Adaptive Inflationary Differential Evolution

Edmondo Minisci and Massimiliano Vasile
Department of Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, Scotland, UK
email: {edmondo.minisci, massimiliano.vasile}@strath.ac.uk)

***Abstract*— In this paper, an adaptive version of Inflationary Differential Evolution is presented and tested on a set of real case problems taken from the CEC2011 competition on real-world applications. Inflationary Differential Evolution extends standard Differential Evolution with both local and global restart procedures. The proposed adaptive algorithm utilizes a probabilistic kernel based approach to automatically adapt the values of both the crossover and step parameters. In addition the paper presents a sensitivity analysis on the values of the parameters controlling the local restart mechanism and their impact on the solution of one of the hardest problems in the CEC2011 test set.**

I. INTRODUCTION

With the continuous progress of technologies, real world design and optimization problems are becoming progressively more complex, and there is the clear need to create and implement more effective and efficient search algorithms. An approach used to create new algorithms is to hybridize existing ones by appropriately mixing some of their building blocks. By following this approach, and based on some new theoretical results on the convergence of Differential Evolution (DE)[2], the authors recently proposed Inflationary Differential Evolution Algorithm (IDEA)[1], which combines DE with the restarting procedure of Monotonic Basin Hopping (MBH) algorithm [3], [4]. Although IDEA showed very good results when applied to problems with a single or multi-funnel landscape, its performance was found to depend on the parameters parameters controlling both the convergence of DE and MBH, and the inflationary stopping criterion used to terminate the DE search.

Despite its simplicity, the standard DE alone shows good performance on a broad range of problems featuring multimodal, separable and non-separable structures, but the performance is strongly influenced by three parameters: the population size, n_{pop} , the crossover probability, CR, and the differential weight (or step parameter), F. In addition, it was reckoned that the chosen strategies for mutation and crossover [5] plays an important role.

The need of self-adapting techniques especially for these two parameters has been widely recognized in literature. In [7] the authors introduced a fuzzy adaptive differential evolution algorithm using fuzzy logic controllers to adapt the parameters for the mutation and crossover operators. The Self-Adaptive DE (SADE), described in [8], incorporates a mechanism that self adapts both the parameters CR and F and the trial vector generation strategy. In [9] an adaptation strategy is proposed for parameter F, while CR is kept constant. In [10] both control parameters are added to each individual of the population and evolve with it.

In this work, an alternative approach is proposed for the on-line adaptation of both CR and F parameters and is embedded into the general framework of IDEA. The proposed approach uses the Parzen kernel method to build a joint probabilistic representation of the most promising region of the bi-variate $CR{-}F$ space. The resulting probability density function (PDF) is updated during the optimization process on the basis of obtained results.

The paper starts with a section that introduces the main characteristics of IDEA and the new adaptive technique. Then the test cases are described and some comparative results are presented, including an analysis of the impact of some key parameters controlling the convergence of IDEA.

II. ADAPTIVE INFLATIONARY DIFFERENTIAL EVOLUTION ALGORITHM

The new algorithm proposed here is a further development of a previously developed algorithm, IDEA[1], which is based on a synergic hybridization of a standard DE algorithm and the strategy behind the MBH algorithm [3], [4]. The resulting algorithm was shown to outperform both standard DE and MBH on a number of challenging space trajectory design problems, featuring multiple funnel structures.

IDEA works as follows: a DE process is run till the population ($\mathbf{x}_{i,k}$, for $i \in [1, ..., n_{pop}]$), contracts below a predefined threshold. When this contraction condition is satisfied, a local search is performed from the best individual in the population. Then, the local minimum is archived and the population is restarted in a bubble around the local minimum. This first restart mechanism was called *local restart*. Local restart is iterated up to a predefined maximum value. When this value is reached the population is restarted at a distance from the cluster of local minima found thus far. The restarting approach allows the algorithm to escape local optima, thus strongly mitigating the risks of premature convergence, a problem affecting standard DE, due to the use of a strong selection criterion with direct competition between one parent and the related offspring [11].

As mentioned in Sec. (I), the performance of IDEA depend on some parameters of the embedded DE, such as the population size, n_{pop} , the crossover probability, CR, and the differential weight, F. In this work the original IDEA is modified to let the algorithm understand and learn the structure of the problem and self-adapt the two parameters

CR and F. As shown in Algorithm 1, the optimization procedure starts by setting values of (n_{pop}, the maximum number of local restarts, iun_{max}, the size of the convergence box, tol_{conv}, $\rho_{A,max}$, and δ_c) and by initializing the population. Then the joint PDF for CR and F, $\mathbf{CRF}_p$, is initialised to be a uniform distribution. At this point, the actual optimization loop starts by sampling the two vectors $\mathbf{CR}_k$ and $\mathbf{F}_k$, where k is the current iteration. DE is run drawing probabilistically a value for F and CR from $\mathbf{CRF}_p$ and $\mathbf{CRF}_p$ is updated on the basis of the improvement of the individual using the drawn values of F and CR. At this point, if the population contracts below the predefined threshold, a local optimizer from current minimum is run, and at the end of local optimization, if the local optimizer failed to improve the value of f_{min} more than iun_{max} times, the population is restarted globally and iun is set to 0, otherwise, the population is restarted within a local bubble and $iun = iun + 1$. At this point, if the population is re-initialized, the loop restarts from the initialization of $\mathbf{CRF}_p$, otherwise just the DE loop restarts. As a terminal criterion, the algorithms stops if the maximum number of function evaluations, $n_{feval,max}$, has been performed.

First, the initialization of the $\mathbf{CRF}_p$ to uniform distribution, step (3) of Alg. (1), is done by building a regular mesh with $(n_D + 1) \times (n_D + 1)$ points (where n_D is the dimensionality of the problem) in the space ($CR \in [0.1, 0.99] \times F \in [-1, 1]$). A Gaussian kernel is then allocated on each node and the PDF is built by Parzen approach [12]. A step change value, dd is linked to each kernel (row of $\mathbf{CRF}_p$) and its initial value is set = 0. At step (4) of Alg. (1) n_{pop} values of CR and F are sampled from the Parzen distribution and each couple of CR and F values is associated to one element of the population and used to create the offspring on the basis of the chosen strategy.

The updating procedure is detailed in Alg. 2. During the optimization, the location of the kernels is updated on the basis of the obtained results. More in details, to update the matrix containing the location of kernel centers ($\mathbf{CRF}_p$) after that rows of $\mathbf{CRF}_p$ are sorted on the basis of the associated value of dd (step 5 of Alg. 1), if the objective function of the offspring has a value that is strictly lees then the parents (it is supposed a minimization problems) then the element of the sorted $\mathbf{CRF}_p$ are sequentially evaluated and the first time that the associated dd value of the row is less than the difference between the objective function of the parent and that of the offspring then the F value used to operate on the individual $\mathbf{x}_{i,k}$ substitutes the element $\mathbf{CRF}_{p,2,j,k}$. The CR value used to operate on the individual $\mathbf{x}_{i,k}$ substitutes the element $\mathbf{CRF}_{p,1,j,k}$ only if the difference between parent and offspring is greater than a predefined threshold CRC. The different approach for updating the CR coordinate of the kernels is meant to dump the learning of the crossover to avoid the too fast convergence toward the extremes of the allowed range that can occur in some cases. Note that, as for other self-adaptive schemes, the adaptive version of IDEA has an additional parameter to be adjusted: the threshold on

Algorithm 1 Adaptive Inflationary Differential Evolution Algorithm (AIDEA)

1: Set values for n_{pop}, iun_{max}, tol_{conv}, $\rho_{A,max}$, and δ_c, set $n_{feval} = 0$ and $k = 1$
2: Initialize Population ($\mathbf{x}_{i,k}$ for all $i \in [1, ..., n_{pop}]$)
3: A regular mesh with $(n_D + 1)^2$ points (where n_D is the dimensionality of the problem) in the space $CR \in [0.1, 0.99]xF \in [-1, 1]$; Initialize $\mathbf{CRF}_p$ with points of the mesh: $\mathbf{CRF}_{p,1,j} \leftarrow \mathbf{CR}_j\mathbf{F}_j$ for all $j \in [1, ..., (n_D + 1)^2]$; Associate to each row of $\mathbf{CRF}_p$ and element $dd_j = 0$ for all $j \in [1, ..., (n_D + 1)^2]$
4: Sample $\mathbf{CR}_{i,k}$ and $\mathbf{F}_{i,k}$, for all $i \in [1, ..., n_{pop}]$, from $\mathbf{CRF}_p$
5: $RowSort(\mathbf{CRF}_p)$ is terms of dd values
6: **for all** $i \in [1, ..., n_{pop}]$ **do**
7: $\quad$ $\mathbf{x}_{i,k+1} \leftarrow$ Apply DE $Strategy(\mathbf{x}_{i,k}, \mathbf{CR}_{i,k}, \mathbf{F}_{i,k})$
8: $\quad$ $n_{feval} = n_{feval} + 1$
9: $\quad$ Update Parzen Distribution (see Alg. (2))
10: **end for**
11: $k = k + 1$
12: $\rho_A = \max(\|\mathbf{x}_{i,k} - \mathbf{x}_{j,k}\|)$ for $\forall \mathbf{x}_{i,k}, \mathbf{x}_{j,k} \in P_{sub} \subseteq P_k$
13: **if** $\rho_A < tol_{conv}\rho_{A,max}$ **then**
14: $\quad$ Run a local optimizer a_l from $\mathbf{x}_{best}$ and let $\mathbf{x}_l$ be the local minimum found by a_l
15: $\quad$ **if** $f(\mathbf{x}_l) < f(\mathbf{x}_{best})$ **then** $f_{best} \leftarrow f(\mathbf{x}_l)$
16: $\quad$ **if** $f(\mathbf{x}_{best}) < f_{min}$ **then**
17: $\quad\quad$ $f_{min} \leftarrow f(\mathbf{x}_{best})$; $iun = 0$
18: $\quad$ **else**
19: $\quad\quad$ $iun = iun + 1$
20: $\quad$ **end if**
21: $\quad$ **if** $iun \leq iun_{max}$ **then**
22: $\quad\quad$ Define a bubble D_l such that $\mathbf{x}_{i,k} \in D_l$ for $\forall \mathbf{x}_{i,k} \in P_{sub}$ and $P_{sub} \subseteq P_k$
23: $\quad\quad$ $A_g = A_g + \{\mathbf{x}_{best}\}$ where $\mathbf{x}_{best} = \arg\min_i f(\mathbf{x}_{i,k})$
24: $\quad\quad$ Initialize Population ($\mathbf{x}_{i,k}$ for all $\in [1, ..., n_{pop}]$) in the bubble $D_l \subseteq D$
25: $\quad$ **else**
26: $\quad\quad$ Define clusters in the archive and compute the baricenter $\mathbf{x}_{c,j}$ of each cluster with $j = 1, ..., n_c$.
27: $\quad\quad$ Initialize Population ($\mathbf{x}_{i,k}$ for all $i \in [1, ..., n_{pop}]$) in D such that $\forall i, j, ||\mathbf{x}_{i,k} - \mathbf{x}_{c,j}|| > \delta_c$.
28: $\quad$ **end if**
29: $\quad$ **Termination** Unless $n_{feval} \geq n_{feval,max}$, *goto* (3)
30: **else**
31: $\quad$ **Termination** Unless $n_{feval} \geq n_{fevalmax}$, *goto* (4)
32: **end if**

the minimum expected improvement of the cost function. This threshold is used to limit the updating of CR, a failsafe procedure that has proven to improve the robustness of the algorithm.

III. TEST CASES

The test cases considered in this paper were taken from the technical report describing the CEC 2011 competition

Algorithm 2 PDF unptating procedure for AIDEA

```
if f(x_{i,k+1}) < f(x_{i,k}) then
    for all do j ∈ [1, ..., (n_D + 1)^2]
        if dd_j < (f(x_{i,k}) − f(x_{i,k+1})) then
            if (f(x_{i,k}) − f(x_{i,k+1})) > CRC then
                CRF_{p,1,j,k} ← CR_{i,k}
            end if
            CRF_{p,2,j,k} ← F_{i,k}; dd_j ← (f(x_{i,k}) − f(x_{i,k+1})); Break For Loop
        end if
    end for
end if
```

[6]. The collection of all minima obtained during the testing campaign allowed building a concise graphical representation of the structure of the problem by using the *intra-level* D_{IL}, and *trans-level* D_{TL} distance graph proposed in [1]. Minima are grouped, according to the value of their objective function, in levels, and for each level D_{IL} is computed as the average value of the relative distance of each local minimum with respect to all other local minima within the same level, while D_{TL} is the average value of the relative distance of each local minimum with respect to all other local minima in the lower level. The D_{TL} for the lowest level is computed as the average distance with respect to the best-known solution. The values D_{IL} and D_{TL} give an immediate representation of the diversity of the local minima and the probability of transition from one level to the lower one. Distances are computed by considering all variables normalized in $[0, 1]$. The reader can find more details on the procedure in [1].

First two cases were chosen to demonstrate how the proposed code works on single funnel multimodal functions, which are challenging but usually solved by other codes in literature, but the vast majority of tests were performed on the third test case which is much harder and there were not optimal solutions available yet.

1) Tersoff Potential Function Minimization Problem: It is case 5 in the report [6]. The problem considers 10 silicon atoms, whose relative positions should be optimized to minimize the Tersoff potential, V_{Ter},governing the inter-atomic interaction. The dimensionality of the problem is $n_D = 30$, and bounds are: LB=[0, 0, 0, -4, -4, -4, -4.25, -4.25, -4.25, -4.5, -4.5, -4.5, -4.75, -4.75, -4.75, -5, -5, -5, -5.25, -5.25, -5.25, -5.5, -5.5, -5.5, -5.75, -5.75, -5.75, -6, -6, -6]; UB=[4, 4, π, 4, 4, 4, 4.25, 4.25, 4.25, 4.5, 4.5, 4.5, 4.75, 4.75, 4.75, 5, 5, 5, 5.25, 5.25, 5.25, 5.5, 5.5, 5.5, 5.75, 5.75, 5.75, 6, 6, 6]. The best solution found, with $f = -36.929$, is x_{opt}=[1.5169, 0.048489, 0.85633, -0.38885, -1.0413, -0.032398, -0.30653, 2.1271, -0.46709, 1.9542, 2.6998, -0.87986, -0.5422, 0.13309, 2.0235, 0.67688, -1.8953, 1.904, 0.27924, 3.2425, -2.4656, 2.4006, 2.492, -3.1928, 0.91344, 0.98226, -2.1435, -1.6823, -1.9195, 1.7419].

As can be seen in Fig. 1, the search space is characterized by a single, multimodal funnel with a flat and broad low region, $f < -36.5$, and two distinct basins for solutions with $-36.9 \leq f \leq -36.8$ (red in Fig. 1), and solutions with $f < -36.9$ (blue in Fig. 1). In what follows this problem is referred as Case 1.

2) Spread Spectrum Radar Polyphase Code Design: This test case is the number 7 in the CEC2011 report [6]. It is related to the polyphase pulse compression code synthesis, and is formulated as a $min{-}max$ optimization problem, with $n_D = 20$ design parameter. The objective is to minimize the module of the biggest among the samples of the so-called auto-correlation function, AC_{max}, which is related to the complex envelope of the compressed radar pulse at the optimal receiver output, while the variables represent symmetrized phase differences [17] All variables are bounded $\in [0, 2\pi]$, and the best solution found, with $f = 0.5$, is x_{opt}=[2.5725, 2.6228, 5.5686, 0.73972, 1.0953, 0.83449, 5.5796, 1.2897, 1.4654, 4.4623, 2.9833, 2.7519, 3.6232, 4.6328, 4.6773, 4.0213, 4.7433, 4.5053, 4.0768, 3.8608].

Also in this case the search space is characterized by a single, multimodal funnel with a flat and broad optimal region, $f < 0.51$ (see Fig. 2). In what follows this problem is referred as Case 2.

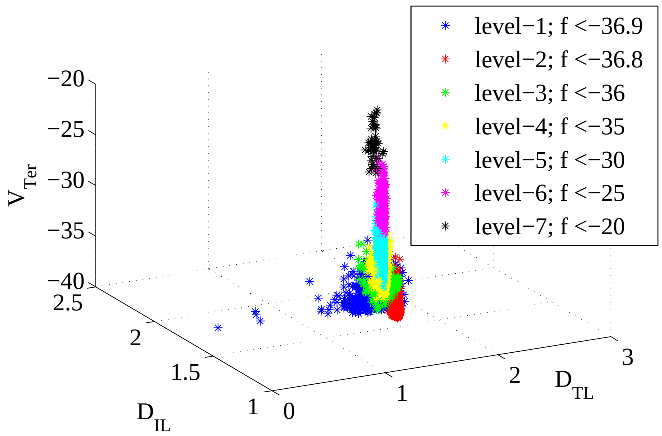

Fig. 1. Relative distances of the local minima for the Tersoff Potential case

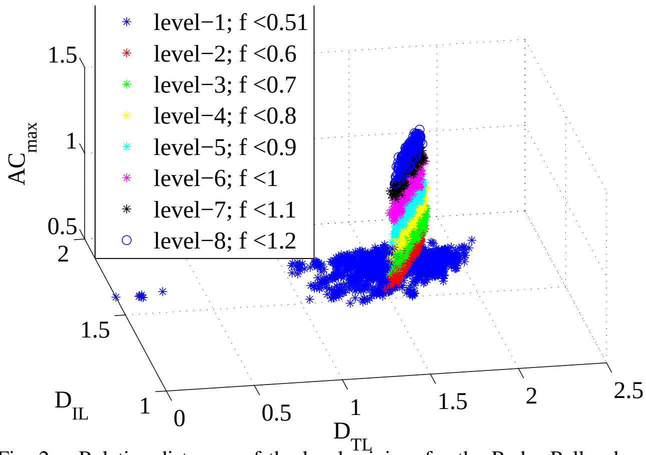

Fig. 2. Relative distances of the local minima for the Radar Polly phase Code Design case

A. *Messenger mission*

The third test case is the optimization of a multigravity assist trajectory with deep space manoeuvres (MGA-DSM)[13]: the multi-gravity assist transfer to Mercury, similar to the Messenger mission. The dimensionality of the problem is $n_D = 26$ and bounds and current known optimal solution are reported into Tab. I. In the table the solution vector is organized as in the ESA-ACT formulation [14]. As in the ESA-ACT formulation, the total ΔV of the spacecraft is minimized. Note that the optimal solution shown in Tab. I has not been published elsewhere before.

TABLE I

BOUNDS AND OPTIMAL SOLUTION FOR MESSENGER MISSION CASE - THE SOLUTION VECTOR IS REARRANGED AS IN THE ESA-ACT FORMULATION

LB	UB	Optimal
1900	2300	2038.03929616519
2.5	4.05	4.049996292063
0	1	0.556671418496
0	1	0.634280071715
100	500	451.600564550433
100	500	224.694751687357
100	500	221.839034379715
100	500	263.91480200672
100	500	359.354749401042
100	600	444.599274004631
0.01	0.99	0.607007547348
0.01	0.99	0.272048501594
0.01	0.99	0.692428663742
0.01	0.99	0.638908117493
0.01	0.99	0.829095716093
0.01	0.99	0.873723700599
1.1	6	1.774896822334
1.1	6	1.100004754835
1.05	6	1.050148253516
1.05	6	1.079891515997
1.05	6	1.40370492038
$-\pi$	π	2.758595938641
$-\pi$	π	1.575027216724
$-\pi$	π	2.602608233794
$-\pi$	π	2.272846047293
$-\pi$	π	1.579719453208

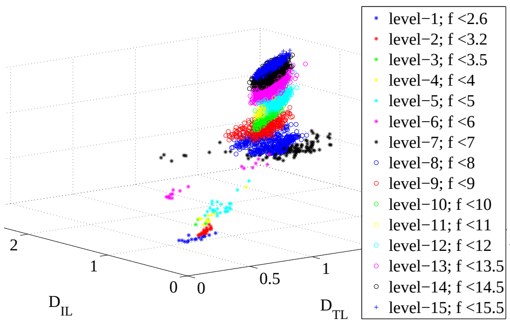

Fig. 3. Relative distances of the local minima for the Messenger mission case

The structure of the search space for the Messenger mission problem, as shown in Figs. (3) and (4), appears to be characterized by two main substructures. Solutions with

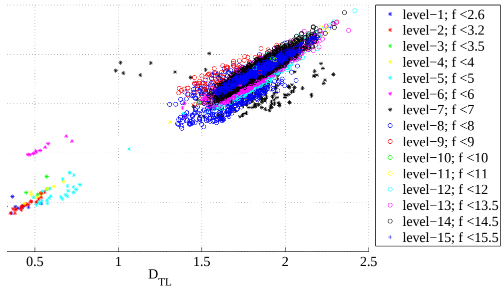

Fig. 4. Relative distances of the local minima for the Messenger mission case - Plane view

$\Delta V > 6km/s$ belong to a macro basin containing a very high number of local minima, characterized by big intra-level and trans-level distances, and is a very large multi-modal funnel. On the other hand, optimal solutions, with $\Delta V < 6km/s$, are located into a secondary basin, have smaller intra-level and trans-level distances, and the local structure is multi-funnel like.

IV. TEST RESULTS

In this Section the results of all test cases are presented and commented. The results on Cases 1 and 2 (see Sections III-.1 and III-.2) are described first, and then the space trajectory problem is used to further analyse the characteristics and critical aspects of the proposed algorithm. For all tests, the adopted DE strategy was $DE/best/1/bin$, $tol_{conv} = 0.2$, $\delta_c = 0.1$, $CRC = 3$ and all reported statistics are computed on the results obtained from 100 independent runs.

A. *Results on Cases 1 and 2*

Few different settings of AIDEA were used to solve these problems, and the algorithm performed always very well, finding the global optima with high reliability within the limit of $1e5$ function evaluations as required for the CEC 2011 competition. The embedded restart mechanism makes AIDEA perfectly suitable for solving problems with funnel like multimodal structures. In Tables (II) and (III) the results obtained by AIDEA on Case 1 and 2 are shown. AIDEA is set with $n_{pop} = 20$, $iun_{max} = 10$, $\delta_b = 0.1$, where $\pm\delta_b$ is added to current solution to create the local bubble for local restart (step 24 in Alg. (1)), and is compared to two of the best performing algorithms of the CEC 2011 competition, the Genetic Algorithm with Multi Parent Crossover (GA-MPC) [15] and the Weed Inspired Differential Evolution (WI-DE) [16].

TABLE II

RESULTS OF AIDEA ON CASES 1 COMPARED TO WI-DE AND GA-MPC

$Alg.$	Min	$Mean$	Max	$Str.Dev.$
$AIDEA$	-36.9286	-36.8527	-35.5171	0.2442
$WI-DE$	-36.8	-35.6	-34.2	0.904
$GA-MPC$	-36.84537	-35.03883	-34.10760	0.8329

TABLE III

RESULTS OF AIDEA ON CASES 2 COMPARED TO WI-DE AND GA-MPC

$Alg.$	Min	$Mean$	Max	$Str.Dev.$
$AIDEA$	0.5	0.5159	0.6384	0.0340
$WI-DE$	0.5	0.656	0.993	0.116
$GA-MPC$	0.5	0.7484	0.9334	0.1249

A better understanding of the performance obtained on these test cases can be achieved by looking at Figures (5) and 6), where the distribution of the best results obtained over the 100 runs are plotted for Case 1 and 2, respectively. As it is expected, when the algorithm can find the global optimum with high reliability, the distribution cannot be approximated by a Gaussian (dashed red curve in the figures) and converges to an exponential one. Moreover, due to the fact that the algorithm can stagnate only into certain basins, the distribution is in general multimodal and discontinuous, as it is evident in Fig. (5). In these cases the distribution can be better approximated by other means, such as the Parzen kernel approach used here (continuous red curve in the figures). It should be noted that the global minimum of case 1, $f_{min} = -36.9286$ is not generally reached by other algorithms [15], [16] which stagnate on the solution $f_{min} = -36.84$ (second peak from left in Fig. (5).

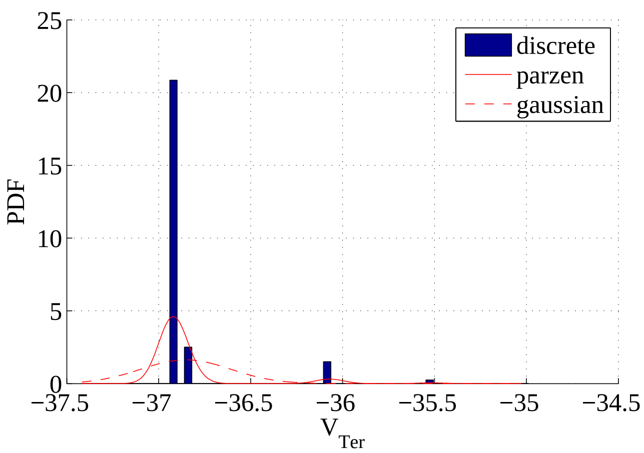

Fig. 5. Distributions of best results obtained by AIDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$) on Case 1

B. Results on Messenger mission

In order to better evaluate the performance and critical aspects of the proposed algorithm on a more difficult and challenging problem, both AIDEA and IDEA were run on the Messenger mission test case with different settings. From the results of the 2011 competition it appeared evident that none of the algorithms achieved near optimal solutions, since none of them was able to jump into the secondary structure of the problem (see Figs. 3 and 4) within the $1.5e5$ function evaluations limit [15], [16]. For this work $5e6$ function evaluations were considered and, again, all reported statistics

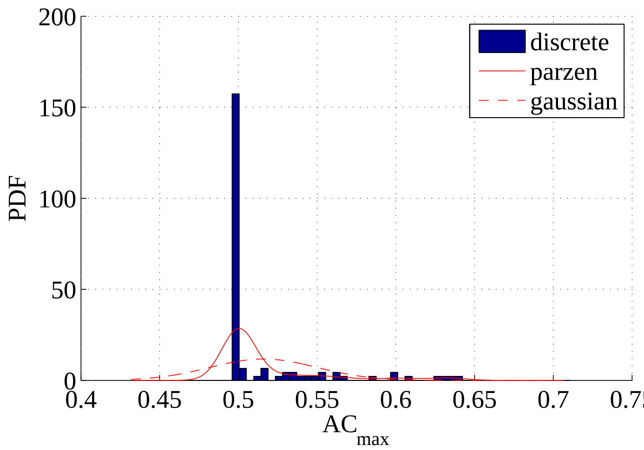

Fig. 6. Distributions of best results obtained by AIDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$) on Case 2

were computed on the results obtained from 100 independent runs.

First a direct comparison between IDEA and AIDEA with equal values of common parameters was performed. Here the case with $n_{pop} = 40$, $iun_{max} = 20$, and $\delta_b = 0.2$ is reported. Main results for AIDEA and six different instances of IDEA are summarized in Tab. (IV). As it can be expected, IDEA can have optimal performance, if DE is well tuned, as in the case $(F, CR) = (0.5, 0.9)$, but performance can get considerably worse, if F and CR are mis-tuned, as in the case $(F, CR) = (0.9, 0.5)$. On the other hand, performance of AIDEA for this test benchmark are always just below those obtained by the best tuned IDEA, also when different values of n_{pop}, iun_{max}, and δ_b are considered.

TABLE IV

COMPARISON BETWEEN IDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$) AND AIDEA PERFORMANCE ON MESSENGER CASE OVER 100 RUNS AFTER $5e6$ FUNCTION EVALUATIONS - THE FIRST COLUMN CONTAINS THE VALUES OF F AND CR USED FOR IDEA

F, CR	Min	$Mean$	Max	$Str.Dev.$
0.1, 0.5	3.1792	5.9029	8.4344	0.9158
0.1, 0.9	3.1774	6.1825	8.1495	0.9254
0.5, 0.5	3.2385	6.2099	13.8993	1.6175
0.5, 0.9	2.7784	5.1268	6.3625	1.1023
0.9, 0.5	6.2466	10.8371	15.6773	3.2088
0.9, 0.9	3.2829	6.1514	7.5227	0.6653
$AIDEA$	3.1270	5.3790	6.4898	0.9218

In order to better understand how both the algorithms work and better compare the results, the distribution of 100 best results obtained by AIDEA is shown in Fig. (7), and compared to the similar distributions of the best and second best instances of IDEA given in Figures (8) and (9), respectively. Again, it is immediately clear that for these cases the distributions are not Gaussian. The shape of the distributions cannot be described by just mean and standard deviation values, and the usually also reported values of min, max, and $median$ can be of little help. The histogram is multimodal, with peaks revealing the attraction basins, and a kernel based approach could be better used to approximate the PDF. For the case in hand, histograms confirm that the performance of AIDEA are close to the best IDEA, and for both cases the solutions in the basins with $\Delta V < 6$ are almost equally distributed.

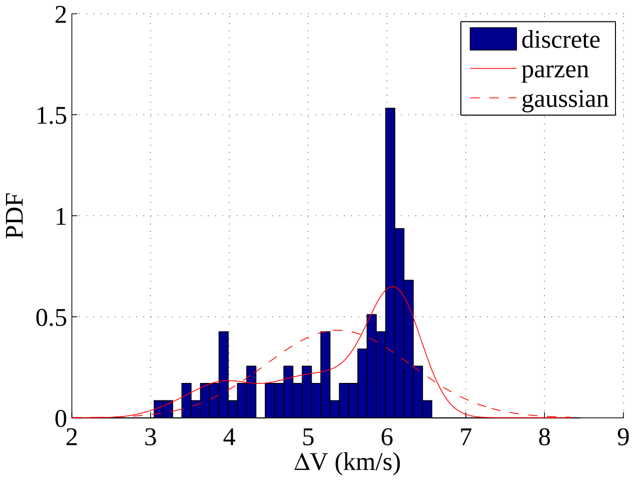

Fig. 7. Distributions of best results obtained by AIDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$) on Messenger case

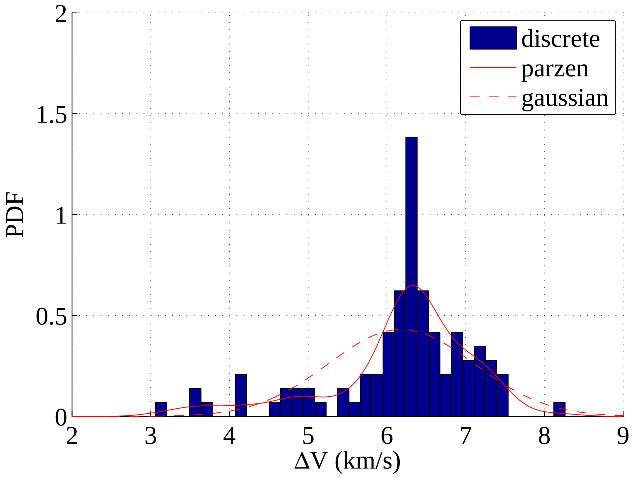

Fig. 8. Distributions of best results obtained by IDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$, $F = 0.1$, $CR = 0.9$) on Messenger case

The evolution of performance with the number of function evaluations for AIDEA and the best IDEA is given in Fig. (10). Data are shown from $1.5e5$ function evaluations, which is the maximum value for the CEC 2011 competition, to $5e6$.

In both cases, the performance obtained at $1.5e5$ function evaluations are comparable with the best performance obtained by other algorithms during the competition [15], [16], as reported in Tab. V, but, differently from what happens to other algorithms, performance keep improving with the number of function evaluations, mainly due to the restart mechanism preventing stagnation.

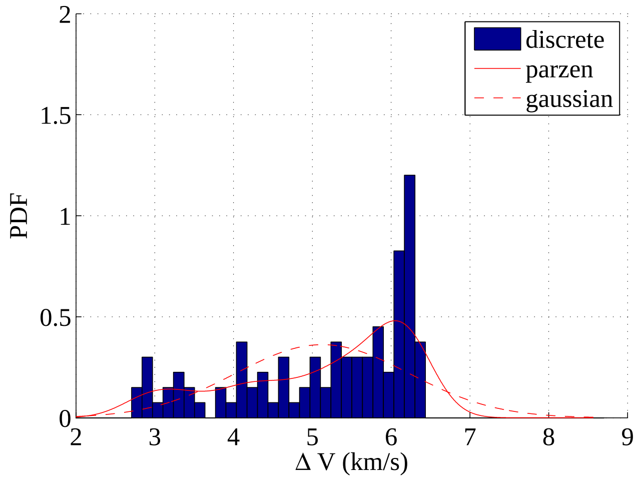

Fig. 9. Distributions of best results obtained by IDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$, $F = 0.5$, $CR = 0.9$) on Messenger case

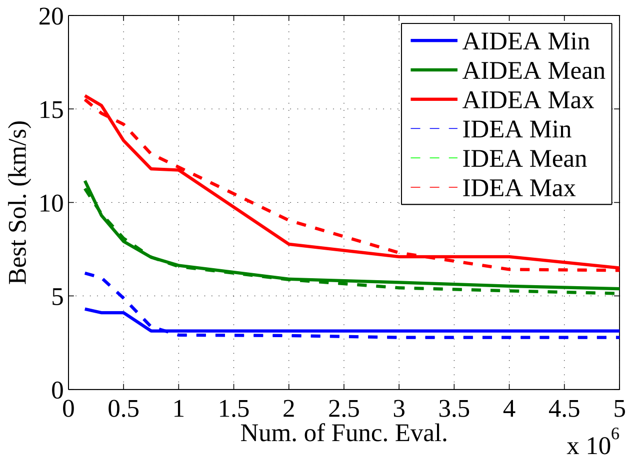

Fig. 10. Evolution of best results obtained by AIDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.2$) and IDEA ($F = 0.5$, $CR = 0.9$) on Messenger case

In Figure 11 the performance of the Genetic Algorithm with Multi-Parent Crossover (GA-MPC) [15] with population size $= 200$ are plotted as function of the function evaluations up to $1e6$ and it can be seen that the algorithm stagnate after $2e5$ evaluations.

In Tab. VI reported results were obtained by considering one population size, $n_{pop} = 40$), but different values of iun_{max}, and δ_b. For all values of iun_{max}, the minimum achievable value is always affected by the size of the local bubble, and the smaller the size the better is the minimum result achieved. This is due to the fact that the algorithm explores better the local area and if it is close to the minimum of the function it is more likely that a transition in the optimal direction occurs. On the other hand, a too small δ_b could prevent the transition to lower levels in other cases. This is particularly evident when the iun_{max} is high and the global restart does not occur often and cannot mitigate

TABLE V

COMPARISON AMONG AIDEA ($n_{pop} = 40$) WI-DE AND GA-MPC ON MESSENGER CASE FOR $1.5e5$ FUNCTION EVALUATIONS

$Alg.$	Min	$Mean$	Max	$Str.Dev.$
$AIDEA$	4.3008	11.16029 15.7070	2.9550	
$WI-DE$	6.78	11.5	13.2	2.44
$GA-MPC$	7.0956	1.2818	1.6925	3.2413

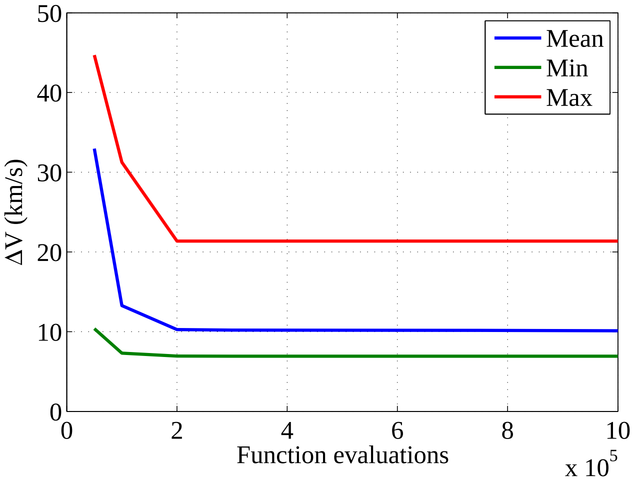

Fig. 11. Evolution of best results obtained by GA-MPC ($n_{pop} = 200$) on Messenger case

the effect of a too small local bubble, such as in the case $(iun_{max}, \delta_b) = (20, 0.1)$. As can be seen in Fig. (12), the algorithm with this setting can get stuck into zones with high values ($\Delta V > 7km/s$) of the objective function and is not able to perform the transition to lower levels, or, overall, to the secondary structure containing the global optimum, within the allowed number of function evaluations.

The combination iun_{max}, δ_b has almost the same influence on the performance also if different population sizes are considered, as can be seen in Tables VII and VIII where the same statistics are reported for tests with $n_{pop} = 20$ and $n_{pop} = 10$, respectively. The comparison of the statistics in the three Tables also demonstrates that the AIDEA is robust against different values of the populations size. It is worth noting that the population size of the embedded DE should be much smaller than the size of a standard DE, to allow a faster convergence and multiple following restarts.

The analysis of the results for this complex test case confirm the validity of the inflationary approach, which is made more robust by the technique for the on-line adaptation of DE control parameters. On the other hand, tests also make clear that to further enhance the algorithm other critical parameters should be automatically set during the optimization process. The population size is for sure one of them, but a correct combination of number of local restarts and dimension of local bubble is even more critical for a system much relying on restart both to exploit (local restart) and explore (global restart) the search space.

TABLE VI

PARAMETRIC ANALYSIS FOR AIDEA PERFORMANCE ($n_{pop} = 40$) ON MESSENGER CASE - THE FIRST COLUMN CONTAINS THE USED VALUES OF iun_{max} AND δ_b

iun_{max}, δ_b	Min	$Mean$	Max	$Str.Dev.$
3, 0.1	3.1679	6.2990	7.8775	0.8931
3, 0.2	3.5990	6.3827	7.5146	0.5812
3, 0.3	3.8868	6.3976	7.9035	0.6764
3, 0.4	4.6436	6.5982	7.6359	0.5496
5, 0.1	2.5010	6.1386	8.9341	1.0020
5, 0.2	3.3372	6.0984	7.2804	0.7345
5, 0.3	3.4961	6.2529	7.6078	0.7241
5, 0.4	4.0272	6.5241	7.7957	0.6375
10, 0.1	2.7254	6.2214	8.1943	0.8281
10, 0.2	3.5191	5.7837	7.1372	0.8263
10, 0.3	3.1761	5.9968	6.9093	0.7058
10, 0.4	4.3288	6.3397	7.3559	0.6389
20, 0.1	2.5350	6.5734	12.1774	1.5011
20, 0.2	3.1270	5.3791	6.4898	0.9218
20, 0.3	3.3930	5.8176	7.0284	0.7492
20, 0.4	4.4610	6.2550	7.3900	0.5257

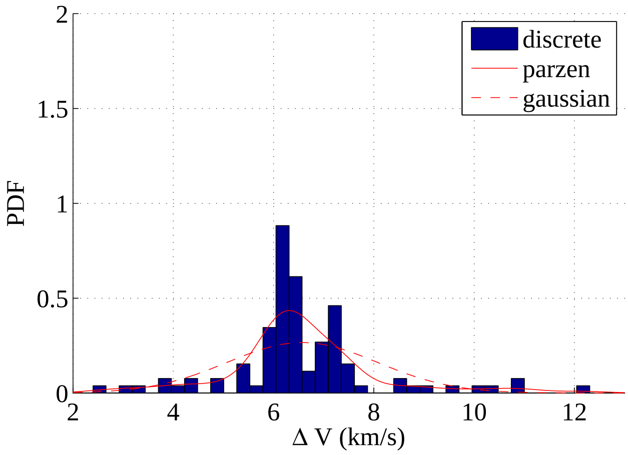

Fig. 12. Distributions of best results obtained by AIDEA ($n_{pop} = 40$, $iun_{max} = 20$, $\delta_b = 0.1$) on Messenger case

Another feature that should be embedded into future versions of the code is the on-line learning of the currently best DE strategy. Preliminary tests show that if the algorithm is exploring a near optimal region, strategy $DE/rand/1/bin$ could be beneficial to escape local structures and converge to the optimal point. But, again, if strategy $DE/rand/1/bin$ is used when the AIDEA is exploring the big basin with high ΔV's, many function evaluations are spent to converge into the non optimal basin.

V. CONCLUSIONS

This paper has introduced an adaptive version of IDEA. Some preliminary tests on real-world problems proposed for the CEC 2011 competition, have shown that the adaptive version of IDEA achieves results comparable to the best settings of the non-adaptive version. Furthermore, the combination of adaptivity and restart strategies brings the algorithm to consistently perform better than the best algorithms tested

TABLE VII

PARAMETRIC ANALYSIS FOR AIDEA PERFORMANCE ($n_{pop} = 20$) ON MESSENGER CASE - THE FIRST COLUMN CONTAINS THE USED VALUES OF iun_{max} AND δ_b

iun_{max}, δ_b	Min	$Mean$	Max	$Str.Dev.$
$3, 0.1$	3.0618	6.2296	7.4400	0.7907
$3, 0.2$	4.4258	6.4749	7.4035	0.5033
$3, 0.3$	3.5811	6.3763	8.1853	0.6909
$3, 0.4$	4.9969	6.7633	9.2208	0.6720
$5, 0.1$	2.8336	6.2347	7.5139	0.8904
$5, 0.2$	3.6415	6.2126	7.6438	0.7790
$5, 0.3$	3.6754	6.4785	7.6099	0.5969
$5, 0.4$	4.9621	6.6339	7.8255	0.5207
$10, 0.1$	3.1494	6.2401	10.8552	1.1323
$10, 0.2$	3.0285	6.0034	7.0366	0.8189
$10, 0.3$	3.8668	6.2699	7.4899	0.5861
$10, 0.4$	5.0080	6.5426	7.7897	0.4925
$20, 0.1$	3.0627	6.2828	10.9439	1.4119
$20, 0.2$	3.3786	5.8209	7.1450	0.6697
$20, 0.3$	3.5840	5.9849	7.3529	0.7348
$20, 0.4$	4.8674	6.2601	7.3455	0.4910

TABLE VIII

PARAMETRIC ANALYSIS FOR AIDEA PERFORMANCE ($n_{pop} = 10$) ON MESSENGER CASE - THE FIRST COLUMN CONTAINS THE USED VALUES OF iun_{max} AND δ_b

iun_{max}, δ_b	Min	$Mean$	Max	$Str.Dev.$
$3, 0.1$	3.0607	6.4858	8.6385	0.8691
$3, 0.2$	3.9232	6.4844	9.2189	0.8424
$3, 0.3$	3.3621	6.7159	9.5182	0.7731
$3, 0.4$	5.4851	6.9356	8.1869	0.5747
$5, 0.1$	4.0149	6.3407	8.7837	0.8755
$5, 0.2$	3.1930	6.3255	8.3996	0.7947
$5, 0.3$	4.0601	6.6772	8.2935	0.7209
$5, 0.4$	4.1606	6.7243	8.2624	0.7036
$10, 0.1$	2.6851	6.0550	7.8807	1.0203
$10, 0.2$	3.4241	6.0912	7.4743	0.7217
$10, 0.3$	4.3824	6.4754	8.1999	0.6169
$10, 0.4$	4.7132	6.7901	7.8223	0.5189
$20, 0.1$	2.5044	6.1944	9.6058	1.4632
$20, 0.2$	3.1834	5.8817	7.0045	0.7866
$20, 0.3$	3.9548	6.1134	7.4883	0.6643
$20, 0.4$	3.7572	6.4462	8.0238	0.6797

on the CEC2011 competition.

The sensitivity analysis on the most difficult problem, the Messenger mission, has shown that the on-line adaptation of the parameters regulating the local restart procedure is a crucial aspect. Furthermore, a clever adaptation of DE strategy could better balance convergence and exploration especially in cases, like the Messenger problem, where the structure of the landscape changes radically when approaching lower values of the cost function.

REFERENCES

[1] M. Vasile, E. Minisci, M. Locatelli, “An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization,” *IEEE Trans. on Evolutionary Computation*, vol. 15, no. 2, pp. 267-281, 2011.

[2] K.V. Price, R.M. Storn, J.A. Lampinen, “Differential Evolution. A Practical Approach to Global Optimization,” *Natural Computing Series*,Springer, 2005.

[3] D.J. Wales, J.P.K. Doye, “Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms,” *J. Phys. Chem. A*, vol. 101, pp. 51115116, 1997.

[4] B.Addis, M. Locatelli, F.Schoen, “Local optima smoothing for global optimization,” *Optimization Methods and Software*, vol. 20, pp. 417-437, 2005.

[5] S. Das, P.N. Suganthan, “Differential Evolution: A Survey of the State-of-the-Art,” *IEEE Trans. on Evolutionary Computation*, vol. 15, no. 1, pp. 4-31, 2011.

[6] S. Das, P.N. Suganthan, “Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems,” *Technical Report*, 2010.

[7] J. Liu and J. Lampinen, “A fuzzy adaptive differential evolution algorithm,” *Soft Comput. A Fusion Founda. Methodol. Applicat.*, vol. 9, no. 6, pp. 448462, 2005.

[8] A.K. Qin, V.L. Huang, P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398417, 2009.

[9] M. M. Ali and A. Trn, “Population set based global optimization algorithms: Some modifications and numerical studies,” *Comput. Oper. Res.*, vol. 31, no. 10, pp. 17031725, 2004.

[10] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,”

[11] R. Storn, “Differential evolution research: Trends and open questions,” In *Advances in Differential Evolution*, U.K. Chakraborty, Ed. Berlin, Germany: Springer, pp. 132, 2008.

[12] K. Fukunaga, “Introduction to Statistical Pattern Recognition,” Academic Press, New York, 1972.

[13] M. Vasile M., P. De Pascale, “Preliminary Design of Multiple Gravity-Assist Trajectories,” *J. of Spacecraft and Rockets*, vol. 43, no. 4, 2006.

[14] *http://www.esa.int/gsp/ACT/inf/op/globopt/MessengerFull.html*

[15] S.M. Elsayed, R.A. Sarker, D.L. Essam, “GA with a New Multi-Parent Crossover for Solving IEEE-CEC2011 Competition Problems,” In: *IEEE Congress on Evolutionary Computation*, pp. 1034-1040, 2011.

[16] U. Halder, S. Das, D. Maity, A. Abraham, P. Dasgupta, “Self Adaptive Cluster Based and Weed Inspired Differential Evolution Algorithm For Real World Optimization”, In: *IEEE Congress on Evolutionary Computation*, pp. 750-756, 2011.

[17] N. Mladenovic, J. Petrovic, V. Kovacevic-Vujicic, and M. Cangalovic, “Solving spreadspectrum radar polyphase code design problem by tabu search and variable neighborhood search,” *European J. of Operational Research*, vol. 153, pp. 389-399, 2003.