Applied Soft Computing 9 (2009) 738–745

Contents lists available at ScienceDirect

Applied Soft Computing

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

A hybrid CMA-ES and HDE optimisation algorithm with application to solar energy potential

Jérôme Henri Kämpf *, Darren Robinson

Solar Energy and Building Physics Laboratory, Station 18, Ecole Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 17 August 2007
Received in revised form 4 September 2008
Accepted 28 September 2008
Available online 17 October 2008

Keywords:
Optimisation
Covariance matrix adaptation evolution strategy (CMA-ES)
Differential evolution (DE)
Hybrid differential evolution (HDE)
Hybrid evolutionary algorithm
Hybrid CMA-ES/HDE
Urban planning
Solar energy potential

ABSTRACT

This paper describes the results of initial experiments to apply computational algorithms to explore a large parameter space containing many variables in the search for an optimal solution for the sustainable design of an urban development using a potentially complicated fitness function. This initial work concentrates on varying the placement of buildings to optimise solar irradiation availability. For this we propose a hybrid of the covariance matrix adaptation evolution strategy (CMA-ES) and hybrid differential evolution (HDE) algorithms coupled with an efficient backwards ray tracing technique. In this paper we concentrate on the formulation of the new hybrid algorithm and its testing using standard benchmarks as well as a solar optimisation problem. The new algorithm outperforms both the standalone CMA-ES and HDE algorithms in benchmark tests and an alternative multi-objective optimisation tool in the case of the solar optimisation problem.

1. Introduction

With increasing pressure to reduce the risks of climate change due to fossil fuel combustion, there is growing interest in optimizing the design of urban settlements for the exploitation of solar radiation both to reduce energy demands for heating and artificial lighting and to convert into heat (for hot water) and electricity. But past optimisation studies [1–7] have been based on manual trial and error, testing a limited number of subjectively conceived propositions using either physical or numerical tools. Far more powerful would be the coupling of computational algorithms with numerical ray tracing techniques to search the available parameter space for an optimum. Moreover the ray tracing tool should be capable of efficiently predicting the integral of solar irradiance received on built surfaces throughout a given time period (seasonal or annual). In this paper we describe how we have achieved these objectives. First we discuss the principles of optimisation algorithms and briefly review some of the different approaches that have thus far been explored, before selecting two specific algorithms (covariance matrix adaptation algorithm (CMA-ES) and hybrid differential evolution (HDE)) which we proposed to hybridise. We then describe the procedure we have followed in performing this hybridisation before finally testing the new algorithm using standard benchmark tests (the Ackley and Rastrigin functions) and a solar energy potential optimisation problem. As part of this latter we describe the method employed to efficiently solve for long-term solar irradiation simulation.

1.1. Optimisation algorithms

In all generality, we want to find the global maximum (or maxima) of a function f that depends on n independent decision variables. In formal terms, we are looking for the supremum (set of variables that maximises the function) as in the following equation:

$$sup\ \{f(\vec{x})|\vec{x}\in M\subseteq\mathbb{R}^n\} \quad (1)$$

with:

$n\in\mathbb{N}$
dimension of the problem
$f: M\rightarrow\mathbb{R}$
objective function
$M=\{\vec{x}\in\mathbb{R}^n|g_j(\vec{x})\geq 0,\ \forall\, j\in\{1,\ldots,m\}\}, M\neq\varnothing$

* Corresponding author. Tel.: +41 21 693 45 47; fax: +41 21 693 27 22.
E-mail address: jerome.kaempf@epfl.ch (J.H. Kämpf).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2008.09.009

feasible region

$m \in \mathbb{N}$

number of constraints

The set of inequality restrictions $g_j : \mathbb{R}^n \to \mathbb{R}, \forall j \in \{1, \ldots, m\}$ includes a special case of constraints due to the domain boundaries $L_i \leq x_i \leq H_i, \forall i = 1, \ldots, n$. $\vec{L}$ is named the lower bound and $\vec{H}$ the upper bound of the domain.

When the function to optimise exhibits a non-differentiable, non-linear behaviour and is multi-modal, it justifies the choice of heuristic methods such as evolutionary algorithms (EA). EA is a family of optimisation methods based upon the principles of Darwinian natural selection [8–10]. They are population-based heuristic algorithms, where each individual represents a potential solution of the function to optimise. The operators of recombination, mutation and selection are applied to get closer to a best solution through population generations.

The first evolutionary algorithms proposals come from the mid-60s with genetic algorithms (GAs) from Holland [11] at the University of Michigan, evolutionary programming (EP) from Fogel [12] at the University of California in San Diego and independently evolution strategies (ES) from Rechenberg [13] at the Technical University of Berlin. Their work brought a wide class of optimisation methods for difficult problems where few is known about the underlying search space. Koza [14], with the development of genetic programming (GP) beginning of 1990s, enriched the trend of evolutionary algorithms.

Facing a multitude of methods and approaches, we have made an arbitrary choice of focusing our attention to the evolution strategies branch of evolutionary algorithms. Evolution strategies were first developed by Rechenberg [13] and Schwefel [15] and have evolved into the cumulative step-path adaptation algorithm (CSA-ES) [16,17] and the CMA-ES [18,19]. The variables of the function to optimise are coded using a floating-point representation and are associated in phenotypes with standard deviations for mutation purpose. CMA-ES have been used to solve many optimisation problems [19] and are regarded as one of the best algorithm for real-value coded variables. In [19] Hansen and Kern concludes that CMA-ES is outperformed by differential evolution (DE) only if the function to optimise is additively separable.

Differential evolution was developed by Storn and Price in 1996 [20] and has proved to be another good candidate for real-value optimisation problem solving. DE is very simple to implement and relies only on variables with a floating point representation. The method is based on stochastic search; one of its drawbacks is the need for a large population to overcome local optima. Chang et al. [21] developed a hybrid algorithm of differential evolution to allow the use of a smaller population. However, for all kinds of DE, the results are very sensitive to the algorithm control parameters [22]. They must be well chosen for a good performance.

From Hansen and Kern's conclusion [19] we considered that a hybrid CMA-ES/HDE algorithm might combine the advantages of the two optimisation methods, since in real life applications we tend to face optimisation problems where the dependence of the function on its variable is unknown. It might be a good compromise in terms of robustness and convergence speed, as it should perform well on additively and non-additively separable functions.

2. Hybrid CMA-ES and HDE algorithm

In this section, we start with a general description of EA, a brief introduction to the CMA-ES and the HDE algorithms, and continue by describing the proposed hybridisation of the two methods (detailed descriptions can be found for CMA-ES in [18,19], for DE in [20,23] and HDE in [21]).

2.1. Evolutionary algorithms

A population of μ individuals, each representing a potential solution within the domain boundaries, is randomly chosen as a starting point. The population goes through three operators to evolve: recombination between individuals, random mutation of their alleles and selection of the fittest. One iteration of the strategy is a step from a population P^n to P^{n+1}, where n is the generation number, and can be written as

$$P^{n+1} := o\,pt_{\mathrm{EA}}(P^n) \tag{2}$$

The optimisation of P^n is defined by the operators *sel* (selection), *mut* (mutation) and *rec* (recombination) in the following way:

$$o\,pt_{\mathrm{EA}} := sel \circ (mut \circ rec)^{\lambda} \tag{3}$$

where λ corresponds to the number of new individuals (children).

According to the type of EA, a phase of adaptation of the parameters or migration of individuals might follow from their selection. The termination criterion for each iteration is met when the maximum number of function evaluations is reached. Since this is roughly proportional to the total computing time, we are able to define an (approximate) upper limit of time required for the optimisation process. When reached, the algorithm exits and returns the best individual.

2.2. Covariance matrix adaptation evolution strategy

Each individual in the population $P = \{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_\mu\}$ referred to by an index $k = 1, \ldots, \mu$ has a phenotype $\vec{a}_k = (\vec{x}_k, \vec{z}_k)$ with $\vec{x}_k, \vec{z}_k \in \mathbb{R}^n$, where $\vec{x}_k$ is the standard ES parameter vector and $\vec{z}_k$ is the associated standard deviation vector. Three matrices are needed for the algorithm: the covariance matrix $C \in \mathbb{R}^{n \times n}$, the eigenvector matrix of C named $B \in \mathbb{R}^{n \times n}$ and the diagonal matrix of the square rooted eigenvalues of C named $D \in \mathbb{R}^{n \times n}$. The μ individuals of the initial population are randomly defined ($\vec{x}_k$ are randomly chosen within the domain boundaries of f and $\vec{z}_k$ are set to the null vector). Matrix B is set to the identity matrix, the diagonal matrix D is set to represent the domain boundaries $D_{ii} = H_i - L_i, \forall i = 1, \ldots, n$. C is calculated as the product of BD and its transpose: $BD \cdot (BD)^t$.

2.2.1. Recombination

Using the global weighted intermediate recombination method in conjunction with a sorted population (the best individual is number 1, the worst is μ), λ identical children are created with a phenotype:

$$(\vec{x}_h)_i = \sum_{k=1}^{\mu} \omega_k \cdot (\vec{x}_k)_i, \quad \forall i = 1, \ldots, n \tag{4}$$

$$(\vec{z}_h)_i = 0, \quad \forall i = 1, \ldots, n \tag{5}$$

in which the individual index h goes from $(\mu + 1)$ to $(\mu + \lambda)$ and ω_k are the weights of the recombination, which are themselves parameters of the algorithm. In this study we take $\omega_k = (\log(\mu + 1) - \log(k)) / \sum_{l=1}^{\mu} (\log(\mu + 1) - \log(l))$ from [18], which gives more weight to the best individuals of the population.

2.2.2. Mutation

The main mechanism of the implemented operator is changing the allele values by adding random noise drawn from a normal distribution. The randomness from the normal distribution is stored in the individual phenotype and used in the adaptation phase. The mutation acts on each of the λ children with a modification of their