.0854562	3.6191509
	Rank	3	2	4	1
F_{15}	*Avg.*	$3.09e-03$	$3.68e-03$	$4.40e-03$	$\mathbf{9.90e-04}$
	Std.	$6.90e-03$	$7.59e-03$	$8.12e-03$	$3.66e-03$
	Rank	2	3	4	1
F_{16}	*Avg.*	−1.03162845222	−1.03162845227	−1.03162845241	**−1.03162845331**
	Std.	$3.19e-10$	$4.82e-10$	$2.02e-09$	$2.04e-09$
	Rank	4	3	2	1
F_{17}	*Avg.*	0.3979427	0.3979661	0.3979500	**0.3979117**
	Std.	$6.52e-05$	$8.18e-05$	$5.81e-05$	$2.59e-05$
	Rank	2	4	3	1
F_{18}	*Avg.*	5.7000011	**3.0000007**	**3.0000007**	3.0000010
	Std.	14.7885089	$9.36e-07$	$7.40e-07$	$1.02e-06$
	Rank	4	1	1	3
F_{20}	*Avg.*	−3.2563368	−3.2635423	**−3.2980638**	−3.2780500
	Std.	0.0635625	0.0645893	0.0486790	0.0600625
	Rank	4	3	1	2
F_{21}	*Avg.*	−9.8163635	−9.2886724	−9.9847094	**−9.9847752**
	Std.	1.2818425	1.9692828	0.9224400	0.9224418
	Rank	3	4	2	1
F_{22}	*Avg.*	−10.2257553	−10.2257344	−10.4028815	**−10.4029327**
	Std.	0.9704291	0.9704252	$4.02e-05$	$5.98e-06$
	Rank	3	4	2	1
F_{23}	*Avg.*	−10.1758610	−10.0856251	−10.5363495	**−10.5364004**
	Std.	1.3720325	1.7518213	$6.00e-05$	$7.40e-06$
	Rank	3	4	2	1
Average rank		3.11	3.11	2.33	1.33
Overall rank		3	3	2	1

Table A.8
Results of fixed dimension multi-modal continuous benchmark functions with $Q = 48$ and $T_{max} = 20$.

F		GWO	mGWO	mixedGWO	amixedGWO
F_{14}	*Avg.*	6.6279133	6.0040172	5.4039278	**4.8373139**
	Std.	4.4045817	4.4292795	4.9401453	3.9127446
	Rank	4	3	2	1
F_{15}	*Avg.*	$6.33e-03$	$5.16e-03$	$2.47e-03$	$\mathbf{2.12e-03}$
	Std.	$8.84e-03$	$8.12e-03$	$5.41e-03$	$4.43e-03$
	Rank	4	3	2	1
F_{16}	*Avg.*	$-\mathbf{1.0316125}$	−1.0315843	−1.0316106	−1.0314833
	Std.	$7.11e-05$	$2.04e-04$	$1.80e-05$	$4.58e-04$
	Rank	1	3	2	4
F_{17}	*Avg.*	0.4028292	0.4061008	0.4059269	**0.4015930**
	Std.	$6.13e-03$	$6.52e-03$	$7.78e-03$	$3.89e-03$
	Rank	2	4	3	1
F_{18}	*Avg.*	3.0068726	3.0047267	3.0038818	**3.0036725**
	Std.	0.0112383	$7.33e-03$	$5.53e-03$	0.0107389
	Rank	4	3	2	1
F_{20}	*Avg.*	−3.2051662	−3.2277292	−3.2343476	$-\mathbf{3.2707007}$
	Std.	0.1354030	0.09749126	0.0816793	0.0732078
	Rank	4	3	2	1
F_{21}	*Avg.*	−7.4236776	−6.7810457	−7.0626588	$-\mathbf{7.7753623}$
	Std.	3.4666437	3.4520468	3.2961559	3.1887887
	Rank	2	4	3	1
F_{22}	*Avg.*	−7.3167512	−8.7080669	−7.3722878	$-\mathbf{9.2020850}$
	Std.	3.5471189	2.5870188	2.7819184	2.3578537
	Rank	4	2	3	1
F_{23}	*Avg.*	−8.8736678	−7.4909987	−8.9657624	$-\mathbf{9.0310124}$
	Std.	2.8332950	3.4994928	1.7369446	2.8311698
	Rank	3	4	2	1
Average rank		3.11	3.22	2.33	1.33
Overall rank		3	4	2	1

Table A.9
Results of the Wilcoxon test on the fixed-dimension multi-modal functions in continuous search space.

Function	mixedGWO vs. GWO	mixedGWO vs. mGWO	amixedGWO vs. GWO	amixedGWO vs. mGWO
F_{14}	0.057 (=)	0.277 (=)	0.154 (=)	0.631 (=)
F_{15}	0.936 (=)	0.612 (=)	0.485 (=)	0.467 (=)
F_{16}	$1.94e-08$ (+)	$4.11e-03$ (+)	$4.20e-11$ (+)	$9.04e-07$ (+)
F_{17}	0.046 (−)	0.644 (=)	0.0345 (+)	0.406 (=)
F_{18}	0.45 (=)	0.096 (=)	0.0120 (+)	$2.41e-05$ (−)
F_{20}	0.21 (=)	0.959 (=)	0.164 (=)	0.020 (+)
F_{21}	$3.06e-08$ (+)	0.018 (+)	$3.13e-09$ (+)	$3.06e-08$ (+)
F_{22}	$1.43e-09$ (+)	$2.34e-03$ (+)	$2.48e-12$ (+)	$1.43e-09$ (+)
F_{23}	$2.16e-11$ (+)	0.0243 (+)	$4.84e-13$ (+)	$3.14e-07$ (+)
+/=/−	4/4/1	4/5/0	6/3/0	5/3/1

Table A.10
Average residual errors on the CEC2014 functions.

Function	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
F_1	3070.2	462.8	1534.9	434.4	**45.6**	152.4	462.6
F_2	396.6	514.7	1126.2	63	**25.5**	508.5	1033.2
F_3	542.5681	342.5489	684.1074	96.8305	**21.4919**	162.4	3631.7
F_4	$1.00e-04$	**0**	$4.6e-03$	**0**	**0**	$1.14e-13$	$3.41e-13$
F_5	$9.6e-03$	**0**	$6.84e-01$	**0**	**0**	$1.7e-13$	$5.96e-11$
F_6	$3.8e-03$	**0**	$8.2e-03$	**0**	**0**	$2.27e-13$	$1.36e-12$
F_7	$4.2e-03$	$4.8e-03$	$9.5e-03$	$3.5e-03$	**0**	$8.4e-03$	$4.4e-03$
F_8	$1.33e-01$	**0**	$7.18e-02$	**0**	**0**	$2.27e-13$	$1.48e-12$
F_9	$1.99e-01$	**0**	$9.96e-02$	**0**	**0**	$3.41e-13$	$2.27e-12$
F_{10}	49.2603	$1.69e-01$	30.30	$1.02e-01$	**0**	$6.24e-3$	$5.32e-07$
F_{11}	4.0561	$2.39e-01$	4.0174	$9.89e-02$	$\mathbf{4.8e-02}$	3.5807	10.0860
F_{12}	$4.96e-01$	$1.12e-01$	$3.73e-01$	$4.49e-01$	$9.39e-02$	$\mathbf{5.12e-13}$	$1.39e-10$
F_{13}	$3.38e-02$	$3.81e-02$	$4.18e-02$	$4.78e-02$	$\mathbf{1.41e-02}$	$2.75e-02$	$6.25e-02$
F_{14}	$7.5e-03$	$1.32e-02$	$1.14e-02$	$2.49e-02$	$\mathbf{3.5e-03}$	$1.67e-02$	$3.94e-02$
F_{15}	$2.24e-02$	**0**	$1.04e-02$	$2.00e-04$	**0**	$1.38e-02$	$1.38e-02$
F_{16}	$6.9e-03$	$1.09e-02$	$1.10e-02$	$1.15e-02$	**0**	$1.94e-02$	$1.94e-02$
F_{23}	99.0327	33.6887	112.9178	39.3915	**7.56**	160.4773	79.6581
F_{24}	80.9750	89.8178	69.7263	57.1990	**19.6872**	100.6363	59.5095
F_{25}	$1.60e-02$	$1e-04$	$4.23e-02$	$4.2e-03$	**0**	47.9145	$1.34e-05$
F_{26}	$2.47e-01$	$3.81e-01$	$1.53e-01$	$1.09e-01$	**0**	1.1451	12.8945
F_{27}	$2.08e-01$	$5.37e-02$	$4.18e-01$	$\mathbf{1.02e-01}$	$5.5e-03$	$5.96e-01$	$2.85e-01$
F_{28}	260.5198	28.8892	297.3808	107.5971	**9.4956**	176.3242	176.3536

Table A.11
Results of unimodal benchmark functions in discrete search space with $T_{max} = 3000$.

F	Step	MODGWO			mixedGWO			amixedGWO		
		Avg.	Std.	Med.	Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	8.100	5.880	6	**0**	**0**	**0**	**0**	**0**	**0**
F_2	0.5	0.183	0.278	**0**	**0**	**0**	**0**	**0**	**0**	**0**
F_3	1	7.100	4.428	5	0.067	0.254	**0**	**0**	**0**	**0**
F_4	1	2	0.695	2	0.067	0.254	**0**	**0**	**0**	**0**
F_5	0.5	23.217	17.849	18	4.650	8.394	**0**	**2.583**	**5.173**	**0**
F_6	0.5	8.692	6.060	7.5	**0**	**0**	**0**	**0**	**0**	**0**

Table A.12
Results of MODGWO on unimodal benchmark functions in discrete search space with $T_{max} = 15{,}000$ and $T_{max} = 30{,}000$.

F	Step	$T_{max} = 15{,}000$			$T_{max} = 30{,}000$		
		Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	2.73	1.64	2	1.20	0.98	1
F_2	0.5	0.08	0.19	0	0.10	0.20	0
F_3	1	2.60	1.81	2	1.40	0.97	1
F_4	1	1.13	0.68	1	1.17	0.53	1
F_5	0.5	7.93	6.42	7.5	4.53	3.77	4
F_6	0.5	2.74	2.04	2	1.72	1.34	1.25

Table A.13
Results of the Wilcoxon test on discrete functions.

Function	mixedGWO vs. MODGWO	amixedGWO vs. MODGWO
F_1	$4.12e-12(+)$	$4.12e-12(+)$
F_2	$1.09e-02(+)$	$1.09e-02(+)$
F_3	$9.92e-13(+)$	$1.96e-12(+)$
F_4	$4.62e-12(+)$	$2.90e-13(+)$
F_5	$5.46e-11(+)$	$8.05e-10(+)$
F_6	$1.18e-12(+)$	$1.18e-12(+)$
$+/=/-$	6/0/0	6/0/0

Table A.14
Results of unimodal benchmark functions in mixed search space with $T_{max} = 3000$.

F	Step	mixedGWO			amixedGWO		
		Avg.	Std.	Med.	Avg.	Std.	Med.
F_1	1	**0**	**0**	**0**	**0**	**0**	**0**
F_2	0.5	**0**	**0**	**0**	**0**	**0**	**0**
F_3	1	0.100	0.257	**0**	**0**	**0**	**0**
F_4	1	0.033	0.182	**0**	**0**	**0**	**0**
F_5	0.5	6.351	8.749	**2.771**	**4.084**	**5.318**	**2.771**
F_6	0.5	$2.93e-13$	$9.18e-13$	$4.74e-14$	$\mathbf{6.39e-14}$	$\mathbf{1.00e-13}$	$\mathbf{1.61e-14}$

Table A.15
Search spaces for the optimization methods. Symbol • means irrelevant.

Expected parameter index i	Search space		
	H_i	$\mathbf{d}_i^{ind}$	$\mathbf{d}_i^{val}$
1,2,3	$\min(I_i, 8)$	$[1, 2, \ldots, I_i]^T$	$\left[1, \frac{I_i}{H_i}, 2\frac{I_i}{H_i}, \ldots, I_i\right]^T$
4	2	$[0, 1]^T$	$\left[f_0^{mix}, f_1^{mix}\right]^T$
5,6	•	•	$[0; 1]^T$

Table A.16
Estimated parameters with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$.

Parameters		Expected values	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
K_1	*Avg.*	32	32.0000	31.5858	31.9088	27.2380	32.0000	24.7749	15.0052
K_2	*Avg.*	32	31.1999	32.0000	31.9635	27.2451	30.1649	30.1359	17.4724
K_3	*Avg.*	4	3.900	3.9631	3.8694	3.1280	3.6024	2.6075	2.7257
f^{mix}	*Avg.*	0	0.2000	0.4023	0.2884	0.4048	0.4173	0.2926	0.5756
λ_1	*Avg.*	0.15	0.1587	0.1387	0.1390	0.1037	0.0539	0.2137	0.1575
λ_2	*Avg.*	0.41	0.5008	0.5352	0.4172	0.4683	0.3790	0.3836	0.3174

Table A.17
Spectrum reconstruction error RE with $SNR_{in} = \infty$ and noise-free image as reference $\mathcal{X}_1$.

SNR_{in}	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
∞	$1.85e-03$	$4.14e-03$	$1.77e-03$	$1.41e-02$	$7.68e-03$	$4.14e-02$	$5.90e-02$
Rank	2	3	1	5	4	6	7

Table A.18
Estimated Parameters with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$.

Parameters		Expected values	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
K_1	$Avg.$	16	16.10	16.84	16.11	22.67	20.39	19.34	13.57
K_2	$Avg.$	16	15.90	15.77	16.05	19.85	19.57	20.05	22.02
K_3	$Avg.$	4	4	3.94	3.77	3.54	3.72	3.75	2.79
f^{mix}	$Avg.$	0	0.2	0.45	0.22	0.40	0.28	0.46	0.57
λ_1	$Avg.$	0.15	0.114	0.097	0.088	0.162	0.124	0.053	0.404
λ_2	$Avg.$	0.41	0.534	0.610	0.456	0.649	0.537	0.495	0.9999

Table A.19
Spectrum reconstruction error RE with $SNR_{in} = \infty$ and $\hat{\mathcal{X}}(16, 16, 4)$ as reference $\mathcal{X}_1$.

SNR_{in}	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
∞	$7.20e-03$	$8.90e-03$	$7.748e-03$	$1.017e-02$	$8.022e-03$	$1.35e-02$	$1.20e-01$
Rank	1	4	2	5	3	6	7

Table A.20
Results denoising SNR_{out} with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$.

SNR_{in}		Ref.	amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		3.99	**9.333**	7.659	7.745	5.560	8.429	8.45	6.42
	Rank	8	1	4	5	6	3	2	7
5 dB		8.79	12.180	11.477	**13.258**	7.878	11.893	7.97	8.34
	Rank	5	2	4	1	7	3	8	6
10 dB		13.24	**17.777**	14.159	17.108	7.719	15.310	8.88	2.49
	Rank	5	1	4	2	7	3	6	8
15 dB		16.61	17.804	18.587	**19.733**	13.388	15.815	15.22	14.43
	Rank	4	3	2	1	8	5	6	7
20 dB		18.26	**22.664**	20.869	22.653	17.417	18.504	15.37	13.21
	Rank	5	1	3	2	6	4	7	8
	Avg. Rank	5.4	1.6	3.4	2.2	6.8	3.6	5.8	7.2
	Overall Rank	5	1	3	2	7	4	6	8

Table A.21
Results spectrum reconstruction error RE with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$.

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		$2.81e-03$	$4.44e-03$	$3.51e-03$	$5.30e-03$	$\mathbf{2.80e-03}$	$4.00e-03$	$2.53e-02$
	Rank	2	5	3	6	1	4	7
5 dB		$\mathbf{2.34e-03}$	$4.41e-03$	$3.41e-03$	$5.45e-03$	$3.12e-03$	$5.63e-03$	$5.59e-03$
	Rank	1	4	3	5	2	7	6
10 dB		$\mathbf{2.47e-03}$	$3.12e-03$	$3.02e-03$	$3.68e-03$	$2.69e-03$	$2.79e-03$	$5.17e-03$
	Rank	1	5	4	6	2	3	7
15 dB		$\mathbf{1.89e-03}$	$2.82e-03$	$3.13e-03$	$4.40e-03$	$2.66e-03$	$2.83e-03$	$5.63e-03$
	Rank	1	3	5	6	2	3	7
20 dB		$\mathbf{1.92e-03}$	$2.82e-03$	$2.09e-03$	$2.75e-03$	$2.59e-03$	$4.26e-03$	$4.28e-03$
	Rank	1	5	2	4	3	6	7
	Avg. Rank	1.2	4.4	3.4	5.4	2	4.6	6.8
	Overall Rank	1	4	3	6	2	5	7

Table A.22
Results Global best with various SNR_{in} values in dB and Fourier Wiener as reference $\mathcal{X}_1$.

SNR_{in}		amixedGWO	PSO	GWO	ABC	TSA	GA	SA
0 dB		**2.094e − 03**	2.894e − 03	2.699e − 03	2.264e − 03	2.433e − 03	6.14e − 03	1.74e − 02
	Rank	1	5	4	2	3	6	7
5 dB		1.372e − 03	1.556e − 03	**1.166e − 03**	3.540e − 03	1.653e − 03	3.81e − 03	4.54e − 03
	Rank	2	3	1	5	4	6	7
10 dB		**6.642e − 04**	1.044e − 03	7.137e − 04	2.046e − 03	7.620e − 04	2.26e − 03	7.88e − 03
	Rank	1	4	2	5	3	6	7
15 dB		5.032e − 04	**4.641e − 04**	5.538e − 04	9.461e − 04	5.480e − 04	1.08e − 03	1.48e − 03
	Rank	2	1	4	5	3	6	7
20 dB		**2.929e − 04**	3.418e − 04	2.931e − 04	4.792e − 04	3.919e − 04	7.71e − 04	1.90e − 03
	Rank	1	3	2	5	4	6	7
	Avg. Rank	1.4	3.2	2.6	4.4	3.4	6	7
	Overall Rank	1	3	2	5	4	6	7

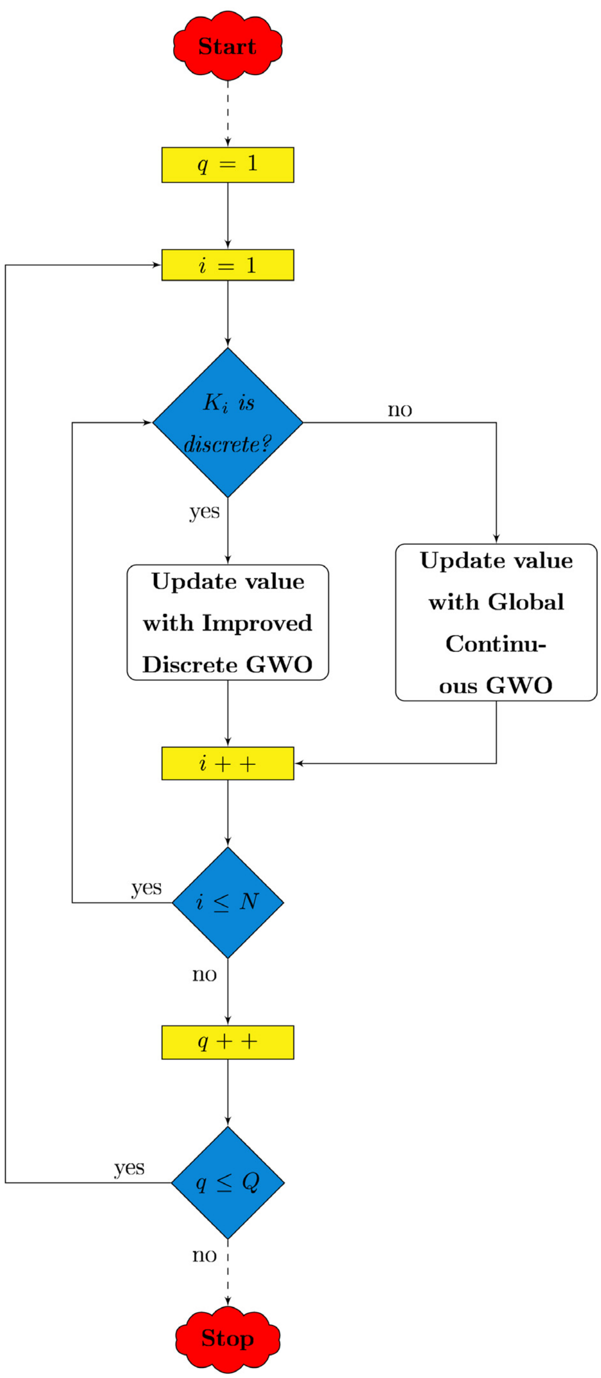


Fig. A.15. The mixedGWO's Update Wolves Step flowchart.

References

[1] Ibrahim Berkan Aydilek, A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems, Appl. Soft Comput. 66 (2018) 232–249, http://dx.doi.org/10.1016/j.asoc.2018.02.025, URL http://www.sciencedirect.com/science/article/pii/S156849461830084X.
[2] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, Gsa: A gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248, http://dx.doi.org/10.1016/j.ins.2009.03.004, special Section on High Order Fuzzy Sets. URL http://www.sciencedirect.com/science/article/pii/S0020025509001200.
[3] S. Mirjalili, A.H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, Appl. Soft Comput. 53 (Supplement C) (2017) 407–419, http://dx.doi.org/10.1016/j.asoc.2017.01.008, URL http://www.sciencedirect.com/science/article/pii/S1568494617300121.
[4] M. Minárová, D. Paternain, A. Jurio, J. Ruiz-Aranguren, Z. Taká, H. Bustince, Modifying the gravitational search algorithm: A functional study, Inform. Sci. 430–431 (Supplement C) (2018) 87–103, http://dx.doi.org/10.1016/j.ins.2017.11.033, URL http://www.sciencedirect.com/science/article/pii/S0020025517310964.
[5] A.K. Bhandari, V.K. Singh, A. Kumar, G.K. Singh, Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using kapur's entropy, Expert Syst. Appl. 41 (7) (2014) 3538–3560, http://dx.doi.org/10.1016/j.eswa.2013.10.059, URL http://www.sciencedirect.com/science/article/pii/S0957417413008919.
[6] L. Fang, H. Zuo, L. Pang, Z. Yang, X. Zhang, J. Zhu, Image reconstruction through thin scattering media by simulated annealing algorithm, Opt. Lasers Eng. 106 (2018) 105–110, http://dx.doi.org/10.1016/j.optlaseng.2018.02.020, URL http://www.sciencedirect.com/science/article/pii/S0143816617313714.
[7] J.H. Holland, Genetic algorithms, Sci. Am. 267 (1) (1992) 66–72.
[8] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference on Neural Networks, Perth, 1995, pp. 1942–1948.
[9] R.C. Eberhart, J. Kennedy, et al., A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Vol. 1, New York, NY, 1995, pp. 39–43.
[10] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61, http://dx.doi.org/10.1016/j.advengsoft.2013.12.007, URL http://www.sciencedirect.com/science/article/pii/S0965997813001853.
[11] N. Mittal, U. Singh, B. Singh Sohi, Modified grey wolf optimizer for global engineering optimization, Appl. Comput. Intell. Soft Comput. (2016) 7950348, http://dx.doi.org/10.1155/2016/7950348, 16 pages.
[12] A. Sadollah, H. Sayyaadi, D.G. Yoo, H.M. Lee, J.H. Kim, Mine blast harmony search: A new hybrid optimization method for improving exploration and exploitation capabilities, Appl. Soft Comput. 68 (2018) 548–564, http://dx.doi.org/10.1016/j.asoc.2018.04.010, URL http://www.sciencedirect.com/science/article/pii/S1568494618301960.
[13] X. Zhang, Q. Kang, J. Cheng, X. Wang, A novel hybrid algorithm based on biogeography-based optimization and grey wolf optimizer, Appl. Soft Comput. 67 (2018) 197–214, http://dx.doi.org/10.1016/j.asoc.2018.02.049, URL http://www.sciencedirect.com/science/article/pii/S156849461830108X.
[14] S. Mirjalili, S. Saremi, S.M. Mirjalili, L. dos S. Coelho, Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization, Expert Syst. Appl. 47 (2016) 106–119, http://dx.doi.org/10.1016/j.eswa.2015.10.039, URL http://www.sciencedirect.com/science/article/pii/S0957417415007435.
[15] M. Kohli, S. Arora, Chaotic grey wolf optimization algorithm for constrained optimization problems, J. Comput. Des. Eng. 5 (4) (2018) 458–472.
[16] M. Mitchell, An Introduction to Genetic Algorithms, MIT press, 1998.
[17] K.-F. Man, K.S. TANG, S. Kwong, Genetic Algorithms: Concepts and Designs, Springer Science & Business Media, 2012.
[18] W.T. Reeves, Particle systems, a technique for modeling a class of fuzzy objects, ACM Trans. Graph. 2 (2) (1983) 91–108.

[19] V.K. Kamboj, A novel hybrid pso–gwo approach for unit commitment problem, Neural Comput. Appl. (2015) 1–13.

[20] S. Mirjalili, How effective is the grey wolf optimizer in training multi-layer perceptrons, Appl. Intell. 43 (1) (2015) 150–161, http://dx.doi.org/10.1007/s10489-014-0645-7.

[21] M.H. Qais, H.M. Hasanien, S. Alghuwainem, Augmented grey wolf optimizer for grid-connected pmsg-based wind energy conversion systems, Appl. Soft Comput. (2018).

[22] A.A. Heidari, P. Pahlavani, An efficient modified grey wolf optimizer with levy flight for optimization tasks, Appl. Soft Comput. 60 (Supplement C) (2017) 115–134, http://dx.doi.org/10.1016/j.asoc.2017.06.044, URL http://www.sciencedirect.com/science/article/pii/S1568494617303873.

[23] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, J. Global Optim. 39 (3) (2007) 459–471, http://dx.doi.org/10.1007/s10898-007-9149-x.

[24] M.S. Kiran, Tsa: Tree-seed algorithm for continuous optimization, Expert Syst. Appl. 42 (19) (2015) 6686–6698, http://dx.doi.org/10.1016/j.eswa.2015.04.055, URL http://www.sciencedirect.com/science/article/pii/S0957417415002973.

[25] A. Babalik, A.C. Cinar, M.S. Kiran, A modification of tree-seed algorithm using deb's rules for constrained optimization, Appl. Soft Comput. 63 (2018) 289–305, http://dx.doi.org/10.1016/j.asoc.2017.10.013, URL http://www.sciencedirect.com/science/article/pii/S156849461730618X.

[26] X.S. Yang, S. Deb, Cuckoo search via levy flights, in: 2009 World Congress on Nature Biologically Inspired Computing (NaBIC), 2009, pp. 210–214, http://dx.doi.org/10.1109/NABIC.2009.5393690.

[27] Z. Li, X. Huang, Glowworm swarm optimization and its application to blind signal separation, Math. Probl. Eng. (2016) 5481602, http://dx.doi.org/10.1155/2016/5481602, 8 pages.

[28] A. Babalik, A. Ozkis, S.A. Uymaz, M.S. Kiran, A multi-objective artificial algae algorithm, Appl. Soft Comput. 68 (2018) 377–395, http://dx.doi.org/10.1016/j.asoc.2018.04.009, URL http://www.sciencedirect.com/science/article/pii/S1568494618301959.

[29] M. Clerc, Tribes-un exemple d'optimisation par essaim particulaire sans parametres de contrôle, Optimisation par Essaim Particulaire (OEP 2003), Paris, France 64.

[30] K. Chen, T. Li, T. Cao, Tribe-pso: A novel global optimization algorithm and its application in molecular docking, Chemom. Intell. Lab. Syst. 82 (1–2) (2006) 248–259, http://dx.doi.org/10.1016/j.chemolab.2005.06.017, selected Papers from the International Conference on Chemometrics and Bioinformatics in AsiaCCBA 2004International Conference on Chemometrics and Bioinformatics in Asia. URL http://www.sciencedirect.com/science/article/pii/S0169743905001346.

[31] Y. Cooren, M. Clerc, P. Siarry, Performance evaluation of tribes, an adaptive particle swarm optimization algorithm, Swarm Intell. 3 (2) (2009) 149–178.

[32] B. Yang, X. Zhang, T. Yu, H. Shu, Z. Fang, Grouped grey wolf optimizer for maximum power point tracking of doubly-fed induction generator based wind turbine, Energy Convers. Manage. 133 (2017) 427–443, http://dx.doi.org/10.1016/j.enconman.2016.10.062, URL http://www.sciencedirect.com/science/article/pii/S0196890416309761.

[33] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.

[34] M. Dorigo, T. Stützle, The ant colony optimization metaheuristic: Algorithms, applications, and advances, in: Handbook of Metaheuristics, Springer, 2003, pp. 250–285.

[35] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, Vol. 5, 1997, pp. 4104–4108, http://dx.doi.org/10.1109/ICSMC.1997.637339.

[36] Z. Beheshti, S.M. Shamsuddin, S. Hasan, Memetic binary particle swarm optimization for discrete optimization problems, Inform. Sci. 299 (2015) 58–84, http://dx.doi.org/10.1016/j.ins.2014.12.016, URL http://www.sciencedirect.com/science/article/pii/S0020025514011530.

[37] A. Gherboudj, A. Layeb, S. Chikhi, Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm, Int. J. Bio-Inspired Comput. 4 (4) (2012) 229–236.

[38] A.C. Cinar, M.S. Kiran, Similarity and logic gate-based tree-seed algorithms for binary optimization, Comput. Ind. Eng. 115 (2018) 631–646, http://dx.doi.org/10.1016/j.cie.2017.12.009, URL http://www.sciencedirect.com/science/article/pii/S0360835217305752.

[39] E. Emary, H.M. Zawbaa, A.E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381.

[40] C. Lu, S. Xiao, X. Li, L. Gao, An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production, Adv. Eng. Softw. 99 (2016) 161–176, http://dx.doi.org/10.1016/j.advengsoft.2016.06.004, URL http://www.sciencedirect.com/science/article/pii/S0965997816301260.

[41] C. Lu, L. Gao, X. Li, S. Xiao, A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry, Eng. Appl. Artif. Intell. 57 (2017) 61–79, http://dx.doi.org/10.1016/j.engappai.2016.10.013, URL http://www.sciencedirect.com/science/article/pii/S0952197616301968.

[42] G. Komaki, E. Teymourian, V. Kayvanfar, Z. Booyavi, Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem, Comput. Ind. Eng. 105 (2017) 158–173, http://dx.doi.org/10.1016/j.cie.2017.01.006, URL http://www.sciencedirect.com/science/article/pii/S0360835217300062.

[43] S.-W. Lin, K.-C. Ying, S.-C. Chen, Z.-J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, Expert Syst. Appl. 35 (4) (2008) 1817–1824, http://dx.doi.org/10.1016/j.eswa.2007.08.088, URL http://linkinghub.elsevier.com/retrieve/pii/S0957417407003752.

[44] Y. Zhang, P. Zhang, Machine training and parameter settings with social emotional optimization algorithm for support vector machine, Pattern Recognit. Lett. 54 (2015) 36–42, http://dx.doi.org/10.1016/j.patrec.2014.11.011, URL http://www.sciencedirect.com/science/article/pii/S0167865514003614.

[45] X. Liu, H. Fu, PSO-based support vector machine with cuckoo search technique for clinical disease diagnoses, Sci. World J. 2014 (2014) 1–7, http://dx.doi.org/10.1155/2014/548483, URL http://www.hindawi.com/journals/tswj/2014/548483/.

[46] C.-F. Chao, M.-H. Horng, The construction of support vector machine classifier using the firefly algorithm, Intell. Neurosci. 2015 (2015) http://dx.doi.org/10.1155/2015/212719, 2:2–2:2.

[47] E. Elhariri, N. El-Bendary, A.E. Hassanien, A. Abraham, Grey wolf optimization for one-against-one multi-class support vector machines, in: 2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR), 2015, pp. 7–12, http://dx.doi.org/10.1109/SOCPAR.2015.7492781.

[48] Y. Zhong, A. Ma, Y. soon Ong, Z. Zhu, L. Zhang, Computational intelligence in optical remote sensing image processing, Appl. Soft Comput. 64 (2018) 75–93, http://dx.doi.org/10.1016/j.asoc.2017.11.045, URL http://www.sciencedirect.com/science/article/pii/S1568494617307081.

[49] S. Suresh, S. Lal, Multilevel thresholding based on chaotic darwinian particle swarm optimization for segmentation of satellite images, Appl. Soft Comput. 55 (2017) 503–522, http://dx.doi.org/10.1016/j.asoc.2017.02.005, URL http://www.sciencedirect.com/science/article/pii/S156849461730073X.

[50] S. Medjahed, T.A. Saadi, A. Benyettou, M. Ouali, Gray wolf optimizer for hyperspectral band selection, Appl. Soft Comput. 40 (2016) 178–186, http://dx.doi.org/10.1016/j.asoc.2015.09.045, URL http://www.sciencedirect.com/science/article/pii/S1568494615006250.

[51] A. Sahoo, S. Chandra, Multi-objective grey wolf optimizer for improved cervix lesion classification, Appl. Soft Comput. 52 (2017) 64–80, http://dx.doi.org/10.1016/j.asoc.2016.12.022, URL http://www.sciencedirect.com/science/article/pii/S1568494616306408.

[52] M. Dorigo, G.D. Caro, L.M. Gambardella, Ant algorithms for discrete optimization, Artif. Life 5 (2) (1999) 137–172, http://dx.doi.org/10.1162/106454699568728.

[53] J. Liang, B.Y. Qu, P. Suganthan, Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization (12 2013).

[54] D. Muti, S. Bourennane, J. Marot, Lower-rank tensor approximation and multiway filtering, SIAM J. Matrix Anal. Appl. 30 (3) (2008) 1172–1204, http://dx.doi.org/10.1137/060653263.

[55] D. Muti, S. Bourennane, Multidimensional filtering based on a tensor approach, Sig. Process. J. Elsevier 85 (12) (2005) 2338–2353.

[56] S.-W. Lin, K.-C. Ying, C.-Y. Lee, Z.-J. Lee, An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection, Appl. Soft Comput. 12 (10) (2012) 3285–3290, http://dx.doi.org/10.1016/j.asoc.2012.05.004, URL http://www.sciencedirect.com/science/article/pii/S1568494612002402.

[57] A. Zidi, J. Marot, S. Bourennane, K. Spinnler, Bio-inspired optimization algorithms for automatic estimation of multiple subspace dimensions in a tensor-wavelet denoising algorithm, J. Remote Sens. Technol. 4 (1) (2016) 90–114.

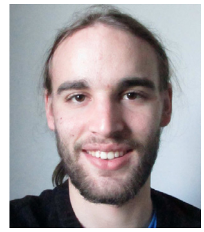
[58] T. Lin, S. Bourennane, Hyperspectral image processing by jointly filtering wavelet component tensor, IEEE Trans. Geosci. Remote Sens. 51 (6) (2013) 3529–3541.

[59] N. Dobigeon, Y. Altmann, N. Brun, S. Moussaoui, Linear and nonlinear unmixing in hyperspectral imaging, Data Handling in Science and Technology: Resolving Spectral Mixtures, 2016, 41.

[60] J.M. Nascimento, J.M. Dias, Vertex component analysis: A fast algorithm to unmix hyperspectral data, IEEE Trans. Geosci. Remote Sens. 43 (4) (2005) 898–910.

[61] V.P. Pauca, J. Piper, R.J. Plemmons, Nonnegative matrix factorization for spectral data analysis, Linear Algebra Appl. 416 (1) (2006) 29–47.

[62] Y. Altmann, A. Halimi, N. Dobigeon, J.-Y. Tourneret, Supervised nonlinear spectral unmixing using a postnonlinear mixing model for hyperspectral imagery, IEEE Trans. Image Process. 21 (6) (2012) 3017–3025.
[63] A. Halimi, Y. Altmann, N. Dobigeon, J.-Y. Tourneret, Nonlinear unmixing of hyperspectral images using a generalized bilinear model, IEEE Trans. Geosci. Remote Sens. 49 (11) (2011) 4153–4162.
[64] J.M. Molero, E.M. Garzón, I. García, A. Plaza, Analysis and optimizations of global and local versions of the rx algorithm for anomaly detection in hyperspectral data, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 6 (2) (2013) 801–814.
[65] R.O. Green, C.M. Sarture, C.J. Chovit, J.A. Faust, P. Hajek, H. Ian Novak, Aviris: A new approach to earth remote sensing, Opt. Photonics News 6 (1) (1995) 30, http://dx.doi.org/10.1364/OPN.6.1.000030, URL http://www.osa-opn.org/abstract.cfm?URI=opn-6-1-30.
[66] S. Holzwarth, A. Muller, M. Habermeyer, R. Richter, A. Hausold, S. Thiemann, P. Strobl, Hysens-dais 7915/rosis imaging spectrometers at dlr, in: Proceedings of the 3rd EARSeL Workshop on Imaging Spectroscopy, 2003, pp. 3–14.
[67] X. Liu, S. Bourennane, C. Fossati, Nonwhite noise reduction in hyperspectral images, IEEE Geosci. Remote Sens. Lett. 9 (3) (2012) 368–372.
[68] J. Kerekes, J. Baum, Full spectrum spectral imaging system analytical model, IEEE Trans. Geosci. Remote Sens. 43 (3) (2005) 571–580.
[69] Z. Liu, Z. Liu, Z. Zhu, Y. Shen, J. Dong, Simulated annealing for a multi-level nurse rostering problem in hemodialysis service, Appl. Soft Comput. 64 (2018) 148–160, http://dx.doi.org/10.1016/j.asoc.2017.12.005.



Benoit Martin received his Master degree from the Institut National des Sciences Appliquées (INSA) of Rennes, France, in 2015. Currently, he his preparing a thesis for an industrial PhD of the Ecole Centrale de Marseille, in between the Fresnel Institute and the company IntuiSense. His research interests include applied image processing and multidimensional signal processing.

Julien Marot received his PhD degree from the Aix Marseille University in 2007. He performed a post-doc in Fraunhofer Institute IIS, Germany, and is now associate professor in the multidimensional signal processing group (GSM), Fresnel Institute, Marseille, France. His research interests include applied image processing, multidimensional signal processing and array processing.

Salah Bourennane received his PhD degree from Institut National Polytechnique de Grenoble, France, in 1990. Currently, he is full Professor at the Ecole Centrale de Marseille, France. His research interests are in statistical signal processing, array processing, image processing, multidimensional signal processing and performances analysis.