Progress in Nuclear Energy 70 (2014) 140–148

Contents lists available at ScienceDirect

Progress in Nuclear Energy

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

Differential evolution with topographical mutation applied to nuclear reactor core design

Wagner F. Sacco [a,*], Nélio Henderson [b]

[a] Instituto de Engenharia e Geociências, Universidade Federal do Oeste do Pará, Rua Vera Paz, s/n, Salé, Santarém, PA 68135-110, Brazil
[b] Depto. de Modelagem Computacional, Instituto Politécnico, Universidade do Estado do Rio de Janeiro, R. Bonfim, 25, Nova Friburgo, RJ 28625-570, Brazil

ARTICLE INFO

Article history:
Received 26 July 2013
Received in revised form
20 September 2013
Accepted 25 September 2013

Keywords:
Reactor core design optimization
Evolutionary computation
Differential evolution
Mutation operator
Topographical heuristic

ABSTRACT

The nuclear reactor core design optimization problem consists in adjusting several reactor cell parameters, such as dimensions, enrichment and materials, in order to minimize the average peak-factor in a three-enrichment-zone reactor, considering restrictions on the average thermal flux, criticality and sub-moderation. This problem is highly multimodal, requiring optimization techniques that overcome local optima, which can be done achieving a balance between the exploration of the search space and the exploitation of its most promising areas. In order to do so, we introduce a variant of the differential evolution algorithm (DE) with a new mutation operator based on a topographical heuristic introduced in the early nineties, as part of a global optimization method. The new method, called TopoMut-DE, is favorably compared against the canonical version of differential evolution, and even to state-of-the-art variants, namely Opposition-Based DE and Trigonometric-Mutation Operator DE. As the problem attacked is quite challenging, the results show the potential of TopoMut-DE to be applied to other nuclear science and engineering problems.

1. Introduction

In the process of nuclear core design, configurations consisting of materials, enrichment and dimensions have to be analyzed with neutronic models. In practice, computational codes simulate the neutrons' behavior and interaction with the materials, given a reactor core geometry. Restrictions related to the average thermal flux, criticality and sub-moderation must be satisfied in order to obtain safe core configurations. Despite the complexity of the nuclear core design process, it is possible to formulate it as an optimization problem (Pereira et al., 1999). Thus, global optimization techniques may automatically perform the search for the best core configurations regarding safety and economic matters.

Some random optimization methods have been applied to solve this nuclear core design optimization problem, among them the genetic algorithm (GA, Holland, 1992) by Pereira et al. (1999), particle swarm optimization (PSO, Kennedy and Eberhart, 1995) by Domingos et al. (2006), and differential evolution (DE, Storn and Price, 1997) by Sacco et al. (2009) and Sacco et al. (2013). This problem is highly multimodal (Sacco et al., 2004), i.e., with many local minima, being, thus a great challenge for optimization algorithms.

Because of this multimodality, the search space should be thoroughly explored so that the optimization algorithm does not converge to a local optimum. To overcome this difficulty, many solutions have been proposed: a parallel genetic algorithm (Pereira and Lapa, 2003), a niching method (Mahfoud, 1995) applied to genetic algorithms (Sacco et al., 2004), and a hybrid algorithm that alternates exploration and exploitation of the search space (Sacco et al., 2008).

Since its early stage of development, new mutation schemes have been proposed for the differential evolution algorithm (for a review, see Das and Suganthan, 2010). Let us mention some relevant contributions. Storn (1996) introduced the scheme known as DE/best/1, where the best-so-far solution is always the base vector. Fan and Lampinen (2003) presented trigonometric mutation, which is applied with a certain probability in order to speed up DE. In this mechanism, the base vector is obtained by the arithmetic mean of randomly selected solutions. Sacco et al. (2013) tested this technique. Price et al. (2005) proposed the DE/rand/1/either-or mutation, where trial vectors are either three-vector recombinants or randomly chosen population vectors to which a randomly chosen vector difference has been added. Kaelo and Ali (2006) introduced a scheme where three vectors are randomly selected, the best one is used as the base vector and the remaining two are

* Corresponding author. Tel.: +55 93 8402 6503; fax: +55 93 2101 4902.
E-mail addresses: wagner.sacco@ufopa.edu.br, wagner.sacco@pq.cnpq.br (W.F. Sacco).

0149-1970/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.pnucene.2013.09.012

used to find the difference vector. This scheme was used with success by Sacco et al. (2009). Recently, Qu et al. (2012) proposed a neighborhood mutation strategy, where the mutation is performed in each neighborhood in order to maintain diversity.

In this paper, we propose a new mutation operator for DE in order to promote a greater balance between the exploration of the search space and the exploitation of its most promising regions than in the canonical version of the algorithm: the topographical mutation. This mutation scheme employs a clustering heuristic based on the topographical information on the objective function, which was part of an optimization algorithm proposed by Törn and Viitanen (1992), the Topographical Algorithm (TA). We use this heuristic with the purpose of determining the best individual (i.e., with the lowest objective function value) in a region, so that it serves as the base vector for the mutation of the nearby individuals. Originally, Törn and Viitanen (1992) used this mechanism to determine minima from a set of sampled points, so that they were initial solutions for a local optimization algorithm. The reader must be aware that the new method proposed here is completely different from the Topographical Differential Evolution created by Ali and Törn (2000). In their DE variant, these authors used the topographical heuristic to obtain minima that were used as initial solutions for local searches, just like in the TA.

The remainder of the paper is described as follows. The nuclear reactor core design is presented in Section 2. The description of the canonical DE algorithm is presented in Section 3. The new DE variant, TopoMut-DE, is introduced in Section 4. The computational experiments and their discussions are in Section 5. Finally, the conclusions are made in Section 6.

2. The nuclear reactor core design problem

Let us describe the optimization problem (for a more detailed exposition, see Pereira et al., 1999): consider a cylindrical 3-enrichment-zone reference reactor, with height $h = 1.63$ m, and zones with radius $R_1 = 0.86$ m and thicknesses $R_2 = 0.38$ m and $R_3 = 0.18$ m, with enrichments E_1, E_2, and E_3, respectively. This reactor's typical cell is composed by moderator (light water), cladding and fuel. Fig. 1 illustrates such reactor. The design parameters that may be varied in the optimization process, as well as their variation ranges, are shown in Table 1. The materials are represented by discrete variables.

The objective of the optimization problem is to minimize the average flux or power peaking factor, f_p, of the proposed reactor, allowing the reactor to be sub-critical or super critical ($k_{\text{eff}} = 1.0 \pm 1\%$), for a given average flux ϕ_0. Let $\mathbf{X} = \{R_f, \Delta_c, \Delta_m, E_1, E_2, E_3, M_f, M_c\}$ be the vector of design variables. Then, the optimization problem may be written as.

Table 1
Range of parameters.

Parameter	Symbol	Range
Fuel radius (cm)	R_f	0.508 to 1.270
Cladding thickness (cm)	Δ_c	0.025 to 0.254
Moderator thickness (cm)	R_e	0.025 to 0.762
Enrichment of zone 1 (%)	E_1	2.0 to 5.0
Enrichment of zone 2 (%)	E_2	2.0 to 5.0
Enrichment of zone 3 (%)	E_3	2.0 to 5.0
Fuel material	M_f	{U-Metal or UO_2}
Cladding material	M_c	{Zircaloy-2, Aluminum or Stainless steel-304}

Minimize
$f_p(\mathbf{X})$
Subject to:

$$\phi(\mathbf{X}) = \phi_0; \tag{1}$$

$$0.99 \le k_{\text{eff}}(\mathbf{X}) \le 1.01; \tag{2}$$

$$\frac{dk_{\text{eff}}}{dV_m} > 0; \tag{3}$$

$$X_i^l \le X_i \le X_i^u, i = 1, 2, \ldots, 6 \tag{4}$$

$$M_f = \{UO_2 \text{ or } U - \text{metal}\};; \tag{5}$$

$$M_c = \{\text{Zircaloy} - 2, \text{Aluminium or Stainless Steel} - 304\},, \tag{6}$$

where V_m is the moderator volume, and the superscripts l and u indicate respectively the lower and upper bounds (of the feasible range) for each design variable.

The HAMMER system (Suich and Honeck, 1967) was used for cell and diffusion equations calculations. It performs a multigroup calculation of the thermal and epithermal flux distribution from the integral transport theory in a unit cell of the lattice (Duderstadt and Hamilton, 1976),

$$\phi(\mathbf{r}) = \int_V \frac{e^{-\sum_t |\mathbf{r}-\mathbf{r}'|}}{4\pi|\mathbf{r}-\mathbf{r}'|^2} \cdot S(\mathbf{r}') \cdot d^3\mathbf{r}'. \tag{7}$$

The integral transport equation for scalar flux $\phi(\mathbf{r})$, where $\mathbf{r}$ is the position vector, is solved for all sub-regions of the unit cell, being the neutron source $S(\mathbf{r})$ isotropic into the energy group under consideration. The transfer kernel in Equation (7) is related to the collision probabilities for a flat isotropic source in the initial region.

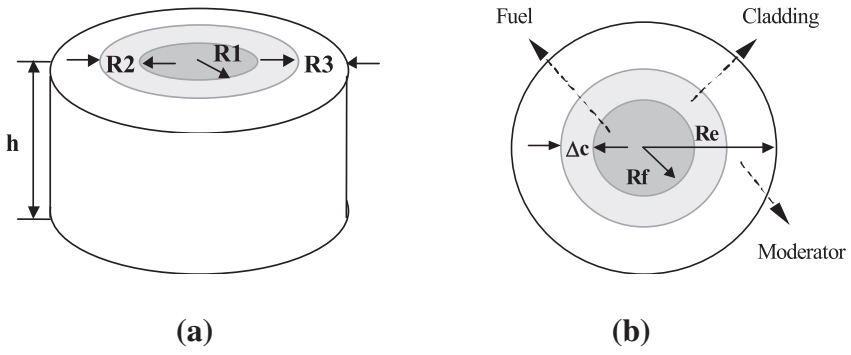

Fig. 1. (a) The nuclear reactor and (b) its typical cell.

The solution is initially performed for a unit cell in an infinite lattice. The integral transport calculation is followed by a multigroup Fourier transfer leakage spectrum theory in order to include the leakage effects in the previous calculation and to proceed with the multigroup flux-volume weighting.

Using the four group constants obtained from the mentioned procedure, a one-dimensional multi-region reactor calculation is performed. The diffusion equation (Duderstadt and Hamilton, 1976) is, then, solved to perform standard criticality calculation,

$$-\vec{\nabla} D_g(\mathbf{r})\vec{\nabla}\phi_g(\mathbf{r})+\Sigma_{t,g}(\mathbf{r})\phi_g(\mathbf{r})=\sum_{g'=1}^{4}\left[\frac{1}{k_{\text{eff}}}\chi_{g'}\Sigma_{fg'}(\mathbf{r})+\Sigma_{sg'g}(\mathbf{r})\right]\phi_{g'}(\mathbf{r}), \tag{8}$$

where $\mathbf{r}$ is the position vector; D_g is the diffusion coefficient for group g;ϕ_g is the neutron flux for group g; $\sum_{t,g}$ is the total group cross section for group g; k_{eff} is the effective multiplication factor; $\chi_{g'}$ is the group fission spectrum for group g'; $\sum_{fg'}$ is the fission group cross section for group g'; $\sum_{sg'g}$ is the scattering cross section form group g' to g, and $\phi_{g'}$ is the neutron flux for group g'.

The flux $\phi_g(\mathbf{r})$ is calculated assuming normalized source density. Equation (8) is solved using the finite difference method and a computational mesh with constant spacing in the spatial coordinate.

The fitness function is given by

$$f=\begin{cases} f_p, & \Delta k_{eff}\le 0.01;\Delta\phi\le 0.01\phi_0;\frac{\Delta' k_{eff}}{\Delta Vm}>0\\ f_p+r_1\cdot\Delta k_{eff}, & \Delta k_{eff}>0.01;\Delta\phi\le 0.01\phi_0;\frac{\Delta' k_{eff}}{\Delta Vm}>0\\ f_p+r_2\cdot\Delta\phi, & \Delta k_{eff}\le 0.01;\Delta\phi>0.01\phi_0;\frac{\Delta' k_{eff}}{\Delta Vm}>0\\ f_p+r_3\cdot\frac{\Delta' k_{\text{eff}}}{\Delta Vm}, & \Delta k_{\text{eff}}\le 0.01;\Delta\phi\le 0.01\phi_0;\frac{\Delta' k_{\text{eff}}}{\Delta Vm}<0\\ f_p+r_1\cdot\Delta k_{\text{eff}}+r_2\cdot\Delta\phi, & \Delta k_{\text{eff}}>0.01;\Delta\phi>0.01\phi_0;\frac{\Delta' k_{\text{eff}}}{\Delta Vm}>0\\ f_p+r_1\cdot\Delta k_{\text{eff}}+r_3\cdot\frac{\Delta' k_{\text{eff}}}{\Delta Vm}, & \Delta k_{\text{eff}}>0.01;\Delta\phi\le 0.01\phi_0;\frac{\Delta' k_{\text{eff}}}{\Delta Vm}<0\\ f_p+r_2\cdot\Delta\phi+r_3\cdot\frac{\Delta' k_{\text{eff}}}{\Delta Vm}, & \Delta k_{\text{eff}}\le 0.01;\Delta\phi>0.01\phi_0;\frac{\Delta' k_{\text{eff}}}{\Delta Vm}<0\\ f_p+r_1\cdot\Delta k_{\text{eff}}+r_2\cdot\Delta\phi+r_3\cdot\frac{\Delta' k_{\text{eff}}}{\Delta Vm}, & \Delta k_{\text{eff}}>0.01;\Delta\phi>0.01\phi_0;\frac{\Delta' k_{\text{eff}}}{\Delta Vm}<0 \end{cases} \tag{9}$$

and was developed in such a way that, if all constraints are satisfied, it has the value of the average peak factor, f_p, otherwise, it is penalized proportionally to the discrepancy on the constraint. Such penalization factors should be setup by the expert, according to the requirements and the priorities of the problem, being weighted by the coefficients r_i, with $i = 1, 2, 3$. As in Pereira et al. (1999), we use $r_1 = r_2 = r_3 = 10$, $\Delta k_{\text{eff}} = |.1.0 - k_{\text{eff}}|.$, $\Delta\phi = |.\phi - \phi_0|.$, and$\Delta' k_{\text{eff}}/\Delta V_m = (k_{\text{eff}} - k'_{\text{eff}})/(0.03V_m)$.

3. The differential evolution algorithm

In its canonical version, introduced by Storn and Price (1997), DE is applied to the minimization of an objective function $f(\mathbf{x})$, where $\mathbf{x}$ is a continuous variable vector with domain $[\mathbf{low}, \mathbf{up}] \subset \mathbb{R}^n$. The algorithm is outlined in Fig. 2 and its operators initialization, mutation, crossover and selection are respectively described in Figs. 3–6. The input parameters, which remain constant along the optimization process, are the population size NP and, to be explained below, the crossover rate CR and the scaling factor *F*.

First of all, as seen in Fig. 3, an initial population is generated with each component j of each individual i initialized in the range

Data: NP, CR, F.

Result: Optimal solution $\mathbf{x}^*$, $f(\mathbf{x}^*)$.

begin

  $NIter \leftarrow 0$

  $f(\mathbf{x}_i^*) \leftarrow 1.0\text{E}6$

  `initialize()`

  repeat

    $NIter \leftarrow NIter + 1$

    `mutate()`

    `crossover()`

    `select()`

  until *a termination criterion is satisfied*

end

Fig. 2. The DE algorithm.

$[\text{low}_j,\text{up}_j]$. In its canonical version, the individuals are generated at random. Each initial solution or individual must meet the boundary constraints.

Afterward, inside a loop, the evolutionary process starts until a stopping criterion is satisfied. The first operation inside the loop is mutation, described by function "mutate", in Fig. 4. In this operation, a trial solution, or perturbed individual $\widehat{\mathbf{x}}_i$, is generated for each individual i as in

$$\widehat{\mathbf{x}}_i = \mathbf{x}_{p(1)} + \text{F}\left(\mathbf{x}_{p(2)} - \mathbf{x}_{p(3)}\right), \tag{10}$$

where $p(1)$, $p(2)$, and $p(3)$ are random indexes mutually different from each other and different from index i, and F is a scaling factor in the range [0,2]. The solution correspondent to the first random index, $\mathbf{x}_{p(1)}$, is known as the base vector, or "donor". This vector is modified by the addition of the weighted difference of the two other solutions with indexes $p(2)$ and $p(3)$. The operation is repeated as long as trial solution $\widehat{\mathbf{x}}_i$ is outside the domain.

After mutation, the population goes through crossover, as in Fig. 5. In this operation, component j of offspring $\mathbf{y}_i$ is found from its parents $\mathbf{x}_i$ and $\widehat{\mathbf{x}}_i$ according to the rule

$$y_i^j = \begin{cases} \widehat{x}_i^j, & \text{if } R^j \le \text{CR or } j = I_i \\ x_i^j, & \text{otherwise} \end{cases}, \tag{11}$$

where I_i is a random integer in the range [1,n], R^j is a random number in [0,1], and crossover rate CR, also in [0,1], controls the fraction of parameter values that are copied from the trial solution $\widehat{\mathbf{x}}_i$. Note that the alternative $j = I_i$ assures that at least one component will receive a mutated value.

Finally, the selection process (Fig. 6) defines the population of next generation as

$$x_{\text{i}}^{\text{Niter+1}} = \begin{cases} y_{\text{i}}^{\text{Niter}}, & \text{if } \text{f}(y_{\text{i}}^{\text{Niter}}) \le \text{f}(x_{\text{i}}^{\text{Niter}}) \\ x_{\text{i}}^{\text{Niter}}, & \text{otherwise} \end{cases}. \tag{12}$$

4. The new method

4.1. The topographical algorithm

Between the early seventies and mid-nineties, a global optimization paradigm based on clustering was studied by some researchers, mainly in Europe. The seminal article by Becker and Lago

```
begin
    for i ← 1 to NP do
        for j ← 1 to n do
            /*component j of vector x_i is randomly generated inside interval
              [up_j, low_j].                                                  */
            x_i^j ← low_j+rand(0,1)*(up_j − low_j)
end
```

Fig. 3. Function initialize.

(1970) was followed by, among others, Törn (1973, 1978), Timmer (1984), Törn and Viitanen (1992, 1994), and Ali and Storey (1994). Ali (1994), and Levi and Haas (2010) present fine reviews on the clustering methods. According to Törn and Žilinskas (1989), the motivation for exploring clustering methods in based on the following:

a. It is possible to obtain a sample of points in the search space consisting of concentration of points in the neighborhood of local minimizers of the objective function *f*.
b. The points in the sample can be clustered giving clusters identifying the neighborhoods of local minimizers and thus permitting local optimization methods to be applied.

The original TA algorithm is non-iterative and based on the exploration of the search space (Ali and Storey, 1994). It consists of three steps (Törn and Viitanen, 1994):

1. A uniform random sampling of *N* points in the search space.
2. The construction of the topograph, which is a graph with directed arcs connecting the accepted sampled points on a *k*-nearest neighbors basis, where the direction of the arc is towards a point with a larger function value. The minima of the graph are the points better than their neighbors, i.e., the nodes with no incoming arcs.
3. The topograph minima are starting points for a local optimization algorithm. The best point obtained from all the executions

```
begin
    for i ← 1 to NP do
        repeat
            p(1) ← floor(1+rand(0,1)*(NP − 1))
            /*function floor(argument) generates an integer random
              number.                                                 */
        until (p(1) ≠ i)
        repeat
            p(2) ← floor(1+rand(0,1)*(NP − 1))
        until (p(2) ≠ i ∧ p(2) ≠ p(1) )
        repeat
            p(3) ← floor(1+rand(0,1)*(NP − 1))
        until (p(3) ≠ i ∧ p(3) ≠ p(1) ∧ p(3) ≠ p(2) )
        repeat
            x̂_i ← x_p(1) + F * (x_p(2) − x_p(3))
        until x̂_i ∈ [low, up]
end
```

Fig. 4. Function mutate.

```
begin
    for i ← 1 to NP do
        I_i ← floor(1+rand(0,1)*(n − 1))
        for j ← 1 to n do
            R^j ← rand(0,1)
            if ((R^j ≤ CR) ∨ (j = I_i)) then
                y_i^j ← x̂_i^j
            else
                y_i^j ← x_i^j
end
```

Fig. 5. Function crossover.

using each minimum as the initial solution is the result of the algorithm.

Originally, Törn and Viitanen (1992, 1994) obtained the initial solutions from step 1 sampling points in a unit hypercube, until *N* points with their nearest neighbors farther than a threshold distance δ were obtained. Then, these points were denormalized. But these authors add that any other method that produces a very uniform covering can be used. In fact, they used the more efficient quasi-random sampling (Gentle, 2003; Press et al., 2007) in an iterative version of TA (Törn and Viitanen, 1996). In their tests, Törn and Viitanen (1994) used mostly $N = 100$ or $N = 200$.

Step 2, the construction of the topograph, is the heart of the method. First of all, a $N \times N$ symmetric distance matrix is computed. Following that, a $N \times k$ matrix called *kNN*-matrix is constructed containing, for each point, the indexes of its *k*-nearest neighbors sorted by distance. Next, this matrix, which is an undirected topograph, is transformed into a directed topograph indicating if the reference is to a point with larger or smaller objective function value by giving the reference a plus or minus sign, respectively (Ali, 1994). The signs represent the directed arcs in the graph, a positive sign representing the "arrow head" of the arc, and the negative sign the "start" of the arc (Törn and Viitanen, 1994). Finally, the points that correspond to rows with only positive signs are the topograph minima.

Let us illustrate how the topographical heuristic works by a simple illustrative example, adapted from Ali (1994). Suppose we want to minimize the function

$$f(x,y) = x^2 + y^2, \tag{13}$$

and that six points were sampled and their function values calculated: $f(P_1) = f(2,5) = 29$, $f(P_2) = f(1,2) = 5$, $f(P_3) = f(3,4) = 25$, $f(P_4) = f(0,1) = 1$, $f(P_5) = f(5,0) = 25$, and $f(P_6) = f(4,2) = 20$.

```
begin
    for i ← 1 to NP do
        if f(y_i) ≤ f(x_i) then
            x_i ← y_i
            if f(x_i) < f(x_i^*) then
                x_i^* ← x_i
                f(x_i^*) ← f(x_i)
end
```

Fig. 6. Function select.

First, the symmetric squared distance matrix **D** is constructed, where, for example, the element d_{13} corresponds to the distance between P_1 and P_3:

$$\mathbf{D} = \begin{bmatrix} 0 & 10 & 2 & 20 & 34 & 13 \\ 10 & 0 & 8 & 2 & 20 & 9 \\ 2 & 8 & 0 & 18 & 20 & 5 \\ 20 & 2 & 18 & 0 & 26 & 17 \\ 34 & 20 & 20 & 26 & 0 & 5 \\ 13 & 9 & 5 & 17 & 5 & 0 \end{bmatrix}. \tag{14}$$

Following that, the *kNN*-matrix is formed by each point's *k*-nearest neighbors. Using $k = 3$, the nearest neighbors of P_1 (the first row of **D**) are the points with indexes 3, 2, and 6, respectively. These elements will constitute the first row of the matrix. The process goes on until the following matrix is obtained:

$$\mathbf{kNN} = \begin{bmatrix} 3 & 2 & 6 \\ 4 & 3 & 6 \\ 1 & 6 & 2 \\ 2 & 6 & 3 \\ 6 & 2 & 3 \\ 3 & 5 & 2 \end{bmatrix}. \tag{15}$$

This matrix represents an undirected graph. Computationally, it is obtained sorting each row of **D** and taking the first *k* elements' indexes. The elements of the main diagonal of **D** receive a very large value (e.g., 10^8) before sorting, so that they are not included in the *kNN*-matrix.

Now, the elements of **kNN** will receive a plus or minus sign according to their functional values in relation to the value of the point represented by the row index. The second row, for example, corresponds to P_2, whose function value is equal to 5, which is more than $f(P_4) = 1$ (P_4 is element knn_{21}), but less than $f(P_3) = 13$ and $f(P_1) = 29$ (elements knn_{22} and knn_{23},respectively). Therefore, knn_{21} will receive a minus sign and the other two elements a plus sign. The signed matrix becomes

$$\mathbf{kNN} = \begin{bmatrix} -3 & -2 & -6 \\ -4 & +3 & +6 \\ +1 & -6 & -2 \\ +2 & +6 & +3 \\ -6 & -2 & +3 \\ +3 & +5 & -2 \end{bmatrix}. \tag{16}$$

Data: NP, CR, F, TMP.

Result: Optimal solution $\mathbf{x}^*$, $f(\mathbf{x}^*)$.

begin
- $NIter \leftarrow 0$
- $f(\mathbf{x}^*) \leftarrow 1.0E6$
- `initialize()`
- **repeat**
  - $NIter \leftarrow NIter + 1$
  - `Topograph()`
  - `mutate()`
  - `crossover()`
  - `select()`
- **until** *a termination criterion is satisfied*

end

Fig. 7. The TopoMut-DE algorithm.

As the only point that corresponds to a row with only positive signs is $P_4 = (0,1)$, this will be the starting point for a local optimization algorithm. When implementing the topographical heuristic, the signs can be attributed in the process of construction of **kNN**.

In step 3, Törn and Viitanen (1994) say that any local optimization method can be used. They employed a gradient-based algorithm, as their tests were performed on algebraic test functions.

4.2. TopoMut-DE

As mentioned in the Introduction, the purpose of this new DE variant is to promote a greater balance between exploration and exploitation than in the canonical version of the algorithm. In order to achieve this balance, besides the conventional mutation described in Fig. 4, we employ the topographical mutation with a certain probability. The traditional mutation promotes a great exploration of the search space (Price et al., 2005). The topographical mutation, by its turn, exploits a certain region of the search space, as the base vector determined by this mechanism is near the *i*th original solution to be mutated $\hat{\mathbf{x}}_i$.

TopoMut-DE is described in Fig. 7. Before mutation, function Topograph, which consists in the construction of the topograph described in the previous subsection (TA's step 2) and the determination of the nearest topograph minimum to each point, is called. The points to be clustered are the NP current individuals of DE, instead of the random sampling from the Topographical Algorithm.

The new function mutate starts as in the traditional mutation scheme, with the determination of random individuals with indexes $p(1)$, $p(2)$, and $p(3)$ in order to mutate individual $\hat{\mathbf{x}}_i$. Then, the new scheme takes place. With a certain probability TMP (which stands for Topographical Mutation Probability), $p(1)$ receives the index of the nearest topograph minimum to $\hat{\mathbf{x}}_i$, which was determined by function Topograph. Fig. 8 shows how it works.

For TMP, besides a constant value, we propose two non-constant TMPs with increasing values. The idea behind these varying

begin
- **for** $i \leftarrow 1$ ***to*** NP **do**
  - **repeat**
    - $p(1) \leftarrow$ `floor(1+`*rand*`(0,1)*(`$NP-1$`))`
    - /*function `floor(`*argument*`)` generates an integer random number. */
  - **until** $(p(1) \neq i)$
  - **repeat**
    - $p(2) \leftarrow$ `floor(1+`*rand*`(0,1)*(`$NP-1$`))`
  - **until** $(p(2) \neq i \wedge p(2) \neq p(1))$
  - **repeat**
    - $p(3) \leftarrow$ `floor(1+`*rand*`(0,1)*(`$NP-1$`))`
  - **until** $(p(3) \neq i \wedge p(3) \neq p(1) \wedge p(3) \neq p(2))$
  - **if** *rand*$(0,1) < TMP$ **then**
    - $p(1) \leftarrow$ index of the nearest best to the original individual obtained by `Topograph()`.
  - **repeat**
    - $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_{p(1)} + F * (\mathbf{x}_{p(2)} - \mathbf{x}_{p(3)})$
  - **until** $\hat{\mathbf{x}}_i \in [\mathbf{low}, \mathbf{up}]$

end

Fig. 8. The new function mutate, with topographical mutation.

Table 2
Results for ten experiments of the canonical DE algorithm.

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7021	0.1879	0.7406	2.7532	2.8217	4.9890	U-Metal	SS-304	1.2765	67,967
#2	0.6818	0.1810	0.7227	2.7500	2.8843	4.9749	U-Metal	SS-304	1.2767	24,832
#3	0.7037	0.1881	0.7417	2.7514	2.8803	4.9864	U-Metal	SS-304	1.2765	46,173
#4	0.7278	0.1953	0.7606	2.7457	2.8801	4.9964	U-Metal	SS-304	1.2763	49,570
#5	0.7042	0.1872	0.7419	2.7450	2.8735	4.9745	U-Metal	SS-304	1.2765	77,814
#6	0.6812	0.1823	0.7235	2.7604	2.8953	4.9935	U-Metal	SS-304	1.2767	22,468
#7	0.6820	0.1789	0.7224	2.7359	2.8697	4.9478	U-Metal	SS-304	1.2767	28,716
#8	0.7068	0.1839	0.7423	2.7206	2.8484	4.9301	U-Metal	SS-304	1.2766	28,410
#9	0.6886	0.1770	0.7259	2.7109	2.8434	4.9043	U-Metal	SS-304	1.2767	25,012
#10	0.6886	0.1770	0.7271	2.7138	2.8462	4.9100	U-Metal	SS-304	1.2767	25,271
Average	0.6967	0.1838	0.7349	2.7387	2.8643	4.9607	–	–	1.2766	39,623.3

From Sacco et al., 2013.

Table 3
Results for ten experiments of TopoMut-DE with TMP = 0.1.

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7026	0.1880	0.7411	2.7540	2.8829	4.9909	U-Metal	SS-304	1.2765	72,978
#2	0.7336	0.1664	0.7540	2.5691	2.6899	4.6559	U-Metal	SS-304	1.2766	38,164
#3	0.6788	0.1824	0.7215	2.7643	2.8993	4.9999	U-Metal	SS-304	1.2767	33,344
#4	0.7145	0.1838	0.7479	2.7017	2.8338	4.9071	U-Metal	SS-304	1.2765	93,441
#5	0.7062	0.1856	0.7428	2.7312	2.8591	4.9493	U-Metal	SS-304	1.2765	93,452
#6	0.7291	0.1957	0.7606	2.7435	2.8777	4.9931	U-Metal	SS-304	1.2763	51,752
#7	0.7034	0.1881	0.7416	2.7526	2.8815	4.9886	U-Metal	SS-304	1.2765	92,189
#8	0.7265	0.1940	0.7604	2.7399	2.8742	4.9833	U-Metal	SS-304	1.2763	44,829
#9	0.6823	0.1804	0.7227	2.7449	2.8790	4.9658	U-Metal	SS-304	1.2767	40,920
#10	0.7030	0.1877	0.7413	2.7504	2.8792	4.9840	U-Metal	SS-304	1.2765	91,010
Average	0.7080	0.1852	0.7434	2.7252	2.8557	4.9418	–	–	1.2765	65,207.9

Table 4
Results for ten experiments of TopoMut-DE with TMP = 0.25.

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7266	0.1946	0.7601	2.7427	2.8768	4.9890	U-Metal	SS-304	1.2763	71,202
#2	0.7298	0.1948	0.7617	2.7370	2.8710	4.9800	U-Metal	SS-304	1.2763	76,427
#3	0.7144	0.1775	0.7456	2.6675	2.7927	4.8328	U-Metal	SS-304	1.2766	78,347
#4	0.7019	0.1666	0.7315	2.6238	2.7522	4.7468	U-Metal	SS-304	1.2768	92,589
#5	0.7267	0.1949	0.7605	2.7464	2.8808	4.9970	U-Metal	SS-304	1.2763	97,491
#6	0.7270	0.1940	0.7601	2.7383	2.8726	4.9811	U-Metal	SS-304	1.2763	93,203
#7	0.7277	0.1955	0.7613	2.7456	2.8800	4.9946	U-Metal	SS-304	1.2763	76,209
#8	0.7300	0.1922	0.7613	2.7222	2.8554	4.9520	U-Metal	SS-304	1.2763	85,991
#9	0.7268	0.1932	0.7604	2.7359	2.8695	4.9767	U-Metal	SS-304	1.2763	90,026
#10	0.7264	0.1943	0.7602	2.7424	2.8766	4.9885	U-Metal	SS-304	1.2763	77,035
Average	0.7237	0.1898	0.7563	2.7202	2.8528	4.9439	–	–	1.2764	83,852.0

Table 5
Results for ten experiments of TopoMut-DE with TMP = 0.5.

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7240	0.1946	0.7581	2.7476	2.8818	4.9976	U-Metal	SS-304	1.2763	39,019
#2	0.7275	0.1953	0.7610	2.7450	2.8794	4.9937	U-Metal	SS-304	1.2763	25,831
#3	0.6810	0.1817	0.7225	2.7563	2.8909	4.9863	U-Metal	SS-304	1.2767	73,297
#4	0.7731	0.1174	0.7617	2.2497	2.3604	4.0731	U-Metal	SS-304	1.2772	52,424
#5	0.7285	0.1915	0.7601	2.7202	2.8531	4.9474	U-Metal	SS-304	1.2763	41,140
#6	0.7710	0.1165	0.7610	2.2490	2.3596	4.0712	U-Metal	SS-304	1.2772	73,655
#7	0.6722	0.2377	0.7619	2.7197	4.2674	4.2691	U-Metal	SS-304	1.3016	90,393
#8	0.7271	0.1951	0.7604	2.7441	2.8786	4.9918	U-Metal	SS-304	1.2763	54,970
#9	0.6863	0.1783	0.7256	2.7266	2.8603	4.9331	U-Metal	SS-304	1.2768	68,553
#10	0.7267	0.1927	0.7594	2.7313	2.8649	4.9678	U-Metal	SS-304	1.2763	72,174
Average	0.7218	0.1801	0.7532	2.6390	2.9096	4.7231	–	–	1.2791	59,145.6

Table 6
Results for ten experiments of TopoMut-DE with TMP = 1.0.

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.6083	0.2326	0.7120	2.8520	4.4647	4.4665	U-Metal	SS-304	1.3022	2699
#2	0.5680	0.1495	0.6275	2.7849	2.9202	4.9943	U-Metal	SS-304	1.2778	4300
#3	0.5166	0.0927	0.5579	2.4649	2.5857	4.4070	U-Metal	SS-304	1.2795	8611
#4	0.5309	0.0925	0.5754	2.4422	2.5611	4.3633	U-Metal	SS-304	1.2791	4782
#5	0.6496	0.2536	0.7514	2.8772	4.5142	4.5161	U-Metal	SS-304	1.3014	3614
#6	0.7952	0.0732	0.7589	2.0128	2.1124	3.6440	U-Metal	SS-304	1.2779	5192
#7	0.6154	0.2533	0.7255	2.9773	4.6681	4.6704	U-Metal	SS-304	1.3018	3640
#8	0.5631	0.1052	0.6007	2.4688	2.5895	4.4229	U-Metal	SS-304	1.2788	8682
#9	0.6889	0.2046	0.7618	2.4911	3.9062	3.9083	U-Metal	SS-304	1.3022	5909
#10	0.6337	0.0772	0.6459	2.1856	2.2949	3.9251	U-Metal	SS-304	1.2790	6501
Average	0.6170	0.1534	0.6717	2.5557	3.2617	4.3318			1.2880	5393.0

Table 7
Results for ten experiments of TopoMut-DE with linear TMP (Eq. (17)).

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7254	0.1932	0.7589	2.7370	2.8710	4.9778	U-Metal	SS-304	1.2763	28.985
#2	0.7021	0.1886	0.7406	2.7575	2.8866	4.9972	U-Metal	SS-304	1.2765	92.411
#3	0.7283	0.1935	0.7609	2.7327	2.8666	4.9704	U-Metal	SS-304	1.2763	29.290
#4	0.7029	0.1871	0.7411	2.7521	2.8812	4.9894	U-Metal	SS-304	1.2766	23.572
#5	0.6748	0.1793	0.7178	2.7530	2.8871	4.9769	U-Metal	SS-304	1.2767	23.037
#6	0.7021	0.1885	0.7404	2.7572	2.8863	4.9970	U-Metal	SS-304	1.2765	38.095
#7	0.7026	0.1883	0.7409	2.7551	2.8841	4.9931	U-Metal	SS-304	1.2765	39.519
#8	0.7106	0.1836	0.7436	2.7104	2.8378	4.9132	U-Metal	SS-304	1.2766	28.908
#9	0.6859	0.1769	0.7242	2.7158	2.8484	4.9123	U-Metal	SS-304	1.2767	24.544
#10	0.7106	0.1824	0.7438	2.7029	2.8302	4.8982	U-Metal	SS-304	1.2766	22.749
Average	0.7045	0.1861	0.7412	2.7374	2.8679	4.9625			1.2765	35.111.0

parameters is quite simple: in the early stages of the algorithm the exploration of the search space is more important, but later the regions explored must be exploited, so that solutions with best objective function values are obtained.

The first non-constant TMP is linearly increasing, given by

$$\mathrm{TMP} = \frac{\mathrm{NFE}}{\mathrm{MaxNFE}}, \tag{17}$$

where NFE is the current number of fitness function evaluations and MaxNFE is the maximum number of fitness evaluations (the stopping criterion). By Eq. (16), TMP starts with 0.0(NFE = 0) and ends with 1.0(NFE = MaxNFE).

The second one is exponentially increasing, given by

$$\mathrm{TMP} = 0.1 \times 10^{\mathrm{NFE}/\mathrm{MaxNFE}}. \tag{18}$$

In this case, TMP goes from 0.1(NFE = 0) to 1.0(NFE = MaxNFE), varying smoothly in the beginning, and abruptly in the final stages of the algorithm.

5. Results and discussion

TopoMut-DE was implemented in the C++ programming language. For the stochastic part of this algorithm, we used the state-of-the-art pseudorandom number generating algorithm developed by Matsumoto and Nishimura (1998). Our source code was connected to the HAMMER reactor physics code (Suich and Honeck, 1967), which calculates the objective function value for each solution proposed by the optimization algorithm.

As mentioned in the description of the nuclear reactor core design optimization problem, there are two discrete decision variables, M_f and M_c. As the DE algorithms are designed for continuous optimization, we handled these variables as in Sacco et al. (2009) and Sacco et al. (2013). The possible fuel and cladding materials received integer indexes, but throughout the optimization process were treated as continuous variables which were truncated only for purposes of objective function evaluation. Hence, the two possible fuel materials and three possible cladding materials received, respectively, indexes 0,1 and 0,1,2, being treated as continuous variables in the intervals [0,2) and [0,3).

The parameters for DE are the same as in Sacco et al. (2013): $NP = 100$, $F = 0.5$, and $CR = 0.9$, being widely employed in the literature (see Vesterstrøm and Thomsen, 2004; Ali and Törn, 2004; Rahnamayan et al., 2007; for example). For the topographical heuristic, as Törn and Viitanen (1994), we used $k = 10$.

All the versions tested were setup to stop at 100,000 objective function evaluations, so that the results were obtained with the same maximum computational effort as previous results (Pereira et al., 1999; Domingos et al., 2006; Sacco et al., 2008a, 2009, 2013). Each execution took about 5 h in an Intel® Core™ i7 PC with 12 Gb RAM. The bulk of the effort is taken up by the fitness evaluations by the reactor physics code. We performed ten independent executions for each version, using a set of ten previously selected random seeds in all of them, so that there would be no bias on the quality of the results obtained.

Table 2 shows the results obtained by the canonical DE (from Sacco et al., 2013). Tables 3–8 show the results obtained in ten executions of TopoMut-DE employing constant values of TMP and the linear and exponential schemes. The first columns display the design parameters obtained for each execution, and the two last columns show the objective function values (or fitness, in the evolutionary optimization terminology) and the number of fitness function evaluations (NFE) necessary to reach the optimum.

Comparing Table 2 with Table 3, we can see that TopoMut-DE with TMP = 0.1 obtained a lower average fitness value with a higher number of function evaluations to reach the optimum. The reason is that, in spite of a low TMP, the new mutation scheme prevented DE from premature convergence to a suboptimum.

A TMP of 0.25 (Table 4) yielded the best overall results: the best fitness value was obtained in 8/10 experiments. However, the

Table 8
Results for ten experiments of TopoMut-DE with exponential TMP (Eq. (18)).

Experiment	R_f (cm)	Δ_c (cm)	Δ_m (cm)	E_1 (%)	E_2 (%)	E_3 (%)	M_f	M_c	Fitness	NFE
#1	0.7235	0.1929	0.7570	2.7374	2.8711	4.9778	U-Metal	SS-304	1.2763	88,346
#2	0.7083	0.1875	0.7454	2.7368	2.8709	4.9707	U-Metal	SS-304	1.2765	91,942
#3	0.7049	0.1828	0.7401	2.7172	2.8447	4.9234	U-Metal	SS-304	1.2766	70,768
#4	0.7302	0.1680	0.7530	2.5856	2.7076	4.6857	U-Metal	SS-304	1.2767	78,560
#5	0.7271	0.1939	0.7606	2.7377	2.8721	4.9793	U-Metal	SS-304	1.2763	82,784
#6	0.7048	0.1871	0.7425	2.7441	2.8726	4.9733	U-Metal	SS-304	1.2765	97,256
#7	0.7032	0.1852	0.7395	2.7335	2.8620	4.9527	U-Metal	SS-304	1.2766	80,224
#8	0.7279	0.1954	0.7619	2.7458	2.8802	4.9950	U-Metal	SS-304	1.2763	85,767
#9	0.6988	0.1874	0.7375	2.7584	2.8876	4.9988	U-Metal	SS-304	1.2766	69,077
#10	0.7113	0.1823	0.7453	2.7048	2.8317	4.9027	U-Metal	SS-304	1.2766	60,459
Average	0.7140	0.1862	0.7483	2.7201	2.8500	4.9359			1.2765	80,518.3

Table 9
Comparison with previously published best results obtained by other populational algorithms.

		GA[a]	PSO[b]	DE[c]	DE[d]	OB-DE[d]	TMO-DE[d]	TopoMut-DE
Objectives and constraints	Fitness	1.3100	1.2767	1.2763	1.2763	1.2763	1.2763	1.2763
	Minimum average peak factor	1.3100	1.2767	1.2763	1.2763	1.2763	1.2763	1.2763
	Average flux	8.02×10^{-5}	8.07×10^{-5}	8.08×10^{-5}	8.06×10^{-5}	8.08×10^{-5}	8.08×10^{-5}	8.08×10^{-5}
	k_{eff}	1.000	0.990	1.000	0.990	0.990	0.990	1.000
Parameters	R_f (cm)	0.5621	0.7459	0.7281	0.7278	0.7240	0.7271	0.7275
	Δ_r (cm)	0.1770	0.1647	0.1932	0.1953	0.1939	0.1925	0.1953
	Δ_m (cm)	0.6581	0.7620	0.7610	0.7606	0.7586	0.7603	0.7610
	E_1 (%)	2.756	2.5364	2.7318	2.7457	2.7445	2.7300	2.7450
	E_2 (%)	4.032	2.6608	2.8657	2.8801	2.8786	2.8638	2.8794
	E_3 (%)	4.457	4.6067	4.9686	4.9964	4.9914	4.9651	4.9937
	M_f	U-metal	U-metal	U-metal	U-metal	U-metal	U-metal	U-metal
	M_c	Stainless-304	Stainless-304	Stainless-304	Stainless-304	Stainless-304	Stainless-304	Stainless-304
NFE		96,000	N.A.	95,730	49,570	29,226	53,144	25,831

[a] Pereira et al., 1999.
[b] Domingos et al., 2006.
[c] Sacco et al., 2009.
[d] Sacco et al., 2013.

number of function evaluations increased, but this is the cost to pay for a more effective exploitation of the most promising areas of the search space.

Tables 5 and 6 demonstrate that using a higher TMP prevents a wider exploration of the search space, relying only on the exploitation factor. From Table 6, one can see that TMP = 1.0 caused an extremely premature convergence to low-quality solutions.

Table 7 displays the results achieved using the linearly increasing TMP given by Eq. (17). The fitness values were similar to those obtained with TMP = 0.1 (2/10 best fitness values and the same average fitness), but the computational effort was the lowest. This means that we managed a fine balance between exploration and exploitation using this scheme.

Taking a glance at Table 8, we can conclude that the exponentially increasing TMP given by Eq. (18) obtained 3/10 best fitness value with more than two times the computational cost required by the linear scheme.

Finally, Table 9 compares the best configuration and computational cost achieved here against results available in the literature applying other techniques: the genetic algorithm (Pereira et al., 1999), particle swarm optimization (Domingos et al., 2006), DE with another setup (Sacco et al., 2009), and DE, Opposition-Based DE (Rahnamayan et al., 2008), and Trigonometric Mutation (Fan and Lampinen, 2003), taken from Sacco et al. (2013). The results obtained by the DE paradigm demonstrate once more for this problem (as previously in Sacco et al., 2009; Sacco et al., 2013), its superiority over the more established genetic algorithm and particle swarm optimization methods.

6. Conclusions

Because of its multimodality, the nuclear reactor core design optimization problem requires the use of techniques that promote a good balance between exploration of the search space and exploitation of its most promising areas. In this work, we tackled this problem introducing a new mutation scheme that promotes this exploitation, along with the exploration provided by the traditional scheme. We compared the DE with this new operator against the canonical DE so that its effects were more clearly perceived than comparing with more sophisticated DE variants, like opposition-based DE. In fact, topographical mutation can be used in conjunction with some of these variants.

Based on the results obtained here, we recommend the use of TopoMut-DE with TMP = 0.25 or with the linearly increasing value.

As further development, we plan to introduce an opposition-based version of TopoMut-DE and, also, to solve the nuclear fuel management optimization problem (Kopaczek and Turinsky, 1991; DeChaine and Feltus, 1995; Jiang et al., 2006; Poursalehi et al., 2013) using the method introduced in this work. As the search space of this problem is highly multimodal (Galperin, 1995), this is a promising approach.

Acknowledgments

W.F.S. and N.H. gratefully acknowledge the financial support provided by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico, Ministry of Science, Technology and

Innovation, Brazil). The authors also would like to thank Prof. Montaz Ali for providing a copy of his doctoral thesis and the two anonymous reviewers, for their constructive comments, which helped us to improve the manuscript. The research by N.H. has been carried out within the framework of project PROCIENCIA-UERJ financed by FAPERJ.

References

Ali, M.M., 1994. Some Modified Stochastic Global Optimization Algorithms with Applications. Doctoral Thesis. Department of Mathematical Sciences, Loughborough University of Technology, Loughborough, UK.
Ali, M.M., Storey, C., 1994. Topographical multilevel single linkage. J. Glob. Optim. 5, 267–276.
Ali, M.M., Törn, A., 2000. Optimization of carbon and silicon cluster geometry for Tersoff potential using differential evolution. In: Floudas, A., Pardalos, M. (Eds.), Optimization in Computational Chemistry and Molecular Biology: Local and Global Approaches. Kluwer Academic Publisher, Dordrecht, The Netherlands, pp. 287–300.
Ali, M.M., Törn, A., 2004. Population set based global optimization algorithms: some modifications and numerical studies. Comp. Oper. Res. 31, 1703–1725.
Becker, R.W., Lago, G.V., 1970. A global optimization algorithm. In: Proceedings of the 8th Allerton Conference on Circuits and System Theory, Monticello, IL, USA, pp. 3–12.
Das, S., Suganthan, P.N., 2010. Differential evolution: a survey of the state-of-the-art. IEEE T. Evolut. Comput. 15, 4–31.
DeChaine, M.D., Feltus, M.A., 1995. Nuclear fuel management optimization using genetic algorithms. Nucl. Technol. 111, 109–114.
Domingos, R.P., Schirru, R., Pereira, C.M.N.A., 2006. Particle swarm optimization in reactor core design. Nucl. Eng. Des. 152, 197–203.
Duderstadt, J.J., Hamilton, L.J., 1976. Nuclear Reactor Analysis. John Wiley and Sons, New York.
Fan, H.Y., Lampinen, J., 2003. A trigonometric mutation operation to differential evolution. J. Glob. Optim. 27, 105–129.
Galperin, A., 1995. Exploration of the search space of the in-core fuel management problem by knowledge-based techniques. Nucl. Sci. Eng. 19, 144–152.
Gentle, J.E., 2003. Random Number Generation and Monte Carlo Methods, second ed. Springer, New York.
Holland, J.H., 1992. Adaptation in Natural and Artificial Systems. MIT Press, Cambridge, MA, USA.
Jiang, S., Ziver, A.K., Carter, J.N., Pain, C.C., Goddard, A.H.J., Franklin, S., Phillips, H.J., 2006. Estimation of distribution algorithms for nuclear reactor fuel management optimisation. Ann. Nucl. Energy 33, 1039–1057.
Kaelo, P., Ali, M.M., 2006. A numerical study of some modified differential evolution algorithms. Eur. J. Oper. Res. 169, 1176–1184.
Kennedy, J., Eberhart, R.C., 1995. Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, N.J., 1942–1948.
Kropaczek, D.J., Turinsky, P.J., 1991. In-core nuclear fuel management optimization for pressurized water reactors utilizing simulated annealing. Nucl. Technol. 95, 9–32.
Levi, A.F.J., Haas, S., 2010. Appendix A – global optimization algorithms. In: Levi, A.F.J., Haas, S. (Eds.), Optimal Device Design. Cambridge University Press, Cambridge, UK, pp. 262–276.
Mahfoud, S.W., 1995. Niching Methods for Genetic Algorithms. PhD Thesis. Illinois Genetic Algorithm Laboratory, University of Illinois at Urbana- Champaign, Urbana, IL.
Matsumoto, M., Nishimura, T., 1998. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8, 3–20.
Pereira, C.M.N.A., Schirru, R., Martinez, A.S., 1999. Basic investigations related to genetic algorithms in core designs. Ann. Nucl. Energy 26, 173–193.
Pereira, C.M.N.A., Lapa, C.M.F., 2003. Coarse-grained parallel genetic algorithm applied to a nuclear reactor core design optimization problem. Ann. Nucl. Energy 30, 555–565.
Poursalehi, N., Zolfaghari, A., Minuchehr, A., Moghaddam, H.K., 2013. Continuous firefly algorithm applied to PWR core pattern enhancement. Nucl. Eng. Des. 258, 107–115.
Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 2007. Numerical Recipes – the Art of Scientific Computing, third ed. Cambridge University Press, Cambridge, UK.
Price, K.V., Storn, R.M., Lampinen, J.A., 2005. Differential Evolution – a Practical Approach to Global Optimization. Springer-Verlag, Berlin.
Qu, B.Y., Suganthan, P.N., Liang, J.J., 2012. Differential evolution with neighborhood mutation for multimodal optimization. IEEE T. Evolut. Comput. 16, 601–614.
Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A., 2007. Opposition-based differential evolution (ODE) with variable jumping rate. In: Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence (FOCI 2007). IEEE, Piscataway, NJ, pp. 81–88.
Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A., 2008. Opposition-based differential evolution. IEEE T. Evolut. Comput. 12, 64–79.
Sacco, W.F., Machado, M.D., Pereira, C.M.N.A., Schirru, R., 2004. The fuzzy clearing approach for a niching genetic algorithm applied to a nuclear reactor core design optimization problem. Ann. Nucl. Energy 31, 55–69.
Sacco, W.F., Alves Filho, H., Henderson, N., de Oliveira, C.R.E., 2008. A metropolis algorithm combined with Nelder-Mead simplex applied to nuclear reactor core design. Ann. Nucl. Energy 35, 861–867.
Sacco, W.F., Henderson, N., Rios-Coelho, A.C., Ali, M.M., Pereira, C.M.N.A., 2009. Differential evolution algorithms applied to nuclear reactor core design. Ann. Nucl. Energy 36, 1093–1099.
Sacco, W.F., Meneses, A.A.M., Henderson, N., 2013. Some studies on differential evolution variants for application to nuclear reactor core design. Prog. Nucl. Energy 63, 49–56.
Storn, R., 1996. On the usage of differential evolution for function optimization. In: Smith, M.H., Lee, M.A., Keller, J., Yen, J. (Eds.), Proceedings of the 1996 Biennial Conference of the North American Fuzzy Information Processing Society. IEEE Press, New York, pp. 519–523.
Storn, R., Price, K., 1997. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11, 341–359.
Suich, J.E., Honeck, H.C., 1967. The HAMMER System Heterogeneous Analysis by Multigroup Methods of Exponentials and Reactor. Savannah River Laboratory, Aiken, South Carolina.
Timmer, G.T., 1984. Global Optimization – a Stochastic Approach. Ph.D. Thesis. Erasmus University, Rotterdam, The Netherlands.
Törn, A., 1973. Global optimization as a combination of global and local search. In: Proceedings of Computer Simulation versus Analytical Solutions for Business and Economic Models, Gothenburg, Sweden, pp. 191–206.
Törn, A., 1978. A search clustering approach to global optimization. In: Dixon, L.C.W., Szegö, G.P. (Eds.), Towards Global Optimization, vol. 2. North-Holland Publishing Company, pp. 49–62.
Törn, A., Žilinskas, A., 1989. Global optimization. In: Goos, G., Hartmanis, J. (Eds.), Lecture Notes in Computer Science, vol. 350. Springer-Verlag, Berlin.
Törn, A., Viitanen, S., 1992. Topographical global optimization. In: Floudas, C.A., Pardalos, P.M. (Eds.), Recent Advances in Global Optimization. Princeton University Press, Princeton, NJ, pp. 384–398.
Törn, A., Viitanen, S., 1994. Topographical global optimization using pre-sampled points. J. Glob. Optim. 5, 267–276.
Törn, A., Viitanen, S., 1996. Iterative topographical global optimization. In: Floudas, C.A., Pardalos, P.M. (Eds.), State of the Art in Global Optimization. Kluwer Academic Publishers, the Netherlands, pp. 353–363.
Vesterstrøm, J., Thomsen, R., 2004. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: Proceedings of the 2004 IEEE Congress on Evolutionary Computation (CEC 2004), vol. 2. IEEE, Piscataway, NJ, pp. 1980–1987.