Swarm and Evolutionary Computation 26 (2016) 86–96

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Regular Paper

A study of the Classical Differential Evolution control parameters

F. Peñuñuri *, C. Cab, O. Carvente, M.A. Zambrano-Arjona, J.A. Tapia

Facultad de Ingeniería, Universidad Autónoma de Yucatán, A.P. 150, Cordemex, Mérida, Yucatán, Mexico

ARTICLE INFO

Article history:
Received 23 December 2014
Received in revised form
28 July 2015
Accepted 28 August 2015
Available online 12 September 2015

Keywords:
Differential evolution
Control parameters
Self-adaptive
Strategy
Complexity

ABSTRACT

An extensive numerical study has been conducted to shed some light on the selection of parameters for the Classical Differential Evolution (DE/rand/1/bin) optimization method with the dither variant. It is well known that the crossover probability (C_r) has an active role in the convergence of the method. Our experiments show that even when the number of generations needed to achieve convergence as a function of the C_r parameter is of a stochastic nature, in some regions a reasonably well defined dependence of this number as a function of C_r can be observed. Motivated by this result, a self-adaptive DE methodology has been proposed. This new methodology applies the DE/rand/1/bin strategy itself to find a good value for the C_r parameter. Regarding the population size parameter, a phenomenological study involving the search space, the tolerance error, and the complexity of the function has been made. The proposed methodology has been applied to 10 of the most common test functions, giving the best success rate (100% in all the studied examples) and in general a faster convergence than the classical DE/rand/1/bin strategy.

1. Introduction

There are good reasons why the Differential Evolution optimization algorithm is so commonly used (see, for example, [1] for a survey of the state-of-the-art of the DE method; [2] for a survey of the state-of-the-art on Real-parameter evolutionary multimodal optimization methods and [3] for a literature review on Memetic algorithms). However, as with practically all metaheuristic optimization methods, DE possesses certain parameters that affect (in many cases negatively) its convergence. One must frequently spend hours trying to find which set of parameters are the most appropriate for a specific problem. The control parameters of the classical DE method are the crossover probability C_r, the mutation scale factor F, and the population size N_p. The number of generations (g_{max}) is not considered as a control parameter because one can implement some stopping criteria. Nevertheless, it would be of great help to have an estimate of this number in order to prevent a very long running time of the program. Empirical recipes are frequently used to choose these parameters. For example, in [4], the values from $5D$ to $10D$, with D being the dimensionality of the objective function, are suggested for N_p. In [5], those values are extended from $2D$ to $40D$. Regarding C_r, most authors suggest values on the order of 0.1 and 0.9 [1,4–6]. The mutation scale factor is a real number greater than or equal to zero, although in the literature, values greater than one are rarely chosen [7–9]. However, for the dither variant, the parameter F is chosen randomly. In our study, F is chosen as a random number in [0,1). An interesting study where F is chosen randomly in the interval $(0.5, 1)$ is presented in [10].

A possible solution to the problem of finding a good set of control parameters consists in implementing an algorithm which maintains the essentials of the DE algorithm but is capable of adapting or self-adapting the control parameters. Following Eiben's definition [11,12], by adaptation we will understand that the control parameters are changed during evolution based on some feedback from the search. On the other hand, self-adaptation will imply that the parameters will evolve during the evolution through an inclusion into the genetic encoding that is itself subjected to the evolutionary process.

The idea of adaptation and/or self-adaptation (AD/SA) for evolutionary programs was introduced in [13]. A survey of operators and strategy parameters for self-adaptation in evolutionary algorithms can be found in [14]. Also there is the recommended reference [15], where some adaptive and self-adaptive strategies are outlined.

Despite the considerable number of papers on AD/SA strategies applied to DE, it is somehow strange that, as far as we know, there is no paper using the DE method itself to implement the concept of AD/SA. Nevertheless, a self-adaptive methodology can be implemented by constructing a new objective function (functional to be more precise) depending on the original objective function to optimize and

* Corresponding author. Tel.: +52 999 9300550x1051,1052.
E-mail addresses: francisco.pa@uady.mx (F. Peñuñuri), cesar.cauich@uady.mx (C. Cab), osvaldo.carvente@uady.mx (O. Carvente), miguel.zambrano@uady.mx (M.A. Zambrano-Arjona), jorge.tapia@uady.mx (J.A. Tapia).

http://dx.doi.org/10.1016/j.swevo.2015.08.003
2210-6502/

the C_r parameter. Thus, applying the DE methodology to this new objective function the C_r parameter will evolve to the best C_r value.

In this study, after a considerable amount of statistics, we present a self-adaptive methodology for the DE/rand/1/bin method with the dither variant. Also, a phenomenological justification for the N_p parameter has been proposed. This justification takes into account the volume of the search space, the tolerance error, and the complexity of the function to optimize. It is plausible that N_p should take into account these concepts. However, it is not clear how to develop a quantitative dependence. Our results shows that the Shannon entropy [16] can be used to quantify the complexity of a function in order to have a reasonable expression for N_p. Finally we present the analysis for 10 test functions frequently used for optimization.

2. The DE/rand/1/bin method

The DE method basically has four operators[1]: the population initialization operator, the mutation operator, the crossover operator, and the selection operator. For the sake of conciseness, we will introduce the following notation. The population initialization operator will be denoted by $\mathbf{X}_0(Np, \mathbf{b}_L, \mathbf{b}_U)$, where the vectors $\mathbf{b}_L$ and $\mathbf{b}_U$ are the lower (upper) parameter limits; the mutation operator by $\mathbf{V}(X, \mathbf{b}_L, \mathbf{b}_U, F)$, where X is a matrix containing the individuals (the population) to be mutated; the crossover operator by $\mathbf{U}(X, V, C_r)$, where V is a matrix containing the mutated individuals; and the selection operator by $\mathbf{S}(U, X, fob(\mathbf{x}))$, where U is a matrix containing individuals after mutation and crossover. The selection is made according to certain criteria determined by the objective function $fob(\mathbf{x})$. In practice, and since not all the mutated individuals will always participate (this will depend on the C_r parameter) in the crossover process, the mutation and crossover operations are made simultaneously. So it is convenient to define a new operator, let us call it the muta-crossover operator, which we will denote by $\mathrm{UM}(X, \mathbf{b}_L, \mathbf{b}_U, C_r, F)$. The specific form of these operators can be found elsewhere in the literature, in particular in [8]. In the Appendix, we present its Fortran code.

Keeping in mind the operators just defined, and defining the operator $\mathbf{B}(X, fob)$, which extracts the best individual $\mathbf{x}_{best}$ (that is, the individual for which fob is optimal) from the matrix X, a pseudocode for the DE method is shown in Algorithm 1.

Algorithm 1. Classical DE.

$X \leftarrow \mathbf{X}_0(Np, \mathbf{b}_L, \mathbf{b}_U)$
While (the stopping criterion (sc) has not been met) **do**
  $U \leftarrow \mathrm{UM}(X, \mathbf{b}_L, \mathbf{b}_U, C_r, F)$
  $X \leftarrow \mathbf{S}(X, U, fob)$
end while
$\mathbf{x}_{best} \leftarrow \mathbf{B}(X, fob)$

For the case when the dither variant is used, the F parameter is randomly chosen, so the only parameters will be Np and C_r. Usually the stopping criteria are the maximum number of generations and/or an acceptable error.

3. Experimental tests for the DE/rand/1/bin method

Before presenting the self-adaptive methodology we propose, we will show the results of some experiments where the DE/rand/1/bin method with the dither variant is used.

3.1. Size of the population

First, let us find a plausible formula for N_p. Suppose that $\mathbf{x}_{min}$ is the point at which a function $f(\mathbf{x})$ has a minimum (we will talk about a minimum but it is clear that the same discussion is also valid for a maximum). Let $\mathcal{V} = \mathcal{V}(\mathbf{b}_L, \mathbf{b}_U)$ be the D-dimensional volume of the search space and let $B(\mathbf{x}_{min}, \delta)$ be a D-dimensional sphere centered in $\mathbf{x}_{min}$ of radius δ and volume $\mathcal{V}_B$. Then, according to the frequentist approach to probability, the probability to throw in a random point $\mathbf{x}$ inside $B(\mathbf{x}_{min}, \delta)$ is

$$P(|\mathbf{x}_{min} - \mathbf{x}| < \delta) = \mathcal{V}_B/\mathcal{V}, \tag{1}$$

if we throw in N_p random points (which will be the initial population) in $\mathcal{V}$, the expected number of points N_s lying in $B(\mathbf{x}_{min}, \delta)$ is

$$N_s = N_p \frac{\mathcal{V}_B}{\mathcal{V}}. \tag{2}$$

Let us think of δ as the maximum distance from $\mathbf{x}_{min}$ for which the DE method can achieve a good[2] success rate. Now if the volume of the search space is increased while δ is kept fixed, we need to increase N_p in order that N_s does not become very small (which would mean that the method could not achieve convergence). Consequently, if $\mathcal{V} \to \infty$, then also $N_p \to \infty$. If we want the method to have practical applications, we could demand a divergence in N_p proportional to $\mathcal{V}$. Nevertheless, if we really want the method to be a good method, a logarithmic divergence is acceptable. From this digression, we will demand that N_p be proportional to the logarithm of the volume of the search space. Taking k as the proportionality constant, we have $N_p = k \ln \mathcal{V}$. Now, if the search space is a hypercube with side lengths $x_U - x_L$, we will have $N_p = kD \ln(x_U - x_L)$. So, for $x_U - x_L \sim e$, we have obtained a phenomenological justification for the empirical values suggested in [4,5]. Due to the fact that $x_U - x_L$ can be less than or equal to 1, we will add the constant e to the formula for N_p, thus

$$N_p = kD \ln(x_U - x_L + e) \tag{3}$$

will be the population size that we will take in our examples.

One could argue that all that is done in Eq. (3) is to replace one parameter by another, namely N_p by k. Unfortunately, the fact that in Eq. (3) the search space is considered is not a sufficient reason to say that we have a good expression for N_p. This situation can change if we find a way to choose k. In the next section we present a phenomenological reasoning for obtaining this parameter.

3.2. Results of the experimental tests

The functions we studied are shown in Table 1. Each function was optimized 10 000 times, choosing (in each optimization) a random value for C_r and for three different sizes of the population: $k = 5, 10, 15$. The stopping criteria used were a maximum number of generations chosen as $g_{max} = 2500\, N_p$ or when an acceptable error, given in the last column of Table 1, was achieved. The value used for g_{max} is completely arbitrary. We chose a relatively large value for g_{max} because we are interested in studying the convergence of the method as a function of C_r. Of course, in real-world problems, this will make the method impractical but in the next section we propose a better value for this parameter. The calculations were made using 50 AMD Opteron processors @2.3 GHz, running SUSE GNU/Linux and using the Intel® Fortran Compiler. Tables 2, 3 and 4 summarize the main results. For the cases when the method converges, we have defined $g_{s;min(max)}$ as the minimum (maximum) number of generations used to achieve the acceptable error. Also we defined $\overline{g}_s$ and

[1] Think of an operator as a general mapping, not the technical definition of operator, since the space on which they act and the image space are not, in general, the same.

[2] At this point, how good is not relevant, but if the success rate is not good enough the only thing we need to do is to take a smaller δ. Also the reader should keep in mind the frequentist approach to probability in all the discussion.