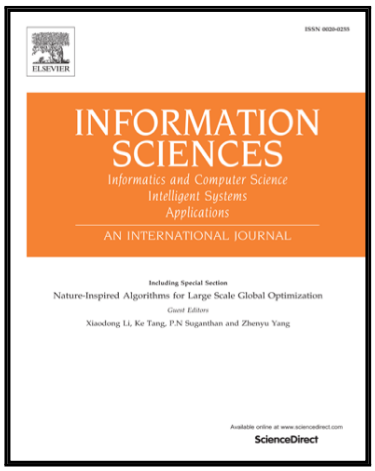


# HyperSPAM: A study on Hyper-heuristic Coordination Strategies in the Continuous Domain

Fabio Caraffini, Ferrante Neri, Michael Epitropakis

PII: S0020-0255(18)30851-X
DOI: https://doi.org/10.1016/j.ins.2018.10.033
Reference: INS 14014

To appear in: *Information Sciences*

Received date: 5 September 2018
Revised date: 16 October 2018
Accepted date: 23 October 2018

Please cite this article as: Fabio Caraffini, Ferrante Neri, Michael Epitropakis, HyperSPAM: A study on Hyper-heuristic Coordination Strategies in the Continuous Domain, *Information Sciences* (2018), doi: https://doi.org/10.1016/j.ins.2018.10.033

# HyperSPAM: A study on Hyper-heuristic Coordination Strategies in the Continuous Domain

Fabio Caraffini, Ferrante Neri

*Institute of Artificial Intelligence, School of Computer Science and Informatics, De Montfort University, Leicester, United Kingdom*

Michael Epitropakis

*Data Science Institute, Department of Management Science, Lancaster University Management School, Lancaster University, Lancaster, United Kingdom*

**Abstract**

This article proposes a simplistic algorithmic framework, namely hyperSPAM, composed of three search algorithms for addressing continuous optimisation problems. The Covariance Matrix Adaptation Evolution Strategy (CMAES) is activated at the beginning of the optimisation process as a preprocessing component for a limited budget. Subsequently, the produced solution is fed to the other two single-solution search algorithms. The first performs moves along the axes while the second makes use of a matrix orthogonalization to perform diagonal moves.

Four coordination strategies, in the fashion of hyperheuristics, have been used to coordinate the two single-solution algorithms. One of them is a simple randomized criterion while the other three are based on a success based reward mechanism. The four implementations of the hyperSPAM framework have been tested and compared against each other and modern metaheuristics on an extensive set of problems including theoretical functions and real-world engineering problems.

Numerical results show that the different versions of the framework display broadly a similar performance. One of the reward schemes appears to be marginally better than the others. The simplistic random coordination also displays a very good performance.

*Email address:* fabio.caraffini@dmu.ac.uk (Fabio Caraffini)
*URL:* www.tech.dmu.ac.uk/~fcaraf00 (Fabio Caraffini)

All the implementations of hyperSPAM significantly outperform the other algorithms used for comparison.

*Keywords:* Automated design of algorithms, Hyper-heuristics, Memetic computing, Optimization algorithms, Adaptive Operator Selection

## 1. Introduction

The development of modern technologies imposes the integration of computational intelligence and soft computing techniques in robotics and engineering [36, 21, 46] with the aim of making accurate and often real-time decisions [42, 24].

Behind a decision problem there is an optimisation problem and behind a machine which makes a decision there is often a search algorithm seeking for the optimal solution or its approximation, see [50]. The design of an algorithm is often not a trivial task and the fact that there is no universal optimiser [48, 20] suggests that efficiently designed algorithms should specifically address the features of the problems to optimise.

Following this consideration, for over thirty years, researchers in metaheuristic optimisation attempted to overcome the difficulties of algorithmic design. A popular approach consists of integrating within the algorithm the knowledge of the problem [16, 35, 51] and a domain specific design based on tuning and ad-hoc operators.

Another major approach is the use of multiple algorithms/operators in the hope that algorithmic frameworks making use of multiple search strategies behave flexibly and adapt to solve a diverse array of problems. In their early implementations, algorithms employing this approach are known as hybrid algorithms. Subsequently, the research in the field of hybrid approaches focussed on the automatic/semiautomatic coordination of the various search strategies integrated into the framework. However, historically, the nomenclature of modern hybrid algorithms is not based on the coordination strategy. On the contrary, the nomenclature reflects how researchers interpret the concept of hybridisation. A possible incomplete classification of hybrid algorithms is the following.

- **Portfolio algorithms** [37, 7]: a hybrid algorithm is a library of search algorithms that share a computational budget to concurrently contribute to the solution of the

problem.

- **Hyperheuristics** [3, 2, 10]: a hybrid algorithm is a library of search algorithms endowed with a coordination engine that selects and activates the various algorithms, or generates dedicated search components for the problem at hand.
- **Memetic Computing algorithms** [33, 34, 49, 22]: a hybrid algorithm is a structure which contains a main solver and multiple search algorithms sitting within an algorithmic framework.
- **Ensemble of algorithms/strategies** [31, 30]: multiple and complementary elements (search strategies or generic elements to handle issues in optimisation) are used within an algorithmic framework.

It must be remarked that this classification, based on historical nomenclature, does not highlights the algorithmic differences among the approaches but only the view/algorithmic philosophies of the researchers [43, 11]. As it can be noticed from their descriptions, these four categories have overlapping features and cannot be distinguished as clear separate algorithmic philosophies.

Within the context of optimisation performed by hybrid algorithms and their design issues, this article investigates the role and effectiveness of diverse coordination strategies for search algorithms. This study is performed on a set of a few and simple search algorithms by embracing some studies previously carried out in the literature, that is the so called Ockham's razor for Memetic Computing. More specifically, paper [23] experimentally demonstrates that many hybrid search algorithms are excessively complex, see also [40, 27]. The simple single-solution algorithm composed of three search strategies has been shown to be able to outperform complex population-based algorithms. Following this simplicity philosophy, in [4] an algorithmic framework is proposed, namely Parallel Memetic Structure, which makes use of two simple local search components to alternitavely perturb a single solution. The Ockham's razor has been employed in other studies [5, 38, 39].

In [6] also a simplistic design inspired by the Ockham's razor is proposed. The algorithm in [6] achieves a very good performance by using a Parallel Memetic Structure

[4] and a pre-processing component based on the Covariance Matrix Adaptive Evolution Strategy (CMAES) [18] as well as an estimator of the problem separability. The estimation of the separability was then used to fix an activation probability for the other search algorithms. The resulting framework was named Separability Analyzer for Automatic Memes (SPAM). In [11] a randomised version of the SPAM framework has been proposed and tested.

The present article makes use of the search algorithms of the SPAM framework and proposes, instead of exploiting the information about the separability, the integration within it of four automated adaptive strategies for coordinating the search algorithms. The four resulting implementations have been tested and compared among them and against modern metaheuristics as well as against competition winner algorithms. Results are given for two theoretical testbeds and three real-world problems. Since the proposed framework is based on the SPAM search algorithms and, like Hyperheuristics, makes use of coordination components we will refer to it as hyperSPAM.

The remainder of this article is organised in the following way. Section 2 described the component of the framework as well as the three coordination strategies. Section 3 describes the experimental setup, shows and comments the results of the proposed framework. Section 4 gives the conclusion of this study.

## 2. The hyperSPAM Framework

Before analysing the components of the framework, let us state the general problem and define the notation. Without a loss of generality, we will refer to the minimization problem of an objective function (or fitness) $f(\mathbf{x})$, where the candidate solution $\mathbf{x}$ is a vector of $n$ design variables $x_1, x_2, \ldots, x_n$ in an $n$-dimensional decision space $\mathbf{D}$. Thus, the optimisation problem considered in this paper consists of the detection of that solution $\mathbf{x}^* \in \mathbf{D}$ such that $f(\mathbf{x}^*) < f(\mathbf{x})$, and this is valid $\forall \mathbf{x} \in \mathbf{D}$. Array variables are highlighted in bold face throughout this paper.

The proposed hyperSPAM is composed of the following three search algorithms:

- the Covariance Matrix Adaptation Evolution Strategy
- the S algorithm

- the R algorithm

and a re-sampling component to prevent from search stagnation.

The following sections describe separately each search algorithm

### 2.1. Pre-processing: Covariance Matrix Adaptation Evolution Strategy

At the beginning of the optimisation process, a set of $\mu$ candidate solutions is sampled within $\mathbf{D}$. For a limited portion of the budget, the Covariance Matrix Adaptation Evolution Strategy (CMAES) with rank-$\mu$-update and weighted recombination, see [18], is applied. This optimisation method consists of

1. sampling from a multivariate distribution $\lambda$ points;
2. compute their fitness values;
3. update the shape of the multivariate distribution in order to progressively adapt to the basins of attraction.

The sampling of the generic $k^{th}$ individual at the generation $g+1$ is given by:

$$x_k^{(g+1)} \sim \mathcal{N}\left(\langle \mathbf{x} \rangle_w^g, (\sigma^g)^2 \mathbf{C}^g\right) \tag{1}$$

where $\mathcal{N}\left(m, \sigma^2 C\right)$ is a multivariate normal distribution of mean $m$, stepsize $\sigma$, and estimated covariance matrix $\mathbf{C}$. The mean value $\langle \mathbf{x} \rangle_w^g$ is a weighted sum of the $\mu$ candidate solutions ($\mu \leq \lambda$) displaying the best performance at the generation $g$, see [18] for details.

The stepsize $\sigma$ and covariance matrix are progressively updated at each generation. The update rule for the covariance matrix is:

$$\begin{aligned}\mathbf{C}^{g+1} = (1 - c_{cov})\mathbf{C}^g + c_{cov}\tfrac{1}{\mu_{cov}}\mathbf{p_c}^{g+1}\left(\mathbf{p_c}^{g+1}\right)^T + \\ + c_{cov}\left(1 - \tfrac{1}{\mu_{cov}}\right)\sum_{i=1}^{\mu}\left(\mathbf{x}_{i-b}^{g+1} - \langle \mathbf{x} \rangle_w^g\right)\left(\mathbf{x}_{i-b}^{g+1} - \langle \mathbf{x} \rangle_w^g\right)^T\end{aligned}$$

where $\mathbf{x}_{i-b}^g$ denotes the $i^{th}$ best individuals at the generation $g$, $c_{cov}$ is a parameter determining the learning rate for the estimated covariance matrix $\mathbf{C}$, and $\mathbf{p_c}$ is a vector namely evolution path that determines the adaptation of the covariance matrix. The update formula is given by:

$$\mathbf{p_c}^{g+1} = (1-c_c)\mathbf{p_c}^{g} + H_\sigma^{g+1}\sqrt{c_c\,(2-c_c)}\,\frac{\sqrt{\mu_{eff}}}{\sigma^g}$$
$$\left(\langle \mathbf{x}\rangle_w^{g+1} - \langle \mathbf{x}\rangle_w^{g}\right)$$

where $\mu_{eff} = \frac{1}{\sum_1^\mu w_i^2}$, $c_c$ is a parameter, $H_\sigma^{g+1}$ is a function defined by cases that can take values 0 or 1. Also the stepsize $\sigma^{g+1}$ is iteratively updated. Details about CMAES implementations can be found e.g. in [19], and [18]. At the end of each generation the $\mu$ individuals displaying the best performance are selected and used to compute $\langle \mathbf{x}\rangle_w^{g+1}$.

The CMAES is run only once and with a limited computational budget. At the end of this CMAES run, that is the preprocessing of the algorithm, a final population is calculated. Within this population, the candidate solution $\mathbf{x}$ with the highest performance is processed by the single solution algorithms described in the following sections.

### *2.2. The first single solution search algorithm: the S algorithm*

The S implementation requires a generic input solution $\mathbf{x}$ and a trial solution $\mathbf{x^t}$. This search algorithm is based on one of the operators used in [45]. S perturbs the candidate by computing, for each coordinate $i$,

$$x_i^t = x_i - \xi, \tag{2}$$

where $\xi$ is the exploratory radius. Subsequently, if $\mathbf{x^t}$ outperforms $\mathbf{x}$, the solution $\mathbf{x}$ is updated (the values of $\mathbf{x^t}$ are copied in it), otherwise a half step in the opposite direction is performed:

$$x_i^t = x_i + \frac{\xi}{2}. \tag{3}$$

Again, $\mathbf{x^t}$ replaces $\mathbf{x}$ if it outperforms it. If there is no update, i.e. the exploration was unsuccessful, the radius $\xi$ is halved. This exploration is repeated for all the design variables. We indicate with $\mathbf{x}'$ the output of the S implementation. For the sake of clarity, Algorithm 1 describes the working principles of the S search algorithm.

### *2.3. The second single solution search algorithm: the R algorithm*

The second search algorithm is the Rosenbrock algorithm (R), see [41]. At the beginning of the optimisation of this component, R is similar to S as it explores each

**Algorithm 1** Pseudo-code of the S search algorithm.

INPUT $\mathbf{x}$
**while** condition on the local computational budget **do**
  $\mathbf{x^t} = \mathbf{x}$
  **for** $i = 1 : n$ **do**
    $x_i^t = x_i - \xi$
    **if** $f\left(\mathbf{x^t}\right) >= f\left(\mathbf{x}\right)$ **then**
      $x_i^t = x_i + \frac{\xi}{2}$
    **end if**
    **if** $f\left(\mathbf{x^t}\right) < f\left(\mathbf{x}\right)$ **then**
      $\mathbf{x} = \mathbf{x^t}$
    **end if**
  **end for**
  **if** $\mathbf{x} \neq \mathbf{x^t}$ **then**
    $\xi = \frac{\xi}{2}$
  **end if**
**end while**
OUTPUT $\mathbf{x}$

of the $n$ directions, perturbing the input solution $\mathbf{x}$ with an initial step size vector $\mathbf{h}$. A matrix $\mathbf{A}$ is also initialized as the identity matrix. Each step of this algorithm consists of the following. As long as new improvements are found, for $j = 1, 2, \ldots, n$, a new trial point $\mathbf{x^t}$ is generated perturbing the each $i^{th}$ design variable of solution $\mathbf{x}$ in the following way:

$$x_i^t = x_i + h_j \cdot A_{i,j} \tag{4}$$

for $i = 1, 2, \ldots, n$. In case of success (the trial solution outperforms the solution $\mathbf{x}$), $\mathbf{x}$ is updated and the step size is increased of a factor $\alpha$ ($h_j = \alpha \cdot h_j$), otherwise it is decreased by means of a factor $\beta$ and the opposite direction is tried ($h_j = -\beta \cdot h_j$). As said, this procedure is repeated until it is possible to improve upon the solution $\mathbf{x}$. Once every possible success has been found and exploited in each base direction, the coordinate system is rotated towards the approximated gradient by means of the Gram-Schmidt orthogonalization procedure. This operation results into an update of the matrix $\mathbf{A}$. After the orthogonalization, the step size vector $\mathbf{h}$ is reinitialized and the procedure is repeated, using the rotated coordinate system, perturbing again the current elite $\mathbf{x}$ according to Eq. (4): it is important to notice that, when a rotated coordinate system is used, i.e. the matrix $\mathbf{A}$ is no longer an identity matrix, this trial generation mechanism corresponds to a diagonal move by following the direction given by the gradient. The Rosenbrock Algorithm terminates when a stop criterion is met. The stop criterion is given by two conditions. The first criterion is based on the minimum element of the perturbation vector $\mathbf{h}$, the second is based on the minimum difference between $\mathbf{x^t}$ and $\mathbf{x}$ design variables. More specifically, R is continued until the following condition is true:

$$\min(|\mathbf{h}|) > \varepsilon \; OR \; \min(|\mathbf{x^t} - \mathbf{x}|) > \varepsilon \tag{5}$$

where min() is the minimum vector element. If for a step there is no improvement at all, only the first condition is considered. At the end of the R application the output is indicated with $\mathbf{x}'$.

Algorithm 2

**Algorithm 2** Pseudo-code of the R search algorithm

INPUT $\mathbf{x}$, $\alpha$, $\beta$, $\varepsilon$

initialise $\mathbf{h}$ to 0.1 of the ranges of the decision space $\mathbf{D}$

initialise $\mathbf{A} = eye(n)$ identity matrix of size $n$

**while** $\min(|\mathbf{h}|) > \varepsilon$ AND $\min(|\mathbf{x^t} - \mathbf{x}|) > \varepsilon$ **do**

  **while** improvements can be found **do**

    **for** $j = 1 : n$ **do**

      **for** $i = 1 : n$ **do**

        $x_i^t = x_i + h_j \cdot A_{i,j}$

      **end for**

      **if** $f(\mathbf{x^t}) <= f(\mathbf{x})$ **then**

        $h_j = \alpha h_j$

        $\mathbf{x} = \mathbf{x^t}$

        Flag improvement tracker

      **else**

        $h_j = -\beta h_j$

      **end if**

    **end for**

  **end while**

  Apply the Gram-Schmidt orthogonalisation to the basis of vectors represented by the matrix $\mathbf{A}$ and generate a new matrix $\mathbf{A}$

**end while**

OUTPUT $\mathbf{x}$

### 2.4. Coordination mechanisms

After the CMAES activation, S and R are alternatively activated in order to improve upon the solution $\mathbf{x}$. The activation of the two search algorithms (operators) is coordinated by selective hyper-heuristic mechanisms, i.e., Adaptive Operator Selection (AOS) models [2, 15, 10].

The adaptive operator selection models utilize theoretical models and empirical algorithms from a wide variety of different scientific fields, to select the most suitable action/operator to be applied in the next step, based on their historical performance gains. In the proposed, framework the AOS models ($\mathscr{M}$) select at each step between the S and R algorithms. A credit assignment module ($\mathscr{C}$) is devoted to estimate the quality of each algorithm and assign a score, or credit, to the applied algorithm based on its performance gains.

Next, we briefly present in detail the two main modules of the coordination mechanism, the utilized credit assignment module (Section 2.4.1) and the adaptive operator selection models (Section 2.4.2).

#### 2.4.1. Credit assignment module

The main role of the credit assignment module is to assess the quality of all available actions to be taken based on their historical performance. The well performing actions should have better quality than the worst performing actions, at the current state of the search based on their search effectiveness.

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_\kappa\}$ denote the set of the $\kappa$ available search algorithms. In our simplistic case, $\kappa = 2$, that is $\alpha_1 = \mathrm{S}$ and $\alpha_2 = \mathrm{R}$. In other words, the set of search algorithms is $A = \{\mathrm{S}, \mathrm{R}\}$.

Let us consider a generic search algorithm $\alpha_i$ and let us indicate with $\mathbf{x}^{\mathbf{in}}$ and $\mathbf{x}^{\mathbf{out}}$ the solution before and after the application of $\alpha_i$, respectively. Let us assume that $\mathbf{x}^{\mathbf{out}}$ is initialized equal to $\mathbf{x}^{\mathbf{in}}$ and updated only when $\alpha_i$ has improved upon the initial solution. Under these conditions, the fitness improvement $s_{\alpha_i}$ of an operator at the time $t$ expressed as

$$s_{\alpha_i}(t) = f\left(\mathbf{x}^{\mathbf{out}}\right) - f\left(\mathbf{x}^{\mathbf{in}}\right) \tag{6}$$

is recorded and stored in a dynamic array $\mathbf{S}_{\alpha_i}$. Clearly, the index $s_{\alpha_i}(t)$ is equal to zero if no improvement is achieved.

After $t$ times the search algorithm $\alpha_i$ has been activated the reward $r_{\alpha_i}(t)$ is calculated:

$$r_{\alpha_i}(t) = \frac{\sum_{j=1}^{|\mathbf{S}_{\alpha_i}|} s_{\alpha_i}(j)}{|\mathbf{S}_{\alpha_i}|} \tag{7}$$

where $|\mathbf{S}_{\alpha_i}|$ denotes the cardinality of the set $\mathbf{S}_{\alpha_i}$.

It can be noted that the reward is the historical average improvement of $\alpha_i$. On the basis of the credit assignment, an empirical correction is performed in order to reward those search algorithms that recently led to improvements rather than those occurred in earlier stages of the search process, see [44, 15].

In order to achieve this aim, the quality index $q_{\alpha_i}(t)$ of the search algorithm $\alpha_i$ at the time $t$ is calculated by:

$$\begin{aligned} q_{\alpha_i}(t) &= r_{\alpha_i}(t) + \gamma(s_{\alpha_i}(t) - r_{\alpha_i}(t)) \\ &= (1-\gamma)\, r_{\alpha_i}(t) + \gamma s_{\alpha_i}(t) \end{aligned} \tag{8}$$

where $\gamma \in (0,1]$ is the adaptation rate which can amplify the influence of the most recent rewards over their history.

### 2.4.2. Adaptive Operator Selection

The quality index calculated in eq. (8) is used by AOS components to assign a selection probability to each search algorithm. In this study, we selected the following adaptive mechanisms.

*Random Selection:.* Random Selection (RS) is a simple sampling procedure that draw values from a uniformly random distribution, as such, the two search algorithms have the same probability (0.5) of being selected throughout the algorithm [1, 26]. Random Selection is used here as a control mechanism in order to investigate the effectiveness of the adaptive schemes.The algorithm employing the random selection is here referred to as hyperSPAM-RS.

*Probability Matching:*. Probability Matching (PM) [44] is a probabilistic model that assigns to each operator a probability proportional to its empirical quality, while respecting the remaining operators. Specifically, given a set of $\kappa$ available operators $A = \{\alpha_1, \alpha_2, \ldots, \alpha_\kappa\}$ and a probability vector $\mathbf{p}(t) = \{p_{\alpha_1}(t), p_{\alpha_2}(t), \ldots, p_{\alpha_\kappa}(t)\}$ of the selection probabilities for all operators, where initially all probabilities are equal $(p_{\alpha_i}(t) = 1/\kappa, \forall \alpha_i \in \mathbf{A})$.

PM uses the following update rule rule to adapt the activation probabilities of each search algorithm:

$$p_{\alpha_i}(t) = p_{\min} + (1 - \kappa \cdot p_{\min}) \frac{q_{\alpha_i}(t)}{\sum_{i=1}^{\kappa} q_{\alpha_i}(t)} \tag{9}$$

where $p_{\min} \in [0, 1]$ is the minimal probability value of each operator, which ensures that the application of each operator will not cease throughout the search process [44] and $q_{\alpha_i}(t+1)$ is the quality index of $\alpha_i$ at the time $t+1$ calculated in eq. (8).

The algorithm employing this coordination mechanism is here referred to as hyperSPAM-PM.

*Adaptive Pursuit:*. Adaptive Pursuit (AP) is a simple probabilistic model that follows a winner-takes-all strategy [44]. It attempts to address PM's drawback by increasing the probability of the best search algorithm ($\alpha_{i^\star}$), while simultaneously decreasing the probabilities of the remaining operators, according to the following formulas:

$$\alpha_{i^\star} = \arg \max_{i=1,2,\ldots,\kappa} \{q_{\alpha_i}(t+1)\} \tag{10}$$

$$p_{\alpha_i}(t+1) = \begin{cases} p_{\alpha_i}(t) + \beta(p_{\max} - p_{\alpha_i}(t)), & \text{if } \alpha_i = \alpha_{i^\star} \\ p_{\alpha_i}(t) + \beta(p_{\min} - p_{\alpha_i}(t)), & \text{otherwise} \end{cases}. \tag{11}$$

where $\beta \in [0, 1]$ is a user-defined parameter (i.e., learning rate) that amplifies the greediness of the winner-takes-all strategy. Higher values of $\beta$ strongly favor the best operator.

The algorithm employing this coordination mechanism is here referred to as hyperSPAM-AP.

*Multinomial Distribution Tracking:*. Multinomial distribution tracking (MT) makes uses of the multinomial distribution with an exponential smoothing mechanism to model an adaptive operator selection mechanism. The underpinning idea of this adaptive scheme is that the selection probabilities (i.e., distribution parameters) quickly react to the changes in performance of the search algorithms [12, 9], whilst it utilizes a forgetting mechanism to amplify recent observations and forget historical ones.

More specifically, Multinomial distribution tracking (MT) utilizes a Recursive Least Squares adaptive filter with an exponential weighted factor on the multinomial distribution. It essentially incorporates an exponential weighted factor (also known as *forgetting factor* $\lambda \in [0,1]$) in the log-likelihood of a multinomial distribution to discount the impact of past observations and enable adaptation of the estimated parameter values.

The parameters of the multinomial distribution are estimated through a Maximum Likelihood Estimator of the new log-likelihood. As such, the selection probability of each action $p_{\alpha_i}(t+1) = \hat{\theta}^{ML}_{\alpha_i}$ can be estimated according to the following formulas:

$$p_{\alpha_i}(t+1) = \frac{n_{\alpha_i}(t)}{\sum_{i=1}^{\kappa} n_{\alpha_i}(t)} \tag{12}$$

$$n_{\alpha_i}(t) = n_{\alpha_i}(t-1) + D_{\alpha_i}(t), \tag{13}$$

for $t = 1, 2, \cdots$ and $n_{\alpha_i}(0) = 0$, where $n_{\alpha_i}(t)$ represents the effective window width and $D_{\alpha_i}(t)$ indicates the number of successes of each action $\alpha_i$ at time $t$.

To calculate the number of successes $D_{\alpha_i}(t)$, a transformation of the empirical quantity $q_{\alpha_i}(t)$ of each action $\alpha_i$ to an integer number has to be performed to comply with the multinomial required input values. Thus, at each time step $t$, the proportion of the empirical quantity that is associated with each action is firstly calculated and then a number of total $c$ slots is proportionally assigned across all actions. The transformation of $D_{\alpha_i}(t)$ is calculated according to the following formula:

$$D_{\alpha_i}(t) = \left\lfloor c \cdot \frac{q_{\alpha_i}(t)}{\sum_{i=1}^{\kappa} q_{\alpha_i}(t)} \right\rfloor \tag{14}$$

The algorithm employing this coordination mechanism is here referred to as hyperSPAM-MT.

### 2.5. Replacement or re-sample

After the application of each search algorithm (S or R) the quality of the output solution $\mathbf{x}'$ is compared with the quality of the input solution $\mathbf{x}$. If the search algorithm improved upon the input solution, i.e. if $f(x') < f(x)$ then the output solution replaces the candidate solution $\mathbf{x}$ and again processed by the search algorithm selected by the coordination mechanism. If the search algorithm was ineffective and the coordination mechanism selects the same search algorithm for the following activation the re-sampling inspired by the exponential crossover of Differential Evolution and described in [6] is performed.

Let us indicate with $\mathbf{x}'$ the input candidate solution. One design variable index $j_{rand}$ is randomly selected and the corresponding design variable in $\mathbf{x}'$ is selected and copied into the output candidate solution $\mathbf{x}$. Then, contiguous design variables are copied, one by one, from $\mathbf{x}'$ into $\mathbf{x}$ until a random number is less than the crossover probability $Cr$. The remaining design variables of $\mathbf{x}$ are filled with random numbers within the corresponding range. The re-sample procedure is explained in Algorithm 3.

**Algorithm 3** Re-sampling of $\mathbf{x}$

INPUT $\mathbf{x}'$
generate a random solution $\mathbf{x}$ within the decision space
generate a random index $j_{rand}$ and assign $x_{j_{rand}} = x'_{j_{rand}}$
generate a random value $h$ from a uniform distribution $\mathscr{U}(0,1)$
$j = j_{rand} + 1$
**while** $h \leq Cr$ AND $j < n$ **do**
    $x_j = x'_j$
    **if** $j == n$ **then**
        $j = 1$
    **end if**
    $k = k + 1$
    generate a random value $h$ from a uniform distribution $\mathscr{U}(0,1)$
**end while**
OUTPUT $\mathbf{x}$

This procedure is applied to avoid that the same solution is processed twice by the same search algorithm after a failure.

In order to directly control the quota of design variables copied from $\mathbf{x}$ to $\mathbf{x}'$, let us introduce

$$q \approx \frac{n_m}{n}$$

where $n_m$ is the number of design variables we expect to copy into and $n$ is the total number of design variables (problem dimensionality). In order to achieve that on average $n_m$ are copied into $\mathbf{x}'$ we need to impose that

$$Cr^{nq} = 0.5.$$

By solving this equation, we set the parameter $Cr$ as

$$Cr = \frac{1}{\sqrt[nq]{2}}. \tag{15}$$

The general hyperSPAM framework that employs an AOS model (e.g., hyperSPAM-PM, hyperSPAM-AP, hyperSPAM-MT) is shown in Algorithm 4.

Fig. 1 depicts a graphical representation of the general hyperSPAM framework. The solid line represents the optimisation data flow while the dashed lines refer to the adaptation/control components. Below the Adaptive Operator Selection block we represent as an electric switch the activation of either S or R search algorithms.

## 3. Experimental Results

This section describes the experiments carried out, displays the results, and provide some comments about them. The experiments have been performed on benchmark tests as well as on a real-world problem of engineering design.

### *3.1. Benchmark Sets*

In order to extensively test the algorithms under examination we have tested them over the following benchmarks:

- CEC2013 benchmark [29] in 10, 30, and 50 variables.

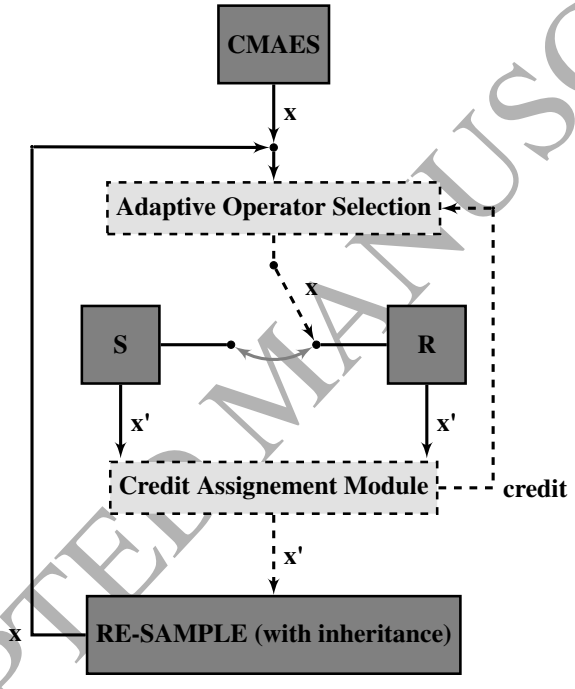


Figure 1: Graphical representation of the hyperSPAM framework

**Algorithm 4** The general hyperSPAM algorithmic framework

1: Perform the pre-processing phase by applying CMAES as shown in Section 2.1 and select the candidate solution $\mathbf{x}$ displaying the best performance.
2: Initialize the AOS model $\mathscr{M}$ and the credit assignment module $\mathscr{C}$.
3: **while** the remaining budget is available **do**
4: Select the search algorithm based on the AOS model $\mathscr{M}$
5: **if** S is selected **then**
6: Apply the S search algorithm to $\mathbf{x}$ and return $\mathbf{x}'$, see Subsection 2.2
7: **else**
8: Apply the R search algorithm to $\mathbf{x}$ and return $\mathbf{x}'$, see Subsection 2.3
9: **end if**
10: Update credit assignment module $\mathscr{C}$ and calculate $q_{\alpha_i}(t)$ for S and R
11: Update the AOS model $\mathscr{M}$ to select the following search algorithm
12: **if** the operator failed at improving upon $\mathbf{x}$ performance ($f(\mathbf{x}') \geq f(\mathbf{x})$) **and** the selected operator is the same that failed **then**
13: Resample $\mathbf{x}$ according to Algorithm 3, see Subsection 2.5
14: **else if** $f(\mathbf{x}') < f(\mathbf{x})$ **then**
15: $\mathbf{x} = \mathbf{x}'$
16: **end if**
17: **end while**

- BBOB2010 benchmark [32] in 100 variables.
- Real-world problems 1, 2, and 7 from CEC2011 [8], i.e. Parameter Estimation for Frequency-Modulated (FM) Sound Waves in six dimensions, Lennard-Jones Potential Problem in 30 dimensions, Spread Spectrum Radar Poly-phase Code Design, respectively.

### *3.2. Experimental Setup*

For the 111 problems under examination, four hyperSPAM implementations have been run. For all these implementations we set each algorithmic component with the following parameters:

- The CMAES pre-processing is parameterless e.g. $\lambda = 4 + 3\log(n)$ and $\mu = \frac{\lambda}{2}$, see [18] and [17];
- S is run with an initial step $\xi = 0.4\mathbf{d}$, where $\mathbf{d}$ is the variable range along one

variable of the decision space **D**, and a local computational budget of 1000 fitness functional calls;

- R is run with precision $\varepsilon = 10^{-5}$, see eq. (5), factor $\alpha = 2$, $\beta = 0.5$, and a local computational budget not exceeding 1000 fitness functional calls;
- The re-sampling has been run with $q = 0.5$, see eq. (15);
- The credit assignment is run with $\gamma = 0.1$ on the basis of the studies in [13, 44].

The parameters specific to the four implementations are the following:

- hyperSPAM-RS: parameterless;
- hyperSPAM-PM: $\alpha = 0.8$ and $p_{min} = 0.05$, see eq. (9);
- hyperSPAM-AP: $p_{min} = 0.05$, $\beta = 0.8$, see eq. (10);
- hyperSPAM-MT: $c = 1000$, $\alpha = 0.8$, $\lambda = 0.99$ and pmin $= 0.05$, see eq. (14) and [9].

Furthermore, the following popular metaheuristics have been included in the comparison:

- CMAES according to the implementation reported in [17]: parameterless and self-tuning;
- MDE-pBX as described in [25], i.e. population size 100, $q = 0.15$, $CR_m = 0.6$, $F_m = 0.5$, $N = 1.5$);
- CCPSO2 as described in [28]: population size 30, $p = 0.5$, number of divisions in 10 and 30 variables $\{2,5,10\}$, number of divisions in 50 and 100 variables $\{2,5,10,50,100\}$).

For each problem, each algorithm has been run for $5000 \times n$ fitness evaluations (function calls), where $n$ is the number of variables of the problem. Thirty independent runs have been performed for each algorithm on each problem.

For each problem, the best average function value is highlighted in bold. Furthermore, for each problem, a statistical pair-wise comparison has been performed by applying the Wilcoxon signed-rank test, see [47]. In all the result tables reported in this study, a $+$ indicates that hyperSPAM-MT significantly outperforms the competitor, a $-$ indicates that hyperSPAM-MT is outperformed, and a $=$ is shown when the performance of the two algorithms is statistically indistinguishable.

In order to further enhance the statistical significance of the numerical results the Holm-Bonferroni procedure, see [14], has been applied to perform a ranking of the algorithms under study with respect to the problems under consideration. A detailed description of the procedure including intermediate steps is reported in [6].

The numerical results displayed in this study have been structured in the following way:

1. At first we compared the four hyperSPAM variants under consideration over the testbed problems.
2. We selected the hyperSPAM implementation with the best performance and tested against the three above-mentioned competitor metaheuristics.
3. We displayed the result of the Holm-Bonferroni procedure summarizing the results for all the problems and algorithms included in this study.
4. We presented the results for the real-world problems.

### *3.3. Comparison among hyperSPAM variants*

The four variants of hyperSPAM have been tested and compared. Tables 1, 2, and 3 display the results on CEC2013 testbed in 10, 30, and 50 dimensions, respectively. Table 4 shows the results on BBOB2010 testbed in 100 dimensions.

Table 5 shows the ranking of the hyperSPAM implementations according to the Holm-Bonferroni procedure.

Numerical results show that the performance of the four HuperSPAM variants are, albeit similar, not identical. This can interpreted that the search algorithms composing the hybrid framework, e.g. hyperheuristic and memetic, is fundamental but their coordination can indeed make a difference.

Table 1: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for the four hyperSPAM implementations on CEC2013[29] in 10 dimensions.

| | hyperSPAM-MT | hyperSPAM-PM | | hyperSPAM-AP | | hyperSPAM-RS | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 0.00e+00$ | + | $0.00e+00 \pm 0.00e+00$ | + | $0.00e+00 \pm 0.00e+00$ | + |
| $f_2$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 1.36e-13$ | = | $8.09e-11 \pm 8.04e-10$ | = | $0.00e+00 \pm 0.00e+00$ | + |
| $f_3$ | $5.23e+03 \pm 5.11e+04$ | $\mathbf{1.31e+00 \pm 3.50e+00}$ | = | $3.92e+01 \pm 2.70e+02$ | = | $3.22e+00 \pm 1.34e+01$ | = |
| $f_4$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 0.00e+00$ | + | $0.00e+00 \pm 0.00e+00$ | + | $0.00e+00 \pm 0.00e+00$ | + |
| $f_5$ | $1.14e-13 \pm 8.95e-14$ | $1.14e-13 \pm 6.63e-14$ | = | $1.14e-13 \pm 7.01e-14$ | = | $\mathbf{1.14e-13 \pm 8.04e-14}$ | = |
| $f_6$ | $3.87e+00 \pm 4.70e+00$ | $4.11e+00 \pm 4.78e+00$ | = | $4.01e+00 \pm 4.76e+00$ | = | $\mathbf{3.55e+00 \pm 4.58e+00}$ | = |
| $f_7$ | $6.69e+01 \pm 8.24e+01$ | $6.04e+01 \pm 6.04e+01$ | = | $\mathbf{5.58e+01 \pm 5.17e+01}$ | = | $6.35e+01 \pm 2.06e+02$ | = |
| $f_8$ | $2.04e+01 \pm 1.17e-01$ | $\mathbf{2.04e+01 \pm 1.38e-01}$ | = | $2.04e+01 \pm 1.09e-01$ | = | $2.04e+01 \pm 1.30e-01$ | = |
| $f_9$ | $6.80e+00 \pm 1.72e+00$ | $\mathbf{6.73e+00 \pm 1.72e+00}$ | = | $6.98e+00 \pm 2.11e+00$ | = | $7.16e+00 \pm 1.88e+00$ | = |
| $f_{10}$ | $1.40e-02 \pm 1.28e-02$ | $1.48e-02 \pm 1.40e-02$ | = | $1.54e-02 \pm 1.35e-02$ | = | $\mathbf{1.36e-02 \pm 1.20e-02}$ | = |
| $f_{11}$ | $\mathbf{5.79e+00 \pm 2.49e+00}$ | $6.30e+00 \pm 2.86e+00$ | = | $6.33e+00 \pm 2.45e+00$ | = | $6.89e+00 \pm 3.27e+00$ | + |
| $f_{12}$ | $\mathbf{1.77e+01 \pm 7.77e+00}$ | $1.80e+01 \pm 9.61e+00$ | = | $1.92e+01 \pm 9.48e+00$ | = | $2.12e+01 \pm 1.11e+01$ | = |
| $f_{13}$ | $3.83e+01 \pm 1.44e+01$ | $3.55e+01 \pm 1.52e+01$ | = | $\mathbf{3.34e+01 \pm 1.51e+01}$ | - | $3.90e+01 \pm 1.69e+01$ | = |
| $f_{14}$ | $2.40e+02 \pm 1.36e+02$ | $\mathbf{2.18e+02 \pm 1.05e+02}$ | = | $2.39e+02 \pm 1.21e+02$ | = | $2.62e+02 \pm 1.24e+02$ | = |
| $f_{15}$ | $1.01e+03 \pm 3.21e+02$ | $\mathbf{1.00e+03 \pm 3.65e+02}$ | = | $1.06e+03 \pm 3.21e+02$ | = | $1.06e+03 \pm 2.91e+02$ | = |
| $f_{16}$ | $\mathbf{2.47e-01 \pm 1.41e-01}$ | $2.92e-01 \pm 1.98e-01$ | = | $2.92e-01 \pm 1.72e-01$ | = | $2.79e-01 \pm 1.57e-01$ | = |
| $f_{17}$ | $1.54e+01 \pm 3.72e+00$ | $1.58e+01 \pm 4.28e+00$ | = | $1.58e+01 \pm 3.38e+00$ | = | $\mathbf{1.52e+01 \pm 3.38e+00}$ | = |
| $f_{18}$ | $4.03e+01 \pm 1.10e+01$ | $4.17e+01 \pm 1.42e+01$ | = | $\mathbf{3.95e+01 \pm 1.38e+01}$ | = | $4.54e+01 \pm 1.50e+01$ | + |
| $f_{19}$ | $\mathbf{7.06e-01 \pm 2.64e-01}$ | $7.33e-01 \pm 3.44e-01$ | = | $7.53e-01 \pm 3.25e-01$ | = | $7.47e-01 \pm 3.39e-01$ | = |
| $f_{20}$ | $4.18e+00 \pm 3.21e-01$ | $4.11e+00 \pm 3.64e-01$ | = | $\mathbf{4.10e+00 \pm 4.36e-01}$ | = | $4.16e+00 \pm 3.82e-01$ | = |
| $f_{21}$ | $3.01e+02 \pm 1.07e+02$ | $\mathbf{2.86e+02 \pm 1.26e+02}$ | = | $3.04e+02 \pm 1.15e+02$ | = | $2.95e+02 \pm 1.26e+02$ | = |
| $f_{22}$ | $\mathbf{3.29e+02 \pm 1.41e+02}$ | $3.34e+02 \pm 1.19e+02$ | = | $3.35e+02 \pm 1.42e+02$ | = | $3.63e+02 \pm 1.45e+02$ | = |
| $f_{23}$ | $\mathbf{1.31e+03 \pm 3.15e+02}$ | $1.52e+03 \pm 4.16e+02$ | + | $1.43e+03 \pm 4.01e+02$ | + | $1.51e+03 \pm 3.52e+02$ | + |
| $f_{24}$ | $2.02e+02 \pm 3.77e+01$ | $1.93e+02 \pm 4.46e+01$ | - | $\mathbf{1.87e+02 \pm 4.48e+01}$ | - | $1.88e+02 \pm 4.26e+01$ | - |
| $f_{25}$ | $2.18e+02 \pm 1.13e+01$ | $\mathbf{2.15e+02 \pm 1.90e+01}$ | = | $2.16e+02 \pm 1.67e+01$ | = | $2.16e+02 \pm 1.81e+01$ | = |
| $f_{26}$ | $1.69e+02 \pm 5.85e+01$ | $\mathbf{1.67e+02 \pm 6.02e+01}$ | = | $1.74e+02 \pm 6.16e+01$ | = | $1.79e+02 \pm 6.67e+01$ | = |
| $f_{27}$ | $\mathbf{3.74e+02 \pm 7.30e+01}$ | $3.81e+02 \pm 7.32e+01$ | = | $3.83e+02 \pm 8.27e+01$ | = | $3.74e+02 \pm 6.43e+01$ | = |
| $f_{28}$ | $3.05e+02 \pm 1.62e+02$ | $\mathbf{2.92e+02 \pm 9.79e+01}$ | = | $3.60e+02 \pm 4.72e+02$ | = | $3.20e+02 \pm 1.44e+02$ | = |

Table 2: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for the four hyperSPAM implementations on CEC2013[29] in 30 dimensions.

| | hyperSPAM-MT | hyperSPAM-PM | | hyperSPAM-AP | | hyperSPAM-RS | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $0.00e+00 \pm 2.02e-13$ | $0.00e+00 \pm 2.05e-13$ | = | $0.00e+00 \pm 2.11e-13$ | = | $\mathbf{0.00e+00 \pm 2.01e-13}$ | = |
| $f_2$ | $1.49e+03 \pm 1.12e+03$ | $1.75e+03 \pm 1.90e+03$ | = | $\mathbf{1.40e+03 \pm 1.28e+03}$ | = | $1.56e+03 \pm 1.22e+03$ | = |
| $f_3$ | $1.17e+06 \pm 3.41e+06$ | $\mathbf{7.88e+05 \pm 1.79e+06}$ | = | $1.93e+06 \pm 5.74e+06$ | = | $8.86e+05 \pm 1.80e+06$ | = |
| $f_4$ | $3.13e-04 \pm 2.45e-03$ | $5.16e-04 \pm 4.54e-03$ | = | $\mathbf{2.48e-05 \pm 8.03e-05}$ | = | $5.12e-05 \pm 1.88e-04$ | = |
| $f_5$ | $1.14e-13 \pm 5.47e-13$ | $1.14e-13 \pm 6.49e-13$ | + | $1.14e-13 \pm 5.85e-13$ | = | $\mathbf{1.14e-13 \pm 5.22e-13}$ | = |
| $f_6$ | $1.77e-01 \pm 8.74e-01$ | $\mathbf{9.73e-02 \pm 4.74e-01}$ | = | $2.85e-01 \pm 2.11e+00$ | = | $2.40e-01 \pm 1.32e+00$ | = |
| $f_7$ | $5.12e+01 \pm 3.57e+01$ | $4.00e+01 \pm 2.58e+01$ | - | $\mathbf{3.73e+01 \pm 2.67e+01}$ | - | $4.93e+01 \pm 3.91e+01$ | = |
| $f_8$ | $\mathbf{2.09e+01 \pm 9.03e-02}$ | $2.09e+01 \pm 7.13e-02$ | = | $2.10e+01 \pm 6.87e-02$ | + | $2.09e+01 \pm 7.62e-02$ | = |
| $f_9$ | $3.07e+01 \pm 4.10e+00$ | $\mathbf{3.00e+01 \pm 3.58e+00}$ | = | $3.07e+01 \pm 3.90e+00$ | = | $3.06e+01 \pm 4.25e+00$ | = |
| $f_{10}$ | $1.29e-02 \pm 8.12e-03$ | $1.28e-02 \pm 7.93e-03$ | = | $\mathbf{1.06e-02 \pm 7.31e-03}$ | - | $1.15e-02 \pm 7.34e-03$ | = |
| $f_{11}$ | $2.91e+01 \pm 5.90e+00$ | $2.94e+01 \pm 6.43e+00$ | = | $2.89e+01 \pm 6.94e+00$ | = | $\mathbf{2.78e+01 \pm 6.47e+00}$ | = |
| $f_{12}$ | $\mathbf{7.96e+01 \pm 5.27e+01}$ | $9.80e+01 \pm 6.27e+01$ | = | $8.82e+01 \pm 5.62e+01$ | = | $8.69e+01 \pm 5.61e+01$ | = |
| $f_{13}$ | $1.99e+02 \pm 7.06e+01$ | $1.99e+02 \pm 6.82e+01$ | = | $\mathbf{1.82e+02 \pm 7.35e+01}$ | = | $1.90e+02 \pm 7.65e+01$ | = |
| $f_{14}$ | $8.04e+02 \pm 2.05e+02$ | $\mathbf{7.83e+02 \pm 2.03e+02}$ | = | $7.99e+02 \pm 2.09e+02$ | = | $8.40e+02 \pm 2.24e+02$ | = |
| $f_{15}$ | $\mathbf{3.81e+03 \pm 7.13e+02}$ | $4.78e+03 \pm 7.59e+02$ | + | $3.97e+03 \pm 6.51e+02$ | = | $4.59e+03 \pm 7.60e+02$ | + |
| $f_{16}$ | $1.48e-01 \pm 9.27e-02$ | $1.42e-01 \pm 1.23e-01$ | = | $1.52e-01 \pm 1.09e-01$ | = | $\mathbf{1.37e-01 \pm 9.61e-02}$ | = |
| $f_{17}$ | $5.59e+01 \pm 9.02e+00$ | $5.71e+01 \pm 7.70e+00$ | = | $\mathbf{5.43e+01 \pm 6.86e+00}$ | = | $5.49e+01 \pm 7.69e+00$ | = |
| $f_{18}$ | $2.36e+02 \pm 4.78e+01$ | $2.37e+02 \pm 5.44e+01$ | = | $\mathbf{2.32e+02 \pm 5.62e+01}$ | = | $2.47e+02 \pm 5.94e+01$ | = |
| $f_{19}$ | $2.63e+00 \pm 6.76e-01$ | $2.64e+00 \pm 6.63e-01$ | = | $\mathbf{2.58e+00 \pm 6.90e-01}$ | = | $2.66e+00 \pm 6.84e-01$ | = |
| $f_{20}$ | $1.44e+01 \pm 5.53e-01$ | $1.45e+01 \pm 5.14e-01$ | = | $\mathbf{1.44e+01 \pm 6.31e-01}$ | = | $1.45e+01 \pm 5.60e-01$ | = |
| $f_{21}$ | $\mathbf{2.31e+02 \pm 5.04e+01}$ | $2.40e+02 \pm 5.62e+01$ | = | $2.38e+02 \pm 6.79e+01$ | = | $2.38e+02 \pm 6.37e+01$ | = |
| $f_{22}$ | $1.10e+03 \pm 2.96e+02$ | $1.08e+03 \pm 3.02e+02$ | = | $\mathbf{1.04e+03 \pm 2.45e+02}$ | = | $1.06e+03 \pm 2.58e+02$ | = |
| $f_{23}$ | $\mathbf{5.00e+03 \pm 7.59e+02}$ | $6.05e+03 \pm 9.37e+02$ | + | $5.07e+03 \pm 9.26e+02$ | = | $6.17e+03 \pm 9.22e+02$ | + |
| $f_{24}$ | $2.80e+02 \pm 9.99e+01$ | $3.00e+02 \pm 1.88e+02$ | = | $\mathbf{2.66e+02 \pm 3.30e+01}$ | = | $2.98e+02 \pm 1.43e+02$ | = |
| $f_{25}$ | $\mathbf{2.93e+02 \pm 1.55e+01}$ | $2.94e+02 \pm 1.84e+01$ | = | $2.94e+02 \pm 1.52e+01$ | = | $2.94e+02 \pm 1.45e+01$ | = |
| $f_{26}$ | $2.84e+02 \pm 8.59e+01$ | $2.80e+02 \pm 8.66e+01$ | = | $2.86e+02 \pm 8.24e+01$ | = | $\mathbf{2.79e+02 \pm 8.38e+01}$ | = |
| $f_{27}$ | $8.21e+02 \pm 1.64e+02$ | $8.27e+02 \pm 1.92e+02$ | = | $8.30e+02 \pm 1.79e+02$ | = | $\mathbf{8.14e+02 \pm 1.95e+02}$ | = |
| $f_{28}$ | $7.86e+02 \pm 1.82e+03$ | $6.27e+02 \pm 1.36e+03$ | = | $6.78e+02 \pm 1.36e+03$ | = | $\mathbf{6.06e+02 \pm 1.28e+03}$ | = |

Table 3: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for the four hyperSPAM implementations on CEC2013[29] in 50 dimensions.

| | hyperSPAM-MT | hyperSPAM-PM | | hyperSPAM-AP | | hyperSPAM-RS | |
|---|---|---|---|---|---|---|---|
| $f_1$ | **2.27e−13 ± 0.00e+00** | 2.27e−13 ± 0.00e+00 | + | 2.27e−13 ± 0.00e+00 | + | 2.27e−13 ± 0.00e+00 | + |
| $f_2$ | **2.59e+04 ± 1.28e+04** | 2.66e+04 ± 1.35e+04 | = | 2.68e+04 ± 1.17e+04 | = | 2.77e+04 ± 1.30e+04 | = |
| $f_3$ | **6.19e+06 ± 8.61e+06** | 8.55e+06 ± 1.43e+07 | = | 6.98e+06 ± 9.27e+06 | = | 7.48e+06 ± 9.35e+06 | = |
| $f_4$ | 4.85e+02 ± 4.85e+02 | 5.59e+02 ± 5.01e+02 | = | **4.39e+02 ± 5.21e+02** | = | 5.07e+02 ± 4.66e+02 | = |
| $f_5$ | **3.41e−13 ± 8.70e−13** | 3.41e−13 ± 1.05e−12 | + | 3.41e−13 ± 9.29e−13 | = | 3.41e−13 ± 9.02e−13 | = |
| $f_6$ | 3.14e+01 ± 1.80e+01 | 2.74e+01 ± 1.75e+01 | = | **2.69e+01 ± 1.77e+01** | = | 2.99e+01 ± 1.72e+01 | = |
| $f_7$ | **4.46e+01 ± 1.93e+01** | 4.75e+01 ± 2.38e+01 | = | 4.84e+01 ± 2.13e+01 | = | 4.66e+01 ± 2.16e+01 | = |
| $f_8$ | 2.11e+01 ± 6.39e−02 | 2.11e+01 ± 6.66e−02 | - | 2.11e+01 ± 6.62e−02 | - | **2.11e+01 ± 6.08e−02** | - |
| $f_9$ | 5.74e+01 ± 4.96e+00 | 5.49e+01 ± 5.03e+00 | - | 5.45e+01 ± 4.79e+00 | - | **5.07e+01 ± 3.96e+00** | - |
| $f_{10}$ | 1.15e−02 ± 7.41e−03 | 1.24e−02 ± 7.26e−03 | = | **1.07e−02 ± 6.19e−03** | = | 1.14e−02 ± 7.13e−03 | = |
| $f_{11}$ | 5.77e+01 ± 9.97e+00 | 5.87e+01 ± 1.05e+01 | = | **5.48e+01 ± 1.06e+01** | = | 5.70e+01 ± 1.08e+01 | = |
| $f_{12}$ | 3.09e+02 ± 1.37e+02 | 2.98e+02 ± 1.43e+02 | = | **2.88e+02 ± 1.33e+02** | = | 3.49e+02 ± 1.48e+02 | + |
| $f_{13}$ | 5.39e+02 ± 1.01e+02 | 5.33e+02 ± 1.12e+02 | = | **5.06e+02 ± 1.25e+02** | = | 5.34e+02 ± 1.04e+02 | = |
| $f_{14}$ | 1.43e+03 ± 3.09e+02 | 1.41e+03 ± 3.09e+02 | = | **1.40e+03 ± 3.13e+02** | = | 1.41e+03 ± 2.80e+02 | = |
| $f_{15}$ | **6.96e+03 ± 6.72e+02** | 8.50e+03 ± 1.09e+03 | + | 7.30e+03 ± 1.06e+03 | + | 8.16e+03 ± 1.14e+03 | + |
| $f_{16}$ | 8.26e−02 ± 4.79e−02 | 8.36e−02 ± 3.88e−02 | = | **8.10e−02 ± 4.18e−02** | = | 8.61e−02 ± 4.76e−02 | = |
| $f_{17}$ | 9.70e+01 ± 1.04e+01 | **9.62e+01 ± 1.11e+01** | = | 9.71e+01 ± 1.06e+01 | = | 9.81e+01 ± 7.86e+00 | = |
| $f_{18}$ | **5.18e+02 ± 8.96e+01** | 5.37e+02 ± 9.93e+01 | = | 5.27e+02 ± 8.95e+01 | = | 5.83e+02 ± 1.49e+02 | + |
| $f_{19}$ | **4.61e+00 ± 9.69e−01** | 4.73e+00 ± 8.92e−01 | = | 4.68e+00 ± 9.90e−01 | = | 4.67e+00 ± 8.84e−01 | = |
| $f_{20}$ | **2.43e+01 ± 5.08e−01** | 2.44e+01 ± 2.86e−01 | = | 2.43e+01 ± 5.25e−01 | = | 2.44e+01 ± 3.24e−01 | = |
| $f_{21}$ | 4.58e+02 ± 3.52e+02 | 4.27e+02 ± 3.30e+02 | = | **4.25e+02 ± 3.60e+02** | = | 5.59e+02 ± 3.88e+02 | + |
| $f_{22}$ | 2.06e+03 ± 4.02e+02 | 2.06e+03 ± 3.17e+02 | = | 2.02e+03 ± 3.78e+02 | = | **2.01e+03 ± 3.98e+02** | = |
| $f_{23}$ | **9.61e+03 ± 1.18e+03** | 1.12e+04 ± 1.27e+03 | + | 9.78e+03 ± 1.41e+03 | = | 1.07e+04 ± 1.20e+03 | + |
| $f_{24}$ | **3.39e+02 ± 3.14e+01** | 3.62e+02 ± 2.14e+02 | = | 3.55e+02 ± 1.60e+02 | = | 3.53e+02 ± 2.09e+02 | - |
| $f_{25}$ | 3.85e+02 ± 2.29e+01 | 3.79e+02 ± 1.99e+01 | = | 3.78e+02 ± 2.03e+01 | - | **3.71e+02 ± 1.45e+01** | - |
| $f_{26}$ | 3.10e+02 ± 1.10e+02 | 3.08e+02 ± 2.97e+02 | = | 2.94e+02 ± 1.05e+02 | = | **2.67e+02 ± 2.46e+02** | - |
| $f_{27}$ | 1.34e+03 ± 2.25e+02 | 1.28e+03 ± 2.33e+02 | = | **1.25e+03 ± 2.25e+02** | - | 1.26e+03 ± 2.32e+02 | - |
| $f_{28}$ | 1.69e+03 ± 3.42e+03 | 1.63e+03 ± 2.22e+03 | = | 1.74e+03 ± 2.72e+03 | = | **1.21e+03 ± 1.37e+03** | = |

Table 4: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for the four hyperSPAM implementations on BBOB2010 [32] in 100 dimensions.

| | hyperSPAM-MT | hyperSPAM-PM | | hyperSPAM-AP | | hyperSPAM-RS | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $2.42e-13 \pm 2.12e-13$ | $2.42e-13 \pm 2.12e-13$ | = | $\mathbf{2.42e-13 \pm 2.12e-13}$ | = | $2.42e-13 \pm 2.12e-13$ | = |
| $f_2$ | $1.42e-13 \pm 1.68e-13$ | $1.71e-13 \pm 1.55e-13$ | + | $1.71e-13 \pm 1.48e-13$ | = | $\mathbf{1.42e-13 \pm 1.58e-13}$ | = |
| $f_3$ | $\mathbf{1.01e+02 \pm 1.79e+01}$ | $1.04e+02 \pm 1.70e+01$ | = | $1.03e+02 \pm 1.86e+01$ | = | $1.04e+02 \pm 1.77e+01$ | = |
| $f_4$ | $1.37e+02 \pm 1.75e+01$ | $1.37e+02 \pm 2.01e+01$ | = | $1.37e+02 \pm 2.03e+01$ | = | $\mathbf{1.35e+02 \pm 2.22e+01}$ | = |
| $f_5$ | $\mathbf{1.26e-11 \pm 4.00e-12}$ | $4.93e-08 \pm 2.91e-07$ | + | $1.17e-06 \pm 7.08e-06$ | + | $1.28e-09 \pm 3.18e-09$ | + |
| $f_6$ | $\mathbf{3.52e-08 \pm 5.57e-08}$ | $6.15e-08 \pm 7.12e-08$ | + | $5.37e-08 \pm 7.12e-08$ | + | $8.19e-08 \pm 1.08e-07$ | + |
| $f_7$ | $\mathbf{5.23e+01 \pm 1.48e+01}$ | $5.26e+01 \pm 1.40e+01$ | = | $5.47e+01 \pm 1.24e+01$ | = | $5.41e+01 \pm 1.51e+01$ | = |
| $f_8$ | $\mathbf{3.56e+01 \pm 7.86e+00}$ | $3.66e+01 \pm 1.11e+01$ | = | $3.61e+01 \pm 1.06e+01$ | = | $3.65e+01 \pm 7.55e+00$ | + |
| $f_9$ | $4.46e+01 \pm 1.12e+01$ | $\mathbf{4.39e+01 \pm 7.26e+00}$ | = | $4.41e+01 \pm 8.56e+00$ | = | $4.58e+01 \pm 8.05e+00$ | + |
| $f_{10}$ | $5.21e+02 \pm 1.42e+02$ | $5.27e+02 \pm 1.50e+02$ | = | $\mathbf{5.13e+02 \pm 1.34e+02}$ | = | $5.45e+02 \pm 1.87e+02$ | = |
| $f_{11}$ | $8.38e+01 \pm 3.46e+01$ | $8.72e+01 \pm 3.08e+01$ | = | $8.34e+01 \pm 2.80e+01$ | = | $\mathbf{8.17e+01 \pm 2.25e+01}$ | = |
| $f_{12}$ | $4.79e-02 \pm 2.52e-01$ | $5.29e-02 \pm 1.98e-01$ | = | $\mathbf{1.19e-02 \pm 4.76e-02}$ | = | $1.65e-02 \pm 8.57e-02$ | = |
| $f_{13}$ | $1.07e+00 \pm 1.43e+00$ | $1.06e+00 \pm 1.24e+00$ | = | $1.25e+00 \pm 1.52e+00$ | = | $\mathbf{4.72e-01 \pm 7.02e-01}$ | - |
| $f_{14}$ | $\mathbf{4.87e-05 \pm 5.88e-06}$ | $5.07e-05 \pm 6.32e-06$ | + | $4.88e-05 \pm 6.95e-06$ | = | $5.01e-05 \pm 6.75e-06$ | = |
| $f_{15}$ | $2.71e+02 \pm 4.03e+01$ | $2.75e+02 \pm 4.48e+01$ | = | $2.75e+02 \pm 4.44e+01$ | = | $\mathbf{2.70e+02 \pm 4.09e+01}$ | = |
| $f_{16}$ | $2.37e+00 \pm 8.08e-01$ | $\mathbf{2.31e+00 \pm 8.27e-01}$ | = | $2.49e+00 \pm 7.90e-01$ | = | $2.43e+00 \pm 9.60e-01$ | = |
| $f_{17}$ | $\mathbf{8.45e+00 \pm 4.46e+00}$ | $8.55e+00 \pm 4.71e+00$ | = | $8.47e+00 \pm 4.50e+00$ | = | $8.66e+00 \pm 4.34e+00$ | = |
| $f_{18}$ | $1.90e+01 \pm 1.06e+01$ | $1.84e+01 \pm 1.15e+01$ | = | $\mathbf{1.75e+01 \pm 1.11e+01}$ | = | $1.81e+01 \pm 1.16e+01$ | = |
| $f_{19}$ | $\mathbf{1.82e+00 \pm 2.94e-01}$ | $1.96e+00 \pm 4.27e-01$ | + | $1.97e+00 \pm 3.79e-01$ | + | $2.04e+00 \pm 4.90e-01$ | + |
| $f_{20}$ | $1.19e+00 \pm 1.61e-01$ | $\mathbf{1.14e+00 \pm 1.09e-01}$ | = | $1.15e+00 \pm 1.17e-01$ | = | $1.16e+00 \pm 1.17e-01$ | = |
| $f_{21}$ | $\mathbf{3.74e+00 \pm 3.84e+00}$ | $3.89e+00 \pm 4.11e+00$ | = | $4.82e+00 \pm 5.66e+00$ | = | $4.40e+00 \pm 7.65e+00$ | = |
| $f_{22}$ | $6.75e+00 \pm 7.22e+00$ | $7.94e+00 \pm 9.28e+00$ | = | $\mathbf{6.10e+00 \pm 7.53e+00}$ | = | $6.36e+00 \pm 7.66e+00$ | = |
| $f_{23}$ | $7.69e-01 \pm 4.37e-01$ | $8.23e-01 \pm 3.93e-01$ | = | $\mathbf{7.67e-01 \pm 4.06e-01}$ | = | $9.24e-01 \pm 5.12e-01$ | + |
| $f_{24}$ | $3.11e+02 \pm 6.12e+01$ | $\mathbf{3.10e+02 \pm 6.43e+01}$ | = | $3.24e+02 \pm 5.42e+01$ | = | $3.16e+02 \pm 6.23e+01$ | = |

Table 5: Holm-Bonferroni procedure (reference: hyperSPAM-MT, Rank = 2.52e+00) ranking the hyperSPAM implementations

| $j$ | Optimizer | Rank | $z_j$ | $p_j$ | $\delta/j$ | Hypothesis |
|---|---|---|---|---|---|---|
| 1 | hyperSPAM-AP | 2.50e+00 | -1.36e-01 | 4.46e-01 | 5.00e-02 | Accepted |
| 2 | hyperSPAM-RS | 2.30e+00 | -1.63e+00 | 5.12e-02 | 2.50e-02 | Accepted |
| 3 | hyperSPAM-PM | 2.19e+00 | -2.38e+00 | 8.62e-03 | 1.67e-02 | Rejected |

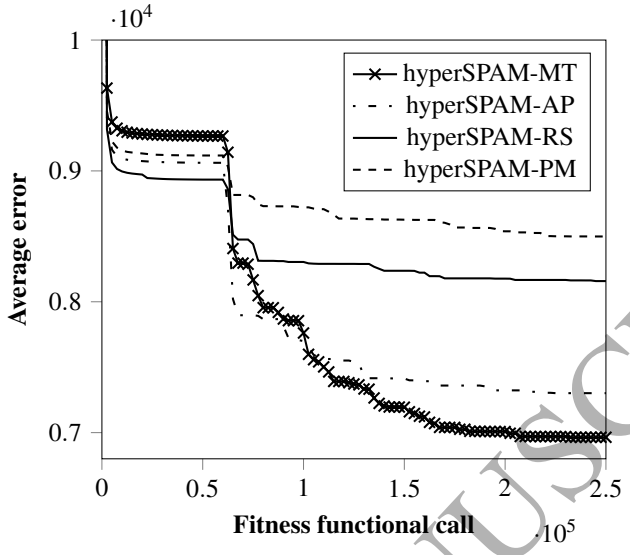


Figure 2: Average error trend of hyperSPAM-MT against the other hyperSPAM variants on $f_{15}$ of the CEC2013 beanchmark in 50 dimensions

Among the four hyperSPAM implementations, hyperSPAM-MT displays the best performance. This performance is significantly superior to that of hyperSPAM-PM and only marginally better that hyperSPAM-AP and hyperSPAM-RS.

It is worthwhile commenting the performance comparison between hyperSPAM-MT and hyperSPAM-RS. Indeed the sophisticated multinomial tracking appears to be, on average, beneficial to the algorithmic performance. However, a simple random selection by the uniform distribution function and without any adaptation is not much worse in this case. Analogous results were found in the context of Differential Evolution in [38]. As a conjecture, we think that similar results are achieved since a pool of only two search algorithms is considered in this study. A longer list of search algorithms where some appear preferable to others might dynamically skew the multinomial distribution thus making it remarkably different to the uniform distribution used by hyperSPAM-RS.

Fig. 2 displays the average error trend for the four hyperSPAM variants and shows that hyperSPAM-MT outperforms the other hyperSPAM schemes.

### 3.4. Comparison against popular metaheuristics

Since hyperSPAM implementation with Multinomial Distribution Tracking, hyperSPAM-MT, displayed the best performance, it has been compared against the popular metaheuristics considered in this studty. Tables 6, 7, and 8 display the results on CEC2013 testbed in 10, 30, and 50 dimensions, respectively. Table 9 shows the results on BBOB2010 testbed in 100 dimensions.

Table 10 summarises the statistically significant wins, draws, and losses in each testbed and in total. Table 11 shows the ranking of the hyperSPAM-MT and metaheuristics included in this study according to the Holm-Bonferroni procedure.

Numerical results show that hyperSPAM-MT significantly displays the best performance, thus demonstrating that the hyperSPAM framework is indeed effective. The second best performance is displayed by MDE-pBX which achieves very good results on a number of problems in 10 dimensions. However the MDE-pBX does not seem to address problems in higher dimensions as effectively as hyperSPAM-MT. Furthermore, the comparison against CMAES shows that CMAES displays the best performance on some problems and a poorer performance on the remaining problems. In particular the performance of CMAES is excellent for ill conditioned problems but appears to degrade when multiple local optima are present in the decision space.

Fig. 3 displays an example of average error trend of hyperSPAM-MT against the metaheuristics considered in this study. In order to enhance the readability of the results, the trends are shown on linear and logarithmic scales.

### 3.5. Ranking of all the algorithms

The seven algorithms in this study have been ranked according to the Holm-Bonferroni procedure with respect to all the CEC2013 and BBOB2010 problems and to all the dimensionality values under consideration. The results of the ranking are displayed in Table 12.

The results of the Holm-Bonferroni procedure show that the hyperSPAM framework, regardless of the coordination mechanism, significantly outperforms the competitor algorithms.

Table 6: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2 on CEC2013[29] in 10 dimensions.

| | hyperSPAM-MT | CMAES | | MDE-pBX | | CCPSO2 | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 0.00e+00$ | = | $0.00e+00 \pm 3.22e-14$ | = | $2.78e-03 \pm 9.28e-03$ | + |
| $f_2$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 0.00e+00$ | = | $2.58e+03 \pm 5.12e+03$ | + | $1.86e+06 \pm 1.22e+06$ | + |
| $f_3$ | $5.23e+03 \pm 5.11e+04$ | $\mathbf{1.54e-01 \pm 8.99e-01}$ | - | $9.14e+03 \pm 3.98e+04$ | + | $7.20e+07 \pm 1.10e+08$ | + |
| $f_4$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | $0.00e+00 \pm 0.00e+00$ | + | $6.64e-01 \pm 3.14e+00$ | + | $1.02e+04 \pm 2.59e+03$ | + |
| $f_5$ | $1.14e-13 \pm 8.95e-14$ | $\mathbf{0.00e+00 \pm 0.00e+00}$ | - | $0.00e+00 \pm 7.01e-14$ | - | $1.04e-02 \pm 2.37e-02$ | + |
| $f_6$ | $3.87e+00 \pm 4.70e+00$ | $6.31e+00 \pm 8.56e+00$ | = | $5.51e+00 \pm 4.86e+00$ | = | $\mathbf{3.81e+00 \pm 4.09e+00}$ | + |
| $f_7$ | $6.69e+01 \pm 8.24e+01$ | $1.27e+14 \pm 8.89e+14$ | = | $\mathbf{6.70e+00 \pm 9.32e+00}$ | - | $3.77e+01 \pm 1.27e+01$ | = |
| $f_8$ | $2.04e+01 \pm 1.17e-01$ | $\mathbf{2.04e+01 \pm 1.17e-01}$ | = | $2.05e+01 \pm 9.89e-02$ | + | $2.04e+01 \pm 8.01e-02$ | = |
| $f_9$ | $6.80e+00 \pm 1.72e+00$ | $1.49e+01 \pm 3.90e+00$ | + | $\mathbf{2.37e+00 \pm 1.57e+00}$ | - | $5.64e+00 \pm 7.38e-01$ | - |
| $f_{10}$ | $\mathbf{1.40e-02 \pm 1.28e-02}$ | $1.89e-02 \pm 1.40e-02$ | + | $1.07e-01 \pm 7.65e-02$ | + | $1.96e+00 \pm 9.60e-01$ | + |
| $f_{11}$ | $5.79e+00 \pm 2.49e+00$ | $2.11e+02 \pm 2.76e+02$ | + | $3.00e+00 \pm 1.98e+00$ | - | $\mathbf{2.70e+00 \pm 1.86e+00}$ | - |
| $f_{12}$ | $1.77e+01 \pm 7.77e+00$ | $3.47e+02 \pm 3.28e+02$ | + | $\mathbf{9.83e+00 \pm 3.94e+00}$ | - | $3.45e+01 \pm 9.16e+00$ | + |
| $f_{13}$ | $3.83e+01 \pm 1.44e+01$ | $2.42e+02 \pm 2.93e+02$ | + | $\mathbf{2.08e+01 \pm 9.71e+00}$ | - | $4.16e+01 \pm 9.15e+00$ | = |
| $f_{14}$ | $2.40e+02 \pm 1.36e+02$ | $1.76e+03 \pm 4.07e+02$ | + | $1.19e+02 \pm 1.02e+02$ | - | $\mathbf{8.89e+01 \pm 6.43e+01}$ | - |
| $f_{15}$ | $1.01e+03 \pm 3.21e+02$ | $1.78e+03 \pm 3.79e+02$ | + | $\mathbf{7.71e+02 \pm 2.45e+02}$ | - | $1.05e+03 \pm 2.89e+02$ | = |
| $f_{16}$ | $\mathbf{2.47e-01 \pm 1.41e-01}$ | $4.12e-01 \pm 3.44e-01$ | + | $5.98e-01 \pm 4.43e-01$ | + | $1.33e+00 \pm 2.26e-01$ | + |
| $f_{17}$ | $1.54e+01 \pm 3.72e+00$ | $9.96e+02 \pm 3.02e+02$ | + | $\mathbf{1.29e+01 \pm 1.69e+00}$ | - | $1.81e+01 \pm 2.90e+00$ | + |
| $f_{18}$ | $4.03e+01 \pm 1.10e+01$ | $1.01e+03 \pm 2.96e+02$ | + | $\mathbf{2.02e+01 \pm 4.69e+00}$ | - | $5.82e+01 \pm 6.20e+00$ | + |
| $f_{19}$ | $7.06e-01 \pm 2.64e-01$ | $1.14e+00 \pm 4.41e-01$ | + | $\mathbf{6.67e-01 \pm 2.22e-01}$ | = | $9.62e-01 \pm 4.13e-01$ | + |
| $f_{20}$ | $4.18e+00 \pm 3.21e-01$ | $4.79e+00 \pm 2.69e-01$ | + | $\mathbf{2.71e+00 \pm 6.46e-01}$ | - | $3.60e+00 \pm 2.12e-01$ | - |
| $f_{21}$ | $\mathbf{3.01e+02 \pm 1.07e+02}$ | $3.88e+02 \pm 4.75e+01$ | + | $3.96e+02 \pm 2.80e+01$ | + | $3.71e+02 \pm 6.00e+01$ | + |
| $f_{22}$ | $3.29e+02 \pm 1.41e+02$ | $2.29e+03 \pm 3.89e+02$ | + | $1.63e+02 \pm 1.35e+02$ | - | $\mathbf{1.25e+02 \pm 6.30e+01}$ | - |
| $f_{23}$ | $1.31e+03 \pm 3.15e+02$ | $2.17e+03 \pm 4.46e+02$ | + | $\mathbf{8.42e+02 \pm 3.20e+02}$ | - | $1.40e+03 \pm 2.87e+02$ | = |
| $f_{24}$ | $\mathbf{2.02e+02 \pm 3.77e+01}$ | $3.56e+02 \pm 1.22e+02$ | + | $2.05e+02 \pm 5.23e+00$ | - | $2.12e+02 \pm 1.77e+01$ | = |
| $f_{25}$ | $2.18e+02 \pm 1.13e+01$ | $2.56e+02 \pm 3.92e+01$ | + | $\mathbf{2.01e+02 \pm 2.97e+00}$ | - | $2.14e+02 \pm 8.51e+00$ | - |
| $f_{26}$ | $1.69e+02 \pm 5.85e+01$ | $2.75e+02 \pm 1.21e+02$ | + | $\mathbf{1.45e+02 \pm 4.34e+01}$ | - | $1.70e+02 \pm 2.35e+01$ | = |
| $f_{27}$ | $3.74e+02 \pm 7.30e+01$ | $3.93e+02 \pm 9.43e+01$ | = | $\mathbf{3.03e+02 \pm 1.43e+01}$ | - | $4.22e+02 \pm 5.01e+01$ | + |
| $f_{28}$ | $3.05e+02 \pm 1.62e+02$ | $1.43e+03 \pm 1.22e+03$ | + | $\mathbf{3.01e+02 \pm 4.80e+01}$ | - | $3.85e+02 \pm 1.49e+02$ | + |

Table 7: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2 on CEC2013[29] in 30 dimensions.

| | hyperSPAM-MT | CMAES | | MDE-pBX | | CCPSO2 | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $0.00e+00 \pm 2.02e-13$ | $\mathbf{0.00e+00 \pm 1.33e-13}$ | - | $2.27e-13 \pm 2.86e-13$ | + | $2.73e-12 \pm 8.10e-12$ | + |
| $f_2$ | $1.49e+03 \pm 1.12e+03$ | $\mathbf{0.00e+00 \pm 1.67e-13}$ | - | $2.85e+05 \pm 3.07e+05$ | + | $2.20e+06 \pm 1.11e+06$ | + |
| $f_3$ | $1.17e+06 \pm 3.41e+06$ | $\mathbf{9.75e+01 \pm 3.99e+02}$ | - | $3.61e+07 \pm 5.99e+07$ | + | $1.22e+09 \pm 1.21e+09$ | + |
| $f_4$ | $3.13e-04 \pm 2.45e-03$ | $\mathbf{0.00e+00 \pm 1.25e-13}$ | - | $3.49e+02 \pm 3.71e+02$ | + | $5.73e+04 \pm 1.66e+04$ | + |
| $f_5$ | $\mathbf{1.14e-13 \pm 5.47e-13}$ | $9.09e-13 \pm 2.17e-12$ | - | $2.02e-10 \pm 1.41e-09$ | - | $1.32e-07 \pm 3.04e-07$ | + |
| $f_6$ | $\mathbf{1.77e-01 \pm 8.74e-01}$ | $6.26e+00 \pm 1.58e+01$ | - | $3.31e+01 \pm 2.77e+01$ | + | $3.76e+01 \pm 2.85e+01$ | + |
| $f_7$ | $\mathbf{5.12e+01 \pm 3.57e+01}$ | $2.61e+05 \pm 1.35e+06$ | = | $5.57e+01 \pm 1.91e+01$ | + | $1.17e+02 \pm 2.37e+01$ | + |
| $f_8$ | $\mathbf{2.09e+01 \pm 9.03e-02}$ | $2.10e+01 \pm 5.61e-02$ | + | $2.10e+01 \pm 5.95e-02$ | + | $2.10e+01 \pm 5.91e-02$ | + |
| $f_9$ | $3.07e+01 \pm 4.10e+00$ | $4.42e+01 \pm 7.80e+00$ | + | $\mathbf{2.08e+01 \pm 4.29e+00}$ | - | $3.05e+01 \pm 2.07e+00$ | = |
| $f_{10}$ | $\mathbf{1.29e-02 \pm 8.12e-03}$ | $2.03e-02 \pm 1.44e-02$ | + | $1.94e-01 \pm 1.21e-01$ | + | $2.09e-01 \pm 9.15e-02$ | + |
| $f_{11}$ | $2.91e+01 \pm 5.90e+00$ | $6.81e+01 \pm 9.69e+01$ | + | $4.80e+01 \pm 1.48e+01$ | + | $\mathbf{5.37e-01 \pm 6.30e-01}$ | - |
| $f_{12}$ | $7.96e+01 \pm 5.27e+01$ | $8.09e+02 \pm 9.34e+02$ | + | $\mathbf{6.81e+01 \pm 2.37e+01}$ | = | $2.18e+02 \pm 5.39e+01$ | + |
| $f_{13}$ | $1.99e+02 \pm 7.06e+01$ | $1.73e+03 \pm 1.66e+03$ | + | $\mathbf{1.52e+02 \pm 3.37e+01}$ | - | $2.62e+02 \pm 4.71e+01$ | + |
| $f_{14}$ | $8.04e+02 \pm 2.05e+02$ | $5.36e+03 \pm 7.30e+02$ | + | $1.18e+03 \pm 4.14e+02$ | + | $\mathbf{6.96e+00 \pm 3.52e+00}$ | - |
| $f_{15}$ | $\mathbf{3.81e+03 \pm 7.13e+02}$ | $5.39e+03 \pm 6.21e+02$ | + | $4.03e+03 \pm 7.60e+02$ | = | $4.01e+03 \pm 5.15e+02$ | + |
| $f_{16}$ | $1.48e-01 \pm 9.27e-02$ | $\mathbf{1.29e-01 \pm 9.88e-02}$ | = | $1.39e+00 \pm 8.13e-01$ | + | $2.45e+00 \pm 3.72e-01$ | + |
| $f_{17}$ | $5.59e+01 \pm 9.02e+00$ | $4.12e+03 \pm 7.55e+02$ | + | $6.93e+01 \pm 1.29e+01$ | + | $\mathbf{3.13e+01 \pm 5.20e-01}$ | - |
| $f_{18}$ | $2.36e+02 \pm 4.78e+01$ | $3.95e+03 \pm 7.83e+02$ | + | $\mathbf{8.24e+01 \pm 1.73e+01}$ | - | $2.42e+02 \pm 6.25e+01$ | = |
| $f_{19}$ | $2.63e+00 \pm 6.76e-01$ | $3.50e+00 \pm 9.68e-01$ | + | $8.95e+00 \pm 4.34e+00$ | + | $\mathbf{8.70e-01 \pm 1.78e-01}$ | - |
| $f_{20}$ | $1.44e+01 \pm 5.53e-01$ | $1.50e+01 \pm 5.59e-09$ | + | $\mathbf{1.09e+01 \pm 7.29e-01}$ | - | $1.39e+01 \pm 4.24e-01$ | - |
| $f_{21}$ | $\mathbf{2.31e+02 \pm 5.04e+01}$ | $3.19e+02 \pm 9.07e+01$ | + | $3.02e+02 \pm 7.13e+01$ | + | $2.63e+02 \pm 5.88e+01$ | + |
| $f_{22}$ | $1.10e+03 \pm 2.96e+02$ | $6.90e+03 \pm 9.65e+02$ | + | $1.11e+03 \pm 5.32e+02$ | = | $\mathbf{1.23e+02 \pm 7.38e+01}$ | - |
| $f_{23}$ | $5.00e+03 \pm 7.59e+02$ | $6.71e+03 \pm 7.22e+02$ | + | $\mathbf{4.40e+03 \pm 8.16e+02}$ | - | $5.23e+03 \pm 5.90e+02$ | + |
| $f_{24}$ | $2.80e+02 \pm 9.99e+01$ | $8.08e+02 \pm 5.68e+02$ | + | $\mathbf{2.30e+02 \pm 1.09e+01}$ | - | $2.81e+02 \pm 1.13e+01$ | + |
| $f_{25}$ | $2.93e+02 \pm 1.55e+01$ | $3.79e+02 \pm 1.40e+02$ | + | $\mathbf{2.76e+02 \pm 1.71e+01}$ | - | $3.02e+02 \pm 6.30e+00$ | + |
| $f_{26}$ | $2.84e+02 \pm 8.59e+01$ | $4.63e+02 \pm 4.44e+02$ | = | $2.16e+02 \pm 4.23e+01$ | - | $\mathbf{2.01e+02 \pm 3.37e+00}$ | = |
| $f_{27}$ | $8.21e+02 \pm 1.64e+02$ | $7.89e+02 \pm 2.13e+02$ | = | $\mathbf{6.53e+02 \pm 1.21e+02}$ | - | $1.08e+03 \pm 6.10e+01$ | + |
| $f_{28}$ | $7.86e+02 \pm 1.82e+03$ | $2.27e+03 \pm 3.81e+03$ | + | $\mathbf{3.00e+02 \pm 2.23e-10}$ | + | $5.23e+02 \pm 5.26e+02$ | + |

Table 8: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2 on CEC2013[29] in 50 dimensions.

| | hyperSPAM-MT | CMAES | | MDE-pBX | | CCPSO2 | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $\mathbf{2.27e-13 \pm 0.00e+00}$ | $2.27e-13 \pm 0.00e+00$ | = | $2.73e-12 \pm 6.28e-12$ | + | $7.50e-12 \pm 4.07e-11$ | + |
| $f_2$ | $2.59e+04 \pm 1.28e+04$ | $\mathbf{2.27e-13 \pm 0.00e+00}$ | - | $9.35e+05 \pm 5.45e+05$ | + | $4.16e+06 \pm 2.05e+06$ | + |
| $f_3$ | $6.19e+06 \pm 8.61e+06$ | $\mathbf{3.17e+04 \pm 1.30e+05}$ | - | $1.36e+08 \pm 1.42e+08$ | + | $2.98e+09 \pm 2.84e+09$ | + |
| $f_4$ | $4.85e+02 \pm 4.85e+02$ | $\mathbf{2.27e-13 \pm 0.00e+00}$ | - | $1.17e+03 \pm 8.74e+02$ | + | $1.10e+05 \pm 4.86e+04$ | + |
| $f_5$ | $\mathbf{3.41e-13 \pm 8.70e-13}$ | $1.89e-09 \pm 7.96e-10$ | + | $4.30e-09 \pm 1.75e-08$ | = | $7.84e-04 \pm 5.48e-03$ | + |
| $f_6$ | $\mathbf{3.14e+01 \pm 1.80e+01}$ | $4.27e+01 \pm 6.15e+00$ | = | $5.89e+01 \pm 2.47e+01$ | + | $4.63e+01 \pm 1.13e+01$ | + |
| $f_7$ | $\mathbf{4.46e+01 \pm 1.93e+01}$ | $4.52e+01 \pm 1.71e+01$ | = | $6.95e+01 \pm 1.23e+01$ | + | $1.47e+02 \pm 2.02e+01$ | + |
| $f_8$ | $\mathbf{2.11e+01 \pm 6.39e-02}$ | $2.12e+01 \pm 3.41e-02$ | = | $2.12e+01 \pm 4.11e-02$ | + | $2.12e+01 \pm 3.41e-02$ | + |
| $f_9$ | $5.74e+01 \pm 4.96e+00$ | $7.79e+01 \pm 9.33e+00$ | + | $\mathbf{4.24e+01 \pm 6.64e+00}$ | - | $5.86e+01 \pm 3.25e+00$ | = |
| $f_{10}$ | $\mathbf{1.15e-02 \pm 7.41e-03}$ | $2.68e-02 \pm 1.76e-02$ | + | $4.37e-01 \pm 4.84e-01$ | + | $1.87e-01 \pm 9.78e-02$ | + |
| $f_{11}$ | $5.77e+01 \pm 9.97e+00$ | $1.99e+02 \pm 4.41e+02$ | + | $1.19e+02 \pm 3.08e+01$ | + | $\mathbf{8.70e-01 \pm 9.21e-01}$ | - |
| $f_{12}$ | $3.09e+02 \pm 1.37e+02$ | $2.39e+03 \pm 1.49e+03$ | + | $\mathbf{1.63e+02 \pm 3.22e+01}$ | - | $4.50e+02 \pm 8.38e+01$ | + |
| $f_{13}$ | $5.39e+02 \pm 1.01e+02$ | $3.23e+03 \pm 1.47e+03$ | + | $\mathbf{3.17e+02 \pm 4.71e+01}$ | - | $5.72e+02 \pm 7.14e+01$ | = |
| $f_{14}$ | $1.43e+03 \pm 3.09e+02$ | $8.73e+03 \pm 9.69e+02$ | + | $2.66e+03 \pm 8.55e+02$ | + | $\mathbf{6.85e+00 \pm 2.91e+00}$ | - |
| $f_{15}$ | $\mathbf{6.96e+03 \pm 6.72e+02}$ | $9.03e+03 \pm 9.59e+02$ | + | $7.44e+03 \pm 7.87e+02$ | + | $8.36e+03 \pm 8.60e+02$ | + |
| $f_{16}$ | $8.26e-02 \pm 4.79e-02$ | $\mathbf{7.78e-02 \pm 3.89e-02}$ | = | $1.86e+00 \pm 8.40e-01$ | + | $2.65e+00 \pm 6.30e-01$ | + |
| $f_{17}$ | $9.70e+01 \pm 1.04e+01$ | $7.06e+03 \pm 9.80e+02$ | + | $1.80e+02 \pm 3.38e+01$ | + | $\mathbf{5.16e+01 \pm 3.68e-01}$ | - |
| $f_{18}$ | $5.18e+02 \pm 8.96e+01$ | $7.05e+03 \pm 9.63e+02$ | + | $\mathbf{1.85e+02 \pm 3.15e+01}$ | - | $4.97e+02 \pm 1.05e+02$ | = |
| $f_{19}$ | $4.61e+00 \pm 9.69e-01$ | $6.02e+00 \pm 1.39e+00$ | + | $4.06e+01 \pm 2.57e+01$ | + | $\mathbf{1.49e+00 \pm 2.30e-01}$ | - |
| $f_{20}$ | $2.43e+01 \pm 5.08e-01$ | $2.50e+01 \pm 1.37e-01$ | + | $\mathbf{2.00e+01 \pm 9.34e-01}$ | - | $2.33e+01 \pm 8.48e-01$ | - |
| $f_{21}$ | $4.58e+02 \pm 3.52e+02$ | $8.22e+02 \pm 3.53e+02$ | + | $9.00e+02 \pm 3.29e+02$ | + | $\mathbf{4.54e+02 \pm 3.43e+02}$ | + |
| $f_{22}$ | $2.06e+03 \pm 4.02e+02$ | $1.18e+04 \pm 1.38e+03$ | + | $3.09e+03 \pm 1.01e+03$ | + | $\mathbf{1.14e+02 \pm 9.38e+01}$ | - |
| $f_{23}$ | $9.61e+03 \pm 1.18e+03$ | $1.18e+04 \pm 1.01e+03$ | + | $\mathbf{9.01e+03 \pm 9.03e+02}$ | - | $1.08e+04 \pm 1.27e+03$ | + |
| $f_{24}$ | $3.39e+02 \pm 3.14e+01$ | $1.78e+03 \pm 1.03e+03$ | + | $\mathbf{2.88e+02 \pm 1.68e+01}$ | - | $3.60e+02 \pm 9.51e+00$ | + |
| $f_{25}$ | $3.85e+02 \pm 2.29e+01$ | $4.87e+02 \pm 2.03e+02$ | + | $\mathbf{3.67e+02 \pm 1.44e+01}$ | - | $3.96e+02 \pm 1.14e+01$ | + |
| $f_{26}$ | $3.10e+02 \pm 1.10e+02$ | $6.56e+02 \pm 7.48e+02$ | = | $3.57e+02 \pm 7.06e+01$ | = | $\mathbf{2.19e+02 \pm 5.43e+01}$ | = |
| $f_{27}$ | $1.34e+03 \pm 2.25e+02$ | $1.33e+03 \pm 3.83e+02$ | = | $\mathbf{1.25e+03 \pm 1.31e+02}$ | - | $1.83e+03 \pm 8.52e+01$ | + |
| $f_{28}$ | $1.69e+03 \pm 3.42e+03$ | $2.52e+03 \pm 3.95e+03$ | - | $\mathbf{4.66e+02 \pm 4.64e+02}$ | + | $6.85e+02 \pm 1.02e+03$ | + |

Table 9: Average error ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2 BBOB2010 [32] in 100 dimensions.

| | hyperSPAM-MT | CMAES | | MDE-pBX | | CCPSO2 | |
|---|---|---|---|---|---|---|---|
| $f_1$ | $2.42e-13 \pm 2.12e-13$ | $\mathbf{2.84e-14 \pm 0.00e+00}$ | = | $1.39e-07 \pm 6.61e-07$ | + | $3.27e-13 \pm 1.93e-13$ | + |
| $f_2$ | $\mathbf{1.42e-13 \pm 1.68e-13}$ | $3.90e-09 \pm 2.46e-08$ | + | $1.44e-03 \pm 6.89e-03$ | + | $1.53e-12 \pm 2.34e-12$ | + |
| $f_3$ | $1.01e+02 \pm 1.79e+01$ | $2.69e+02 \pm 5.26e+01$ | + | $4.93e+02 \pm 8.69e+01$ | + | $\mathbf{7.87e+00 \pm 8.26e+00}$ | - |
| $f_4$ | $1.37e+02 \pm 1.75e+01$ | $4.12e+02 \pm 5.23e+01$ | + | $8.75e+02 \pm 1.25e+02$ | + | $\mathbf{2.22e+01 \pm 1.34e+01}$ | - |
| $f_5$ | $\mathbf{1.26e-11 \pm 4.00e-12}$ | $1.26e+02 \pm 2.19e+01$ | + | $7.55e+00 \pm 1.03e+01$ | + | $2.11e-04 \pm 1.00e-03$ | + |
| $f_6$ | $3.52e-08 \pm 5.57e-08$ | $\mathbf{7.82e-14 \pm 3.07e-14}$ | - | $4.43e+01 \pm 3.80e+01$ | + | $8.76e+01 \pm 4.14e+01$ | + |
| $f_7$ | $5.23e+01 \pm 1.48e+01$ | $\mathbf{3.74e+01 \pm 7.21e+00}$ | - | $2.96e+02 \pm 7.80e+01$ | + | $3.49e+02 \pm 5.19e+01$ | + |
| $f_8$ | $3.56e+01 \pm 7.86e+00$ | $\mathbf{9.20e-01 \pm 1.44e+00}$ | - | $2.00e+02 \pm 7.69e+01$ | + | $1.22e+02 \pm 3.73e+01$ | + |
| $f_9$ | $4.46e+01 \pm 1.12e+01$ | $\mathbf{1.13e+00 \pm 1.73e+00}$ | - | $1.34e+02 \pm 3.71e+01$ | + | $1.10e+02 \pm 3.19e+01$ | + |
| $f_{10}$ | $5.21e+02 \pm 1.42e+02$ | $\mathbf{3.35e-09 \pm 8.68e-09}$ | - | $1.69e+04 \pm 8.86e+03$ | + | $2.59e+04 \pm 7.29e+03$ | + |
| $f_{11}$ | $8.38e+01 \pm 3.46e+01$ | $\mathbf{0.00e+00 \pm 2.84e-14}$ | - | $1.58e+01 \pm 7.69e+00$ | - | $5.33e+02 \pm 1.96e+02$ | + |
| $f_{12}$ | $4.79e-02 \pm 2.52e-01$ | $\mathbf{3.41e-13 \pm 5.68e-13}$ | - | $1.57e+01 \pm 2.38e+01$ | + | $8.66e+00 \pm 1.23e+01$ | + |
| $f_{13}$ | $\mathbf{1.07e+00 \pm 1.43e+00}$ | $1.73e+00 \pm 2.45e+00$ | = | $4.54e+00 \pm 7.29e+00$ | + | $3.09e+00 \pm 3.72e+00$ | + |
| $f_{14}$ | $4.87e-05 \pm 5.88e-06$ | $\mathbf{2.04e-08 \pm 3.27e-09}$ | - | $2.56e-03 \pm 2.37e-03$ | + | $1.28e-03 \pm 1.44e-04$ | + |
| $f_{15}$ | $\mathbf{2.71e+02 \pm 4.03e+01}$ | $2.87e+02 \pm 4.38e+01$ | + | $6.76e+02 \pm 1.15e+02$ | + | $1.35e+03 \pm 2.39e+02$ | + |
| $f_{16}$ | $\mathbf{2.37e+00 \pm 8.08e-01}$ | $2.39e+00 \pm 6.48e-01$ | = | $1.71e+01 \pm 3.69e+00$ | + | $2.73e+01 \pm 4.55e+00$ | + |
| $f_{17}$ | $8.45e+00 \pm 4.46e+00$ | $9.95e+00 \pm 4.69e+00$ | + | $\mathbf{3.36e+00 \pm 4.45e-01}$ | - | $8.72e+00 \pm 1.61e+00$ | = |
| $f_{18}$ | $1.90e+01 \pm 1.06e+01$ | $2.13e+01 \pm 1.50e+01$ | = | $\mathbf{1.25e+01 \pm 1.84e+00}$ | - | $3.22e+01 \pm 5.47e+00$ | + |
| $f_{19}$ | $1.82e+00 \pm 2.94e-01$ | $\mathbf{1.38e+00 \pm 2.30e-01}$ | - | $2.37e+00 \pm 8.13e-01$ | + | $7.96e+00 \pm 1.19e+00$ | + |
| $f_{20}$ | $1.19e+00 \pm 1.61e-01$ | $1.87e+00 \pm 1.13e-01$ | + | $2.10e+00 \pm 1.18e-01$ | + | $\mathbf{5.00e-01 \pm 6.29e-02}$ | - |
| $f_{21}$ | $3.74e+00 \pm 3.84e+00$ | $1.19e+01 \pm 1.30e+01$ | + | $4.31e+00 \pm 5.83e+00$ | = | $\mathbf{3.03e+00 \pm 3.42e+00}$ | = |
| $f_{22}$ | $6.75e+00 \pm 7.22e+00$ | $1.77e+01 \pm 1.48e+01$ | + | $9.38e+00 \pm 8.52e+00$ | + | $\mathbf{5.15e+00 \pm 5.89e+00}$ | + |
| $f_{23}$ | $\mathbf{7.69e-01 \pm 4.37e-01}$ | $2.44e+00 \pm 2.03e+00$ | + | $2.30e+00 \pm 7.64e-01$ | + | $2.50e+00 \pm 4.44e-01$ | + |
| $f_{24}$ | $3.11e+02 \pm 6.12e+01$ | $\mathbf{3.11e+02 \pm 5.85e+01}$ | = | $3.74e+02 \pm 4.73e+01$ | + | $1.08e+03 \pm 1.48e+02$ | + |

Table 10: Number of statistically significant wins +, draws =, and losses - of hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2

| | CMAES | MDE-pBX | CCPSO2 |
|---|---|---|---|
| CEC2013-10D | 20+, 6=, 2- | 7+,3=,21- | 16+,7=,5- |
| CEC2013-30D | 18+, 4=, 6- | 15+,3=, 10- | 19+, 3=, 6- |
| CEC2013-50D | 18+, 6=, 4- | 17+, 2=, 9- | 18+, 4=, 6- |
| BBOB-100D | 10+, 4=, 10- | 20+, 1=, 3- | 19+, 2=, 3- |
| Total | 66+, 20=, 22- | 59+, 9=, 53- | 72+, 16=, 20- |

Table 11: Holm-Bonferroni procedure (reference: hyperSPAM-MT, Rank = 3.04e+00) ranking hyperSPAM-MT against three popular metaheuristics

| $j$ | Optimizer | Rank | $z_j$ | $p_j$ | $\delta/j$ | Hypothesis |
|---|---|---|---|---|---|---|
| 1 | MDE-pBX | 2.58e+00 | -3.33e+00 | 4.28e-04 | 5.00e-02 | Rejected |
| 2 | CCPSO2 | 2.22e+00 | -5.99e+00 | 1.06e-09 | 2.50e-02 | Rejected |
| 3 | CMAES | 2.08e+00 | -7.01e+00 | 1.21e-12 | 1.67e-02 | Rejected |

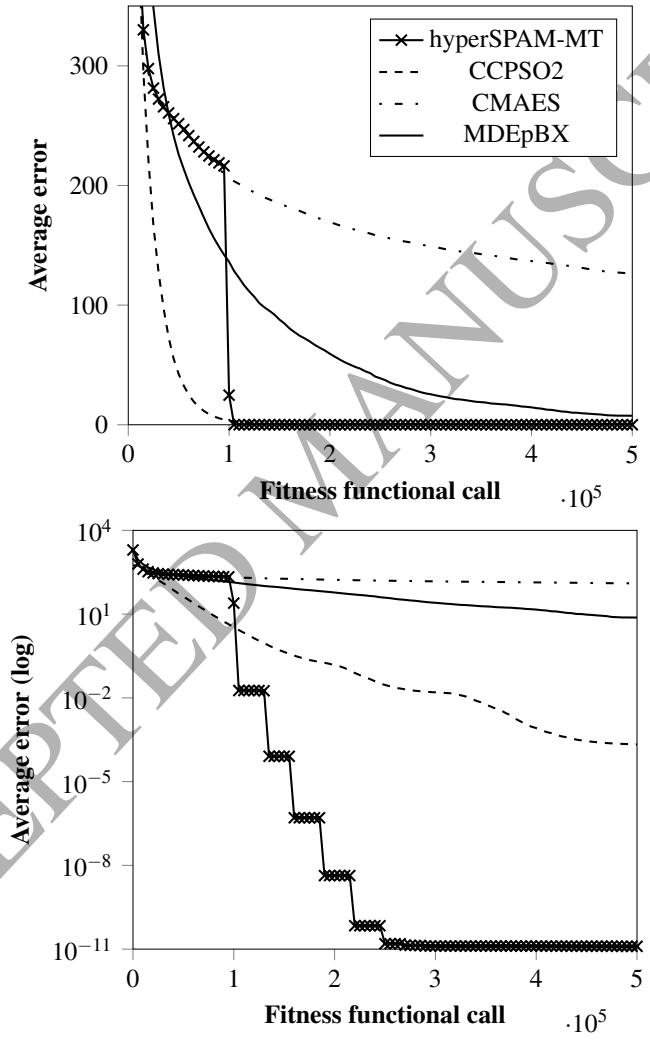


Figure 3: Average error trend of hyperSPAM-MT against the CMAES, MDE-pBX, and CCPSO2 on $f_5$ of the BBOB2010 beanchmark in 100 dimensions. The trend is shown according to a linear (above) and logarithmic (below) scale

Table 12: Holm-Bonferroni procedure (reference: hyperSPAM-MT, Rank = 4.56e+00)ranking all the algorithms present in this study

| $j$ | Algorithm | Rank | $z_j$ | $p_j$ | $\delta/j$ | Hypothesis |
|---|---|---|---|---|---|---|
| 1 | hyperSPAM-AP | 4.51e+00 | -1.82e-01 | 4.28e-01 | 5.00e-02 | Accepted |
| 2 | hyperSPAM-RS | 4.25e+00 | -1.20e+00 | 1.15e-01 | 2.50e-02 | Accepted |
| 3 | hyperSPAM-PM | 4.19e+00 | -1.42e+00 | 7.80e-02 | 1.67e-02 | Accepted |
| 4 | MDE-pBX | 3.79e+00 | -3.02e+00 | 1.27e-03 | 1.25e-02 | Rejected |
| 5 | CCPSO2 | 3.23e+00 | -5.20e+00 | 9.92e-08 | 1.00e-02 | Rejected |
| 6 | CMAES | 2.75e+00 | -7.09e+00 | 6.61e-13 | 8.33e-03 | Rejected |

Furthermore, when the full list of algorithms is considered, although hyperSPAM-MT displays the best performance, it does not significantly outperform the other hyperSPAM implementations. This fact confirms that the search algorithms composing a hybrid framework are extremely important. Even a random coordination of the search algorithms of the type in hyperSPAM-RS displays a good performance which is better than MDE-pBX, CCPSO2, CMAES, and better than one of the adaptive scheme. However, the multinomial tracking implemented in hyperSPAM-MT is the most promising scheme. According to our interpretation, this is due to a suitable balance between a randomisation of the selection of the search algorithms and a mechanism rewarding the algorithms displaying the best performance. This scheme makes use of a multinomial distribution to sample the search algorithm. Although this distribution is biased to adaptively prefer the most promising search algorithms, the forgetting factor prevents the distribution from focusing on search algorithms that were successful at earlier stages of the optimisation. Thus, the multinomial tracking tend to keep the randomization level of the selection mechanism quite high.

### *3.6. Numerical results on real-world applications*

HyperSPAM-MT has been run on the above-mentioned real-world problems sampled from CEC2011 [8]. The competitor metaheuristics used in this article have also been run on the real-world problems for comparison. Table 13 displays the results and Wilcoxon test. Table 14 displays the result of the Holm-Bonferroni procedure.

Numerical results show that there is no algorithm which outperforms the others. On average the algorithm are statistically all equivalent. On two problems the CCPSO2

Table 13: Average fitness ± standard deviation and statistic comparison (reference: hyperSPAM-MT) for hyperSPAM-MT against CMAES MDE-pBX, and CCPSO2 on CEC2011[8] in 6, 30, and 20 dimensions.

| | hyperSPAM-MT | CMAES | | MDE-pBX | | CCPSO2 | |
|---|---|---|---|---|---|---|---|
| $Prob_1(6D)$ | $1.87e+01 \pm 1.21e+01$ | $3.37e+01 \pm 1.28e+01$ | + | $8.16e+00 \pm 6.77e+00$ | - | $\mathbf{6.77e+00 \pm 3.79e+00}$ | - |
| $Prob_2(30D)$ | $-1.86e+01 \pm 4.68e+00$ | $-2.53e+01 \pm 2.74e+00$ | - | $-2.20e+01 \pm 4.32e+00$ | - | $\mathbf{-2.67e+01 \pm 1.74e+00}$ | - |
| $Prob_7(20D)$ | $7.29e-01 \pm 1.47e-01$ | $\mathbf{5.82e-01 \pm 8.31e-02}$ | - | $1.08e+00 \pm 1.77e-01$ | + | $1.16e+00 \pm 1.25e-01$ | + |

Table 14: Holm-Bonferroni procedure (reference: hyperSPAM-MT, Rank = 2.00e+00) ranking hyperSPAM-MT, MDE-pBX, CCPSO2, and CMAES on the CEC2011 real-world problems

| $j$ | Optimizer | Rank | $z_j$ | $p_j$ | $\delta/j$ | Hypothesis |
|---|---|---|---|---|---|---|
| 1 | CCPSO2 | 3.00e+00 | 1.22e+00 | 8.90e-01 | 5.00e-02 | Accepted |
| 2 | CMAES | 2.67e+00 | 8.16e-01 | 7.93e-01 | 2.50e-02 | Accepted |
| 3 | MDE-pBX | 2.33e+00 | 4.08e-01 | 6.58e-01 | 1.67e-02 | Accepted |

appears to be promising while on one CMAES appears to offer a better performance. According to our interpretation, these results, in accordance with the No Free Lunch Theorems [48], highlight that neither S nor R are very suitable to address these specific problems. However, hyperSPAM-MT still performs a reasonably well on these problems.

## 4. Conclusion

This paper proposes a simple hyperheuristic framework for continuous optimisation problems, namely hyperSPAM, and presents a study on the mechanism for coordination of the search algorithms. Numerical results show that the proposed framework appears to flexibly address a large number of different problems and outperforms popular metaheuristics, regardless of the coordination mechanism. Thus, the choice of correct search algorithms composing a hybrid algorithm appears to be fundamental. This is an important *caveat* in algorithmic design when sophisticated adaptive schemes are designed: if the search algorithms are correctly selected a simple random coordination can lead to satisfactory results.

The study on four mechanism for adaptive coordination demonstrates that, although there is no clear outperformance of any scheme over the others, an adaptation using an

evolving multinomial distribution appears to display the best performance. The randomised element of this adaptation is high due to a forgetting factor which inhibits that the distribution is skewed around the search algorithms displaying the best performance. However, the adaptation biases the selection preference and rewards the search algorithms that appears to be the most effective during the optimisation process. The detection of the correct balance between these two elements seems to be one important factor for the success of adaptation in hybrid algorithms.

## References

**References**

[1] Benlic, U., Epitropakis, M. G., Burke, E. K., 2017. A hybrid breakout local search and reinforcement learning approach to the vertex separator problem. European Journal of Operational Research 261 (3), 803–818.

[2] Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., zcan, E., Qu, R., 2013. Hyper-heuristics: a survey of the state of the art. Journal of the Operational Research Society 64 (12), 1695–1724.

[3] Burke, E. K., Kendall, G., Soubeiga, E., 2003. A Tabu Search hyperheuristic for Timetabling and Rostering. Journal of Heuristics 9 (6), 451–470.

[4] Caraffini, F., Neri, F., Iacca, G., Mol, A., 2013. Parallel memetic structures. Information Sciences 227 (0), 60 – 82.

[5] Caraffini, F., Neri, F., Passow, B., Iacca, G., 2014. Re-sampled inheritance search: High performance despite the simplicity. Soft Computing 17 (12), 2235–2256.

[6] Caraffini, F., Neri, F., Picinali, L., 2014. An analysis on separability for memetic computing automatic design. Information Sciences 265, 1–22.

[7] Cauwet, M.-L., Liu, J., Rozière, B., Teytaud, O., Feb 2016. Algorithm portfolios for noisy optimization. Annals of Mathematics and Artificial Intelligence 76 (1), 143–172.

[8] Das, S., Suganthan, P., 2010. Problem definitions and evaluation criteria for cec 2011 competition on testing evolutionary algorithms on real world optimization problems. Jadavpur Univ., Nanyang Technol. Univ., Kolkata, India.

[9] Epitropakis, M., Tasoulis, D., Pavlidis, N., Plagianakos, V., Vrahatis, M., 2012. Tracking particle swarm optimizers: An adaptive approach through multinomial distribution tracking with exponential forgetting. In: IEEE Congress on Evolutionary Computation (CEC2012). pp. 1–8.

[10] Epitropakis, M. G., Burke, E. K., 2018. Hyper-heuristics. In: Martí, R., Panos, P., Resende, M. G. C. (Eds.), Handbook of Heuristics. Springer International Publishing, Cham, pp. 1–57.
URL https://doi.org/10.1007/978-3-319-07153-4_32-1

[11] Epitropakis, M. G., Caraffini, F., Neri, F., Burke, E. K., 2014. A separability prototype for automatic memes with adaptive operator selection. In: 2014 IEEE Symposium on Foundations of Computational Intelligence, FOCI 2014, Orlando, FL, USA, December 9-12, 2014. IEEE, pp. 70–77.

[12] Epitropakis, M. G., Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P., Vrahatis, M. N., 2012. Tracking differential evolution algorithms: An adaptive approach through multinomial distribution tracking with exponential forgetting. In: Maglogiannis, Plagianakos, Vlahavas (Eds.), Artificial Intelligence: Theories and Applications. No. 7297 in LNCS. Springer, pp. 214–222.

[13] Fialho, A., 2010. Adaptive operator selection for optimization. Ph.D. thesis, Université Paris-Sud XI, Orsay, France.

[14] Garcia, S., Fernandez, A., Luengo, J., Herrera, F., 2008. A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Computing 13 (10), 959–977.

[15] Gretsista, A., Burke, E. K., 2017. An iterated local search framework with adaptive operator selection for nurse rostering. In: Battiti, R., Kvasov, D. E., Sergeyev,

Y. D. (Eds.), Learning and Intelligent Optimization - 11th International Conference, LION 11, Nizhny Novgorod, Russia, June 19-21, 2017, Revised Selected Papers. Vol. 10556 of Lecture Notes in Computer Science. Springer, pp. 93–108. URL https://doi.org/10.1007/978-3-319-69404-7_7

[16] Hallawa, A., Yaman, A., Iacca, G., Ascheid, G., 2017. A framework for knowledge integrated evolutionary algorithms. In: Squillero, G., Sim, K. (Eds.), Applications of Evolutionary Computation: 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part I. Springer International Publishing, pp. 653–669.

[17] Hansen, N., 2012. The CMA Evolution Strategy. http://www.lri.fr/ hansen/cmaesintro.html.

[18] Hansen, N., Müller, S. D., Koumoutsakos, P., 2003. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation 11 (1), 1–18.

[19] Hansen, N., Ostermeier, A., 2001. Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation 9 (2), 159–195.

[20] Ho, Y.-C., Pepyne, D. L., Mar 2002. Simple explanation of the no free lunch theorem of optimization. Cybernetics and Systems Analysis 38 (2), 292–298.

[21] Iacca, G., Caraffini, F., Neri, F., 2013. Memory-saving memetic computing for path-following mobile robots. Appl. Soft Comput. 13 (4), 2003–2016.

[22] Iacca, G., Caraffini, F., Neri, F., 2014. Multi-strategy coevolving aging particle optimization. Int. J. Neural Syst. 24 (1).

[23] Iacca, G., Neri, F., Mininno, E., Ong, Y. S., Lim, M. H., 2012. Ockham's Razor in Memetic Computing: Three Stage Optimal Memetic Exploration. Information Sciences 188, 17–43.

[24] Iliya, S., Neri, F., 2016. Towards artificial speech therapy: A neural system for impaired speech segmentation. Int. J. Neural Syst. 26 (6), 1–16.

[25] Islam, S., Das, S., Ghosh, S., Roy, S., Suganthan, P., 2012. An Adaptive Differential Evolution Algorithm With Novel Mutation and Crossover Strategies for Global Numerical Optimization. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42 (2), 482–500.

[26] Karafotias, G., Hoogendoorn, M., Eiben, A. E., 2014. Parameter Control in Evolutionary Algorithms: Trends and Challenges. IEEE Transactions on Evolutionary Computation PP (99), 1–1.

[27] Kononova, A. V., Corne, D. W., Wilde, P. D., Shneer, V., Caraffini, F., 2015. Structural bias in population-based algorithms. Information Sciences 298, 468 – 490.

[28] Li, X., Yao, X., 2012. Cooperatively Coevolving Particle Swarms for Large Scale Optimization. Evolutionary Computation, IEEE Transactions on 16 (2), 210–224.

[29] Liang, J. J., Qu, B. Y., Suganthan, P. N., Hernndez-Daz, A. G., 2013. Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization. Tech. Rep. 201212, Zhengzhou University and Nanyang Technological University, Zhengzhou China and Singapore.

[30] Mallipeddi, R., Iacca, G., Suganthan, P. N., Neri, F., Mininno, E., 2011. Ensemble Strategies in Compact Differential Evolution. In: Proceedings of the IEEE Congress on Evolutionary Computation. pp. 1972–1977.

[31] Mallipeddi, R., Mallipeddi, S., Suganthan, P. N., 2010. Ensemble strategies with adaptive evolutionary programming. Information Sciences 180 (9), 1571–1581.

[32] N. Hansen, Auger, A., Finck, S., Ros, R., et al., 2010. Real-Parameter Black-Box Optimization Benchmarking 2010: Noiseless Functions Definitions. Tech. Rep. RR-6829, INRIA.

[33] Neri, F., Cotta, C., 2012. Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation 2, 1–14.

[34] Neri, F., Cotta, C., Moscato, P., 2012. Handbook of Memetic Algorithms. Vol. 379 of Studies in Computational Intelligence. Springer.

[35] Neri, F., del Toro Garcia, X., Cascella, G. L., Salvatore, N., 2008. Surrogate Assisted Local Search on PMSM Drive Design. COMPEL: International Journal for Computation and Mathematics in Electrical and Electronic Engineering 27 (3), 573–592.

[36] Neri, F., Mininno, E., 2010. Memetic compact differential evolution for cartesian robot control. IEEE Comp. Int. Mag. 5 (2), 54–65.

[37] Peng, F., Tang, K., Chen, G., Yao, X., 2010. Population-Based Algorithm Portfolios for Numerical Optimization. IEEE Transactions on Evolutionary Computation 14 (5), 782–800.

[38] Piotrowski, A. P., 2013. Adaptive memetic differential evolution with global and local neighborhood-based mutation operators. Information Sciences 241, 164 – 194.

[39] Piotrowski, A. P., 2015. Regarding the rankings of optimization heuristics based on artificially-constructed benchmark functions. Information Sciences 297, 191 – 201.

[40] Piotrowski, A. P., Napiorkowski, J. J., 2018. Some metaheuristics should be simplified. Information Sciences 427, 32–62.

[41] Rosenbrock, H. H., 1960. An automatic Method for finding the greatest or least Value of a Function. The Computer Journal 3 (3), 175–184.

[42] Rostami, S., Neri, F., 2016. Covariance matrix adaptation pareto archived evolution strategy with hypervolume-sorted adaptive grid algorithm. Integrated Computer-Aided Engineering 23 (4), 313–329.

[43] Segredo, E., Segura, C., León, C., Apr 2014. Memetic algorithms and hyperheuristics applied to a multiobjectivised two-dimensional packing problem. Journal of Global Optimization 58 (4), 769–794.

[44] Thierens, D., 2007. Adaptive strategies for operator allocation. In: Lobo, F. G., Lima, C. F., Michalewicz, Z. (Eds.), Parameter Setting in Evolutionary Algorithms. No. 54 in Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 77–90, 00042.

[45] Tseng, L.-Y., Chen, C., 2008. Multiple trajectory search for Large Scale Global Optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation. pp. 3052–3059.

[46] Wang, X., Zhang, G., Neri, F., Jiang, T., Zhao, J., Gheorghe, M., Ipate, F., Lefticaru, R., 2016. Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots. Integrated Computer-Aided Engineering 23 (1), 15–30.

[47] Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biometrics Bulletin 1 (6), 80–83.

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

[49] Zeng, Y., Chen, X., Ong, Y., Tang, J., Xiang, Y., 2017. Structured memetic automation for online human-like social behavior learning. IEEE Trans. Evolutionary Computation 21 (1), 102–115.

[50] Zhang, G., Rong, H., Neri, F., Pérez-Jiménez, M. J., 2014. An optimization spiking neural P system for approximately solving combinatorial optimization problems. Int. J. Neural Syst. 24 (5).

[51] Zhu, X., Li, X., 2016. An enhanced greedy random adaptive search procedure with path-relinking for no-wait flowshop problem with setup times. Integrated Computer-Aided Engineering 23 (1), 51–68.