# MODIFIED DIFFERENTIAL EVOLUTION ALGORITHM IN LOAD DISPATCH OPTIMIZATION PROBLEM

*O. Šubrt, M. Virius*

Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Břehová 7, 115 19 Prague 1, Czech Republic

*K. Macek*

Academy of Sciences of the Czech Republic, Institute of Information Science and Automation, Department of Adaptive Systems, Pod Vodárenskou věží 4, 182 08 Prague 8, Czech Republic

**Abstract**

**The Load Dispatch (LD) is one of the essential problems in the power grids. The LD problem consists in scheduling of the output power of particular generators in time to cover the overall power demand while minimizing the overall generation costs, eventually addressing also other (e.g. ecological) objectives. The LD problem is generally non-convex, especially when involving the valve-point effect. This non-convexity challenges analytical and heuristic methods in finding optimal solution in reasonable time. Differential Evolution (DE) is one of evolutionary algorithms, which has been used in many optimization problems due to its simplicity and efficiency. This paper presents LD solution considering valve loading effect via the Modified Differential Evolution (MDE) algorithm. The MDE is inspired from DE and has new mapping function thus represents a new efficient stochastic search technique. The proposed MDE is examined on the LD test system and compared with other LD solution methods.**

Table 1: NOMENCLATURE

| Symbol | Interpretation |
|---|---|
| $N$ | number of generators in the system |
| $M$ | number of hours in the time horizon |
| $P_{i,t}$ | the output power generation of generator $i$ in time $t$ |
| $a_i, b_i, c_i$ | fuel function parameters of generator $i$ |
| $e_i, f_i$ | valve-point effect parameters of generator $i$ |
| $P_{Dt}$ | the total power system demand in time $t$ |
| $P_{Lt}$ | the total system transmission losses in time $t$ |
| $P_i^{min}$ | the minimum limit on output power generation of generator $i$ |
| $P_i^{max}$ | the maximum limit on output power generation of generator $i$ |
| $UR_i$ | the ramp-up rate limits of the $i$th generator |
| $DR_i$ | the ramp-down rate limits of the $i$th generator |
| $\mathbf{B}$ | the $N \times N$ matrix of loss coefficients |
| $\mathbf{x}, \mathbf{p}$ | vectors representing individuals |
| rand() | the generatation of a random number in the range [0,1] |
| rand($j$) | the $j$th valuation of function rand() function |
| round($\cdot$) | function that rounds its argument to the nearest integer value |
| $s$ | the counter of iteration of MDE |
| $s_{max}$ | the maximum iteration of MDE |
| $f(\cdot)$ | fitness function of MDE |
| $N_p$ | number of individuals in each population of MDE |
| $c_1, c_2, k_1, k_2$ | constants for MDE |

# 1 Introduction

The power industry faces the challenge of adapting to circumstances where the optimal use of resources is crucial because of rapidly decrease of reserves of fossil fuels and high volatility of renewable resources [9]. Load Dispatch (LD) is defined as the process of allocating generation levels to the generating units in the mix, so that the system load is supplied entirely and most economically [4]. Several classical optimization techniques, such as linear programming [11], quadratic programming [6], non-linear programming [10], dynamic programming [3] were proposed to solve LD problem.

This paper is organized as follows. Section 2 provides LD problem formulation considering valve-point effect. In Section 3, the MDE algorithm is described and its subroutines are discussed in detail. Obtained results from the MDE to solve the non-convex LD problem are presented in Section 4. Besides, the MDE is compared with some of the most recently published LD solution methods. Section 5 concludes the paper.

# 2 Problem Formulation

The formulation of the LD problem [13] scheduled over a period of time consists in the minimization of total fuel costs, subject to the real power balanced with the total load demand, as well as the limits on generators outputs, i.e.:

$$F = \sum_{t=1}^{M} \sum_{i=1}^{N} F_{i,t}(P_{i,t}) \tag{1}$$

with the fuel cost functions of the generation units with valve-point loading represented as:

$$F_{i,t}(P_{i,t}) = a_i + b_i P_{i,t} + c_i P_{i,t}^2 + |e_i \sin(f_i(P_i^{min} - P_{i,t}))| \tag{2}$$

where $a_i, b_i, c_i, e_i, f_i$ are given parameters.

It is subjected to

- real power balance

$$\sum_{i=1}^{N} P_{i,t} = P_{Dt} + P_{Lt} \tag{3}$$

- real power operation limits

$$P_i^{min} \leq P_i \leq P_i^{max} \quad i = 1, \ldots, N \tag{4}$$

- generating unit ramp rate limits[1]

$$\begin{aligned} P_{i,t} - P_{i,(t-1)} &\leq UR_i \quad i = 1, \ldots, N, t = 1, \ldots, M \\ P_{i,(t-1)} - P_{i,t} &\leq DR_i \quad i = 1, \ldots, N, t = 1, \ldots, M \end{aligned} \tag{5}$$

where the total system transmission losses $P_{Lt}$ in time $t$ are approximated as:

$$P_{Lt} = \sum_{i=1}^{N} \sum_{j=1}^{N} P_{i,t} B_{i,j} P_{j,t} \tag{6}$$

where $B_{i,j}$ represents the element at $i$th row and $j$th column of the matrix of loss coefficients $\mathbf{B}$.

[1] Particularly for $t = 1, \forall i \in \hat{N}$: $P_{i,(t-1)} \implies P_{i,0} = P_{i,M}$. It guarantees the periodic application of scheduling.

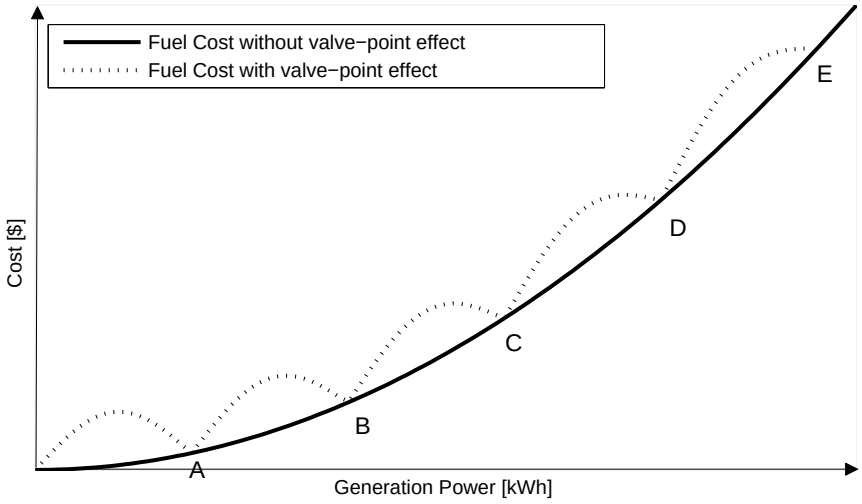


Figure 1: Valve-point effect

Probably the most complete definition for the valve-point effect phenomenon [1] can be: "the ripples in the input-output curve in the thin line express the result of the sharp increase in losses due to wire drawing effects which occur as each steam admission valve starts to open". The sinusoidal term added to the fuel cost function introduces ripples to heat-rate curve. In Figure 1, the addition of the valve-point effect increases the non-linearity of the search space as well as the number of local minima. Furthermore, the application of the absoute value in Equation (2) makes the objective function generally non-differentiable.

# 3 Modified Differential Evolution

The Modified Differential Evolution (MDE) [2] is a heuristic algorithm based on Genetic Algorithm (GA) [7] and has new mutation operation and selection mechanism inspired from Particle Swarm Optimization (PSO) [5] and Simulated Annealing (SA) [8]. We extend the basic version by introducing Mapping Function (MF) for constraints handling which satisfies all constraints within whole evolution in contrast to the original version. In this paper, an individual is represented by vector with $n$ elements $\mathbf{x} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ or $\mathbf{p} = (p_1, p_2, \ldots, p_n)^{\mathrm{T}}$, where the $\mathbf{x}$ notation is used for general optimization operations, while the $\mathbf{p}$ notation is used for domain specific operations. However, both are used for elements of the same population.

In this section, at first, we present the whole MDE algorithm. Consequently its parts are described in more detail. Namely, Mutation operator, Crossover operation, and Selection mechanism are presented, so the original version of MDE is recapitulated. Finally the novel mapping function is provided.

## 3.1 The MDE Algorithm

At first, the parameters are set. Then the initialization of population is executed. To create new population, mutation operator, crossover operator and selection mechanism, respectively, are applied. After applying, a new mapping function must be used because of constraints handling. If the stopping criterion is not satisfied go to produce new population by evolutionary operators, else return the individual with the best fitness as the solution. Here, the maximum number of iterations is selected as the stopping criterion. The MDE algorithm [2] is described in Algorithm 1:

**Algorithm 1** MDE algorithm

set size of population ($N_p$), the maximum iteration ($s_{max}$), constants ($c_1, c_2, k_1, k_2, k_0, \alpha$)
set initial value of the temperature ($T$);
initializationOfPopulation();
**for** $s = 1 \rightarrow s_{max}$ **do** ▷ loop for MDE iterations
  set the mutation factor $\beta = c_1 - c_2 * s/s_{max}$;
  set the crossover rate $CR = k_1 - k_2 * s/s_{max}$;
  set $k = k_0 - \text{round}((k_0 - 1) * s/s_{max})$;
  $T = \alpha * T$;
  **for** $i = 1 \rightarrow N_p$ **do** ▷ loop for updating the whole population by evolutionary operators
    $xParent = \text{getIndividual}(i)$;
    $xRandom = \text{getRandomIndividual}()$;
    $kBestIndividuals = \text{getKBestIndividuals}(k)$;
    $uTrial = \text{mutation}(kBestIndividuals, xRandom, k, \beta)$;
    $xOffspring = \text{crossover}(uTrial, xParent, CR)$;
    $xResult = \text{selection}(xParent, xOffspring, T)$;
    $\text{setIndividual}(i, xResult)$;
  **end for**
  **for** $i = 1 \rightarrow N_p$ **do** ▷ loop for ensuring the constraints
    $x = \text{getIndividual}(i)$;
    $x = \text{mappingFunction}(x)$;
    $\text{setIndividual}(i, x)$;
  **end for**
**end for**
$BestIndividual = \text{getBestIndividual}()$;
$Costs = \text{getCosts}(BestIndividual)$;

### 3.2 Mutation operator

The mutation operator produces a trial vector for each individual of the current population. This trial vector will then be used by the crossover operator to produce offspring.

In the original DE [12], there is no criterion for selecting the individuals to apply the mutation operator. The individuals are changed in this mutation to increase the diversity of population without any assurance that these changes result in better offspring. To overcome this deficiency, the mutation operation [2] uses the information of $k$ best individuals. In this operation, the best individuals of current population are selected based on their fitness values and used for mutation as follows:

$$\mathbf{u}_i(s) = \mathbf{x}_{i,1}(s) + \beta\left(\sum_{j=1}^{k}\left(\frac{\frac{1}{f(\mathbf{x}_{j,b}(s))}}{\sum_{m=1}^{k}\frac{1}{f(\mathbf{x}_{m,b}(s))}}\right)\mathbf{x}_{j,b}(s) - \mathbf{x}_{i,3G}(s)\right) \tag{7}$$

where $\mathbf{x}_{j,b}(s)$, $j = 1, \ldots, k$ and $\mathbf{x}_{m,b}(s)$, $m = 1, \ldots, k$ are $k$ best individuals owning the highest fitness values in the current population. Since $f(\cdot)$ is the objective function of a minimization problem, one can mention that $\frac{1}{f(\mathbf{x}_{j,b}(s))}$ is a measure of the fitness of $\mathbf{x}_{j,b}(s)$. Hence it can be applied on calculation of a normalized weight for $\mathbf{x}_{j,b}(s)$ (summation of these normalized weights is equal to one).

Vector $\mathbf{x}_{i,1}(s)$ is randomly selected individual from current population for the parent $\mathbf{x}_i(s)$. $\beta$ is the scaling factor, controlling the amplification of the differential variation. Theoretically $\beta \in (0, \infty)$, but it is usually taken from the range [0.1, 1].

Also, to enhance the diversity of the search process, $\mathbf{x}_{i,3G}(s)$, inspired from GA, is the result of variable weight arithmetic crossover between randomly selected parents $\mathbf{x}_{i,31}(s)$ and $\mathbf{x}_{i,32}(s)$

from current population:

$$\mathbf{x}_{i,3G}(s) = (1-\omega_a)\mathbf{x}_{i,31}(s) + \omega_a \mathbf{x}_{i,32}(s) \tag{8}$$

where the variable weight $\omega_a$ is randomly chosen in the range of [0,1].

The motivation for using the proposed mutation is as follows. The information content of the best individuals is used to guide the search process of the MDE in contrast to DE which only relies on the randomly selected individuals. The initial idea of the mutation has been taken from PSO where the direction vectors from each particle to the best one are used to modify the position of each particular particle. However, in the mutation a combination of the $k$ best individuals is used instead of the best one (each of the $k$ individuals contribute to the combination based on its fitness value). In this way, it is avoided that individuals move toward one individual that reduces the diversity of population and causes the convergence of the algorithm to a local minimum. The parameter $k$ depends on the dimension of individuals and size of population. Large values of $k$ may cause that poor individuals are involved in the mutation operation and decelerates the convergence speed. On the other hand, small values of $k$ could lead to premature termination and trapping in a local minimum like the problem of $k = 1$. To enhance exploitation of the mutation operator, the parameter $k$ is adaptively changed along the evolution process. In other words, the MDE begins with a large value of $k$ and then it is linearly decreased along the iterations:

$$k(s) = k_0 - \text{round}\left((k_0 - 1)\frac{s}{s_{max}}\right) \tag{9}$$

where $k_0$ is the initial value of $k$.

### 3.3 Crossover operation

The crossover operator in the MDE [2], it has identical figure as in DE. The MDE crossover operator implements a discrete recombination of the trial vector $\mathbf{u}_i(s)$ and the parent vector $\mathbf{x}_i(s)$ to produce offspring $\mathbf{x}'_i(s)$. The crossover is implemented as follows:

$$x'_{i,j}(s) = \begin{cases} u_{i,j}(s) & \text{if} \quad \text{rand}(j) \leq CR \\ x_{i,j}(s) & \text{otherwise} \end{cases} \tag{10}$$

where $x_{i,j}(s)$ refers to the $j$th element of the vector $\mathbf{x}_i(s)$. Elements $u_{i,j}(s)$ and $x'_{i,j}(s)$ are similarly defined. CR is the crossover rate in the range [0,1].

### 3.4 Selection mechanism

In the MDE, a probabilistic selection mechanism [2] is used instead of the deterministic selection of the original DE. The selection mechanism has been inspired from Simulated Annealing (SA). SA uses a random search strategy, which not only accepts new solutions that decrease the objective function value (assuming a minimization problem), but may also accept new solutions that rather increase the objective function value based on a predetermined probability distribution function. Exponential probability distribution function is usually used for this purpose. Based on this idea, the selection mechanism of the MDE can be described as follows:

$$\mathbf{x}_i(s+1) = \begin{cases} \mathbf{x}'_i(s) & \text{if} \quad f(\mathbf{x}'_i(s)) \leq f(\mathbf{x}_i(s)) \\ \mathbf{x}'_i(s) & \text{if} \quad f(\mathbf{x}'_i(s)) > f(\mathbf{x}_i(s)) \wedge h(\mathbf{x}_i(s), \mathbf{x}'_i(s)) > \text{rand}() \\ \mathbf{x}_i(s) & \text{otherwise} \end{cases} \tag{11}$$

$$h(\mathbf{x}_i(s), \mathbf{x}'_i(s)) = \exp\left(\frac{f(\mathbf{x}_i(s)) - f(\mathbf{x}'_i(s))}{f(\mathbf{x}_i(s))T}\right) \tag{12}$$

where $T$ is temperature like that defined in SA technique. Here, the temperature $T$ is adaptively changed in the evolution process as follows:

$$\begin{aligned} T(s+1) &= \alpha T(s) \\ T(0) &= T_0 \end{aligned} \tag{13}$$

The parameter $\alpha$ is the rate of reducing the temperature ($\alpha < 1$). $T_0$ is the initial temperature. Normalized difference between the parent and offspring objective functions has been considered in (11) to eliminate the effect of different ranges of objective functions (the adjustment of the temperature $T$ becomes independent from the range of objective function). The selection mechanism begins with a large value for the initial temperature. In other words, at the beginning of the evolution process, many new worse solutions $\mathbf{x}_j(s)$ have chance to be selected to increase the exploration of the MDE. However, by evolving the individuals, the temperature $T$ decreases along the iterations and so the probability of selecting the worse solutions is decreased.

### 3.5 The scaling factor and recombination rate

The scaling factor $\beta$ and recombination rate $CR$ affect the exploration and exploitation of algorithm [2]. Exploration is the algorithm ability to cover and explore different areas in the feasible search space while exploitation is the ability to concentrate only on promising areas in the search space and to enhance the quality of the potential solution in the promising region. The scaling factor $\beta$ controls the amplification of the differential variations. The smaller the value of $\beta$, the smaller the mutation step sizes, and the longer it will be for the algorithm to converge. Larger values for $\beta$ facilitate exploration, but may cause the algorithm to overshoot good optima. The value of $\beta$ should be small enough to allow differentials to explore tight valleys, and large enough to maintain diversity. As the population size increases, the scaling factor should be decreased. In this paper, an adaptive scaling factor is adopted to have a good compromise between exploration and exploitation. For increasing exploration, initial $\beta$ is chosen large. Then, it is reduced linearly along the iterations for good exploitation:

$$\beta(s) = c_1 - c_2 \frac{s}{s_{max}} \tag{14}$$

In this way the mutation operator performs a wider search in the solution space at the early stages of the evolution, and at the later stages the search is restricted around the local area of mature individuals, resembling a hill-climbing operator.

The probability of recombination, $CR$, has a direct influence on the diversity of MDE. The higher the probability of recombination, the more variation is introduced in the new population, thereby increasing diversity and exploration. Increasing $CR$ often results in faster convergence, while decreasing $CR$ increases search robustness. In this paper, an adaptive $CR$ is similarly adopted. $CR$ is changed along the evolution process like $\beta$ as follows:

$$CR(s) = k_1 - k_2 \frac{s}{s_{max}} \tag{15}$$

### 3.6 Mapping function

#### 3.6.1 Mapping function in original MDE

A key factor in the application of optimization methods is how the algorithm handles the constraints related to the problem. The MF maps unfeasible solution to a feasible one that satisfies the equality constraint. Suppose that the candidate solution $\mathbf{p} = (P_1, P_2, \ldots, P_n)^{\mathrm{T}}$ violates the equality constraint (3). The proposed MF maps each component $P_i$ of $\mathbf{p}$ as follows:

$$P_i^{mapped} = P_i \frac{P_D + P_L}{\sum_{j=1}^{n} P_j}, \quad 1 \le i \le n \tag{16}$$

where $\mathbf{p}^{mapped} = (P_1^{mapped}, P_2^{mapped}, \ldots, P_n^{mapped})^{\mathrm{T}}$ indicates the mapped feasible solution corresponding to the candidate solution $\mathbf{p}$. After applying this MF, all candidate solutions will satisfy the equality constraint (3).

On the other hand after applying this MF, some mapped generations $P_i^{mapped}, i \in \hat{n}$ may deviate from the inequality limits (4) and (5). These mapped generations should be cut to their associated limits. By combining (4) and (5), ramp rate constrained operation limits of units can be represented as follows:

$$P_i^{min,r} \leq P_i \leq P_i^{max,r} \quad i = 1, \ldots, n \tag{17}$$

where

$$P_i^{min,r} = \max(P_i^{min}, P_{i,(t-1)} - DR_i) \tag{18}$$

$$P_i^{max,r} = \min(P_i^{max}, P_{i,(t-1)} + UR_i) \tag{19}$$

To satisfy the constraints of (17), whenever generation of each generator exceeds from its ramp rate constrained operation limits, the amount of generation is cut to the associated limit. In other words, we have:

$$P_i = \begin{cases} P_i^{min,r} & \text{if} \quad P_i < P_i^{min,r} \\ P_i & \text{if} \quad P_i^{min,r} \leq P_i \leq P_i^{max,r} \\ P_i^{max,r} & \text{if} \quad P_i > P_i^{max,r} \end{cases} \tag{20}$$

#### 3.6.2 Proposed changes in the mapping function

In this paper, however, we propose a novel approach how to deal with these constraints. By applying origin MF, some slight violation from the equality constraint (3) may still be appeared. This insufficiency is removed by iterative applying of the equation (16) and equation (20) until the satisfactory accuracy of the satisfaction of the equality constraint (3) is reached. This modification is introduced by us and does not enlarge the execution time.

In addition to constraints handling, it is necessary to apply MF on each individual in the population within whole evolution. At the beginning of evolution, the application of the MF merely on initialization of the population is not enough. Namely during the evolution process, strong violation of all constraints occurs without application of the MF. We have removed this drawback in this paragraph unlike [2].

## 4 Numerical Results

The MDE algorithm has been implemented in Matlab 7 (R2010a, 64-bit) computing environment on a personal cumputer Intel(R) Core(TM)2 i5-2430M CPU 2,40 GHz and 6 GB RAM memory.

### 4.1 Test Case Formulation

In this section, the Test Case of Load Dispatch Problem is introduced and investigated. It solves the LD problem for twenty four hours with different load demand in each hour. The changing load demand throughout a twenty four hour period reflects realistic situations that the control engineers in power plants usually encounter. The Test Case is taken from [1].

In Figure 2, the network is consists of five power generation units and the data, transmission loss formula coefficients and load demand for twenty four hours are listed in tables 2 and 3, respectively, whereas the transmission loss formula coefficients are in matrix $\mathbf{B}$.

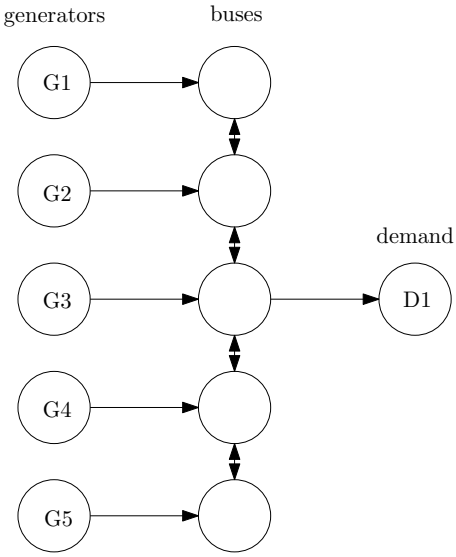


Figure 2: Network is represented as Directed Graph

$$\mathbf{B} = 10^{-6} \begin{pmatrix} 49 & 14 & 15 & 15 & 20 \\ 14 & 45 & 16 & 20 & 18 \\ 15 & 16 & 39 & 10 & 12 \\ 15 & 20 & 10 & 40 & 14 \\ 20 & 18 & 12 & 14 & 35 \end{pmatrix}$$

| Coefficient | Generator 1 | Generator 2 | Generator 3 | Generator 4 | Generator 5 |
|---|---|---|---|---|---|
| $a_i$ [\$/h] | 25 | 60 | 100 | 120 | 40 |
| $b_i$ [\$/MWh] | 2.0 | 1.8 | 2.1 | 2.0 | 1.8 |
| $c_i$ [\$/(MW)$^2$h] | 0.0080 | 0.0030 | 0.0012 | 0.0010 | 0.0015 |
| $e_i$ [\$/h] | 100 | 140 | 160 | 180 | 200 |
| $f_i$ [1/MW] | 0.042 | 0.040 | 0.038 | 0.037 | 0.035 |
| $P_i^{min}$ [MW] | 10 | 20 | 30 | 40 | 50 |
| $P_i^{max}$ [MW] | 75 | 125 | 175 | 250 | 300 |
| $UR$ [MW]$^2$ | 30 | 30 | 40 | 50 | 50 |
| $DR$ [MW]$^3$ | 30 | 30 | 40 | 50 | 50 |

Table 2: Coefficients for Network with 5 generators

| Time [h] | Load [MW] | Time [h] | Load [MW] | Time [h] | Load [MW] | Time [h] | Load [MW] |
|---|---|---|---|---|---|---|---|
| 1 | 410 | 7 | 626 | 13 | 704 | 19 | 654 |
| 2 | 435 | 8 | 654 | 14 | 690 | 20 | 704 |
| 3 | 475 | 9 | 690 | 15 | 654 | 21 | 680 |
| 4 | 530 | 10 | 704 | 16 | 580 | 22 | 605 |
| 5 | 558 | 11 | 720 | 17 | 558 | 23 | 527 |
| 6 | 608 | 12 | 740 | 18 | 608 | 24 | 463 |

Table 3: Load Demand for 24 hours

[2]The original unit is [MW/h] in source [1].
[3]The original unit is [MW/h] in source [1].

The parameters of the MDE are chosen on the basis of [2]. Though, the reference shows the low sensitivity of the MDE with respect to its adjustable parameters, which indicates another aspect of the robustness of the MDE. To execute MDE, no wide experience with heuristics of an executor is required. The parameters of the MDE to solve the Load Dispatch problem are:

| Parameter | $N_p$ | $s_{max}$ | $c_1$ | $c_2$ | $k_1$ | $k_2$ | $k_0$ | $T_0$ | $\alpha$ |
|---|---|---|---|---|---|---|---|---|---|
| Value | 50 | 500 | 0.6 | 0.4 | 0.3 | 0.1 | 5 | 1 | 0.7 |

Table 4: Table with parameters for MDE

## 4.2 Test Case Results

The MDE runs ten times and the best solution is the solution with the lowest fitness at the end of Evolution Process. Figure 3 represents the Evolution Process proposed MDE of the best solution. At the beginning of the Evolution Process, the figure shows the fast convergence behavior of the proposed MDE. Nevertheless, there are several "teeth" in evolution of costs, which are caused by selection mechanism, based on the simulate annealing. It sometimes selects individuals with worse fitness. They have chance to show their potential to produce next population. Due to modification of selection mechanism, this feature weakens at the end of the Evolution Process.

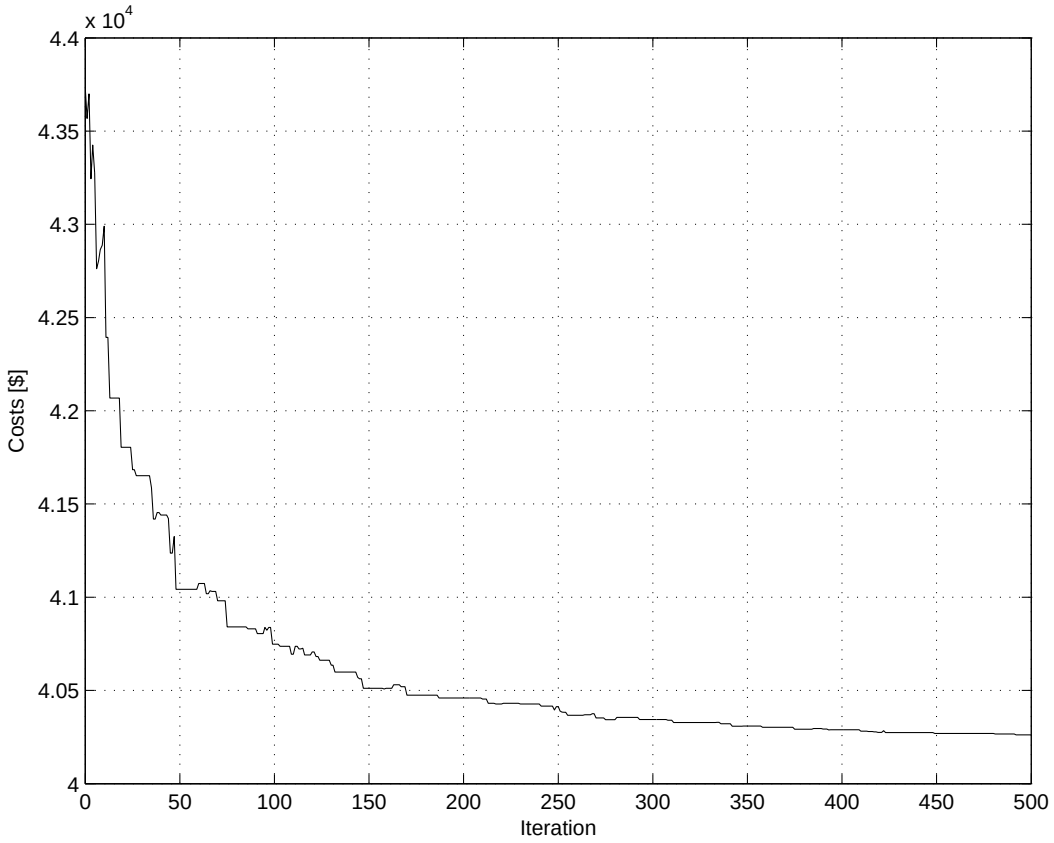


Figure 3: The Evolution Process of the best solution produced by MDE

Numerical results are in following tables. In Table 5, details of the Costs of each Evolution Process are listed. At first step, the table shows initialization and then the value of costs (the best solution in given population in given iteration) after fifty iterations for each Evolution. The best Evolution is the tenth Evolution with the lowest value of costs at the end of Evolution. In Table 6, the scheduling of power production of generators for 24 hours of the best solution produced by MDE is stated. This scheduling is related to the best solution at the end of tenth Evolution. Because of better visualization, in Figure 4, the solution is displayed in bar graph.

| Iteration | Evolution 1 [$/day] | Evolution 2 [$/day] | Evolution 3 [$/day] | Evolution 4 [$/day] | Evolution 5 [$/day] | Evolution 6 [$/day] | Evolution [$/day] | Evolution 8 [$/day] | Evolution 9 [$/day] | Evolution 10 [$/day] |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 43629 | 44086 | 44040 | 43984 | 43446 | 43945 | 44046 | 43965 | 43779 | 43727 |
| 50 | 41149 | 41125 | 41098 | 41243 | 41117 | 41077 | 41329 | 41119 | 41076 | 41042 |
| 100 | 40798 | 40806 | 40653 | 40923 | 40742 | 40808 | 40723 | 40740 | 40791 | 40748 |
| 150 | 40563 | 40634 | 40587 | 40688 | 40551 | 40669 | 40533 | 40559 | 40584 | 40512 |
| 200 | 40469 | 40468 | 40513 | 40508 | 40526 | 40555 | 40497 | 40466 | 40511 | 40460 |
| 250 | 40443 | 40416 | 40441 | 40391 | 40425 | 40486 | 40432 | 40426 | 40455 | 40413 |
| 300 | 40400 | 40388 | 40425 | 40391 | 40454 | 40432 | 40395 | 40378 | 40399 | 40344 |
| 350 | 40356 | 40367 | 40383 | 40352 | 40419 | 40400 | 40367 | 40336 | 40391 | 40309 |
| 400 | 40323 | 40352 | 40365 | 40341 | 40367 | 40358 | 40367 | 40333 | 40363 | 40289 |
| 450 | 40316 | 40324 | 40344 | 40311 | 40364 | 40332 | 40354 | 40326 | 40334 | 40270 |
| 500 | 40296 | 40314 | 40308 | 40312 | 40357 | 40322 | 40332 | 40317 | 40319 | 40262 |
| Run Time [s] | 117.6 | 119.9 | 118.0 | 116.1 | 113.7 | 117.0 | 116.1 | 111.9 | 118.9 | 119.4 |

Table 5: Details of the Costs in Evolution Processes

| Hour | Generator 1 [MW] | Generator 2 [MW] | Generator 3 [MW] | Generator 4 [MW] | Generator 5 [MW] |
|---|---|---|---|---|---|
| 1 | 10.68 | 38.66 | 60.59 | 141.27 | 162.45 |
| 2 | 11.64 | 43.66 | 61.52 | 136.56 | 185.75 |
| 3 | 14.53 | 55.18 | 85.48 | 139.74 | 184.88 |
| 4 | 40.21 | 74.62 | 87.56 | 148.92 | 184.64 |
| 5 | 39.75 | 74.23 | 82.03 | 180.47 | 188.18 |
| 6 | 13.09 | 86.11 | 103.52 | 180.38 | 232.88 |
| 7 | 40.16 | 90.86 | 98.78 | 194.17 | 210.41 |
| 8 | 35.61 | 87.76 | 118.67 | 190.10 | 230.94 |
| 9 | 21.22 | 98.89 | 131.67 | 214.72 | 233.69 |
| 10 | 31.28 | 100.64 | 134.50 | 195.04 | 253.09 |
| 11 | 30.73 | 104.71 | 137.05 | 192.84 | 265.73 |
| 12 | 27.22 | 102.11 | 156.65 | 196.45 | 269.18 |
| 13 | 24.45 | 89.03 | 154.57 | 188.11 | 258.32 |
| 14 | 38.22 | 84.61 | 146.34 | 176.08 | 254.78 |
| 15 | 27.57 | 89.28 | 124.35 | 172.68 | 249.23 |
| 16 | 34.19 | 67.76 | 107.61 | 159.46 | 218.09 |
| 17 | 29.22 | 66.21 | 92.74 | 168.48 | 208.00 |
| 18 | 24.30 | 79.65 | 104.98 | 203.01 | 203.99 |
| 19 | 32.39 | 78.22 | 127.47 | 204.28 | 220.69 |
| 20 | 28.27 | 90.83 | 147.97 | 189.98 | 257.44 |
| 21 | 36.04 | 92.27 | 129.65 | 184.41 | 247.44 |
| 22 | 33.09 | 83.52 | 96.87 | 193.85 | 205.52 |
| 23 | 21.85 | 73.48 | 67.84 | 163.65 | 206.25 |
| 24 | 12.01 | 58.03 | 92.76 | 143.74 | 160.99 |

Table 6: Power production of generators for 24 hours of the best solution produced by MDE

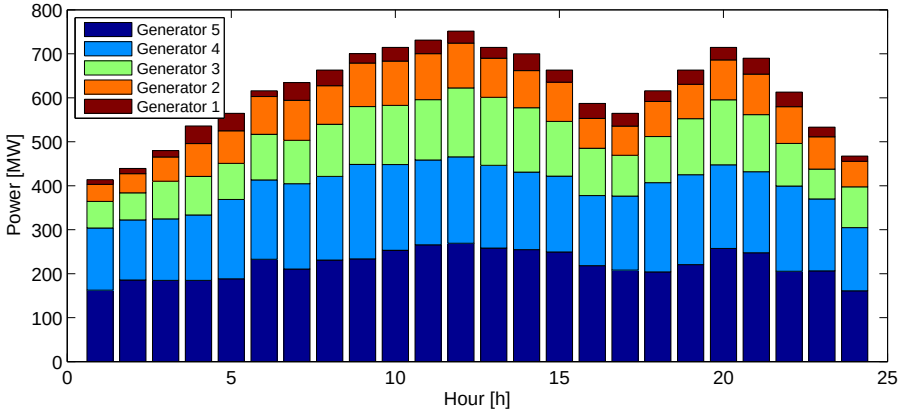


Figure 4: Power production of generators for 24 hours of the best solution produced by MDE in bar graph

### 4.3 Comparison of Results with other LD solution methods

In Table 7, the improved MDE is compared with the original MDE and some of the most recently published LD solution methods, namely Pattern Search (PS) method [1], Simulated Annealing (SA) and Matlab function `fmincon` [14]. Results produced by PS and SA are taken from [1], where the scheduling of power production is stated as well and is less importing for us in this place. Function `fmincon` is based on gradient optimization hence have problems with execution

in some points of continuous space because of the absence of derivative. On the basis of Table 7, the improved MDE produces solution on the same Fuel Cost level as the original one and in addition, it satisfies the equality constraint (3). In the original MDE, some slight violation from the equality constraint (3) may be appeared for the solution. Moreover, the improved MDE produced solution with lower costs about 8% against function `fmincon` and even about 16% against PS method and about 17% against SA method.

| Test Case | Improved MDE | Ori. MDE[4] | Ori. MDE[5] | PS | SA | `fmincon` |
|---|---|---|---|---|---|---|
| Fuel Cost [$/day] | 40262 | 40293 | 40263 | 47911 | 48621 | 43672 |
| Run Time [s] | 119.4 | 82.9 | 119.7 | 514.3 | — | 78.8 |

Table 7: Comparison of Results

# 5 Conclusion

In this paper a hybrid stochastic search technique named Modified Differential Evolution (MDE) is applied to solve the non-convex LD problem. MDE is in the framework of DE owning a mutation operator inspired from PSO and GA and a selection mechanism inspired from SA. Also, an efficient constraints handling with help of new mapping function (MF) is also suggested for the LD problem. The new MF maps unfeasible solution to a feasible one that satisfies the equality constraint. The mentioned solution method was compared with some of the most recently published techniques in the area (PS method, SA method and Matlab function `fmincon`) and original MDE.

New MF ensures the feasible solution and moreover, the improved MDE produces solution with Fuel Cost on the same level as the original one. In Test Case, the improved MDE returned even solution with lower Fuel Cost than the original MDE. Other methods such as Pattern Search (PS) method, Simulated Annealing (SA) and Matlab function `fmincon` produced markedly worse solutions than the improved MDE.

This work shows that by combining positive characteristics of individual search techniques, the resulted hybrid method can present more search capability and robustness than the individual techniques being combined. This enhanced search capability and robustness is suitable to solve complex optimization problems like non-convex LD.

The formulated LD problem has several drawbacks which could be resolved in future work. In proposed version, the LD problem is too deterministic. In real world, the demands are not known so accurately and other coefficients as well. Then, LD does not consider the possibility of commitment[6] of generators in time horizon and changes of commitments during time period. The elimination of all these drawbacks is challenge for future work.

# References


[1] J. AL-SUMAIT. *Solving Dynamic Economic Dispatch problems using Pattern Search based methods with particular focus on the West Doha Power Station in Kuwait.* PhD thesis, University of Southampton, Faculty of engineering, science and mathematics, September 2010.

[2] N. AMJADY and H. SHARIFZADEH. Solution of non-convex economic dispatch problem considering valve loading effect by a new modified differential evolution algorithm. *Elsevier – Electrical power and energy system*, January 2010.


[4] The maximum iteration of original MDE is the same as for improved MDE, i.e. $s_{max} = 500$.

[5] To reach the same level of Run Time as improved MDE, the maximum iteration of original MDE is $s_{max} = 700$.

[6] Unit commitment determines whether given unit is switched on or off.

[3] F. BORRELLI, A. BEMPORAD, and M. MORARI. *Constrained Optimal Control and Predictive Control for linear and hybrid systems*. Springer, Berlin, 2010.

[4] S. CHOWDHURY, S. P. CHOWDHURY, and P. CROSSLEY. *Microgrids and Active Distribution Networks*. The Institution of Engineering and Technology, London, 2009.

[5] R. EBERHART and J. KENNEDY. A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pages 39–43, 1995.

[6] JI-Y FAN and L. ZHANG. Real-time economic dispatch with line flow and emission constraints using quadratic programming. *IEEE Transactions on Power Systems*, May 1998.

[7] D. E. GOLDBERG. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[8] S. KIRKPATRICK, C. D. GELATT, and M. P. VECCHI. Optimization by simulated annealing. *Science*, May 1983.

[9] K. MACEK and M. STŘELEC. The micro-grid as a stochastic hybrid system - two formal frameworks for advanced computing. In *SMARTGREENS*, pages 141–144, 2012.

[10] J. NANDA, L. HARI, and M.L. KOTHARI. Economic emission load dispatch with line flow constraints using a classical technique. *IEE Proc Gener Transm Distrib*, January 1994.

[11] J. PARIKH and D. CHATTOPADHYAY. A multi-area linear programming approach for analysis of economic operation of the indian power system. *IEEE Transactions on Power Systems*, February 1996.

[12] K. V. PRICE, R. M. STORN, and J. A. LAMPINEN. *Differential Evolution A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag, Berlin, Germany, 2005.

[13] O. ŠUBRT. Heuristics vs. mathematical programming in load dispatch optimization. Bachelor's thesis, Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, June 2011.

[14] Matlab documentation of function fmincon. `http://www.mathworks.com/help/toolbox/optim/ug/fmincon.html`. October 2012.

---

Bc. Ondřej Šubrt
subrton2@fjfi.cvut.cz

Mgr. Karel Macek
karel.macek@utia.cas.cz

Doc. Ing. Miroslav Virius, CSc.
miroslav.virius@fjfi.cvut.cz