A Novel Movable Partitions Approach with Neural Networks and Evolutionary Algorithms for Solving the Hydroelectric Unit Commitment Problem

Pedro de Lima Abrão
Centro Federal de Educação Tecnológica de Minas Gerais
CEFET-MG
Av. Amazonas, 7675
Belo Horizonte, Brazil
plabrao@lsi.cefetmg.br

Elizabeth F. Wanner
Centro Federal de Educação Tecnológica de Minas Gerais
CEFET-MG
Av. Amazonas, 7675
Belo Horizonte, Brazil
efwanner@decom.cefetmg.br

Paulo E. M. Almeida
Centro Federal de Educação Tecnológica de Minas Gerais
CEFET-MG
Av. Amazonas, 7675
Belo Horizonte, Brazil
pema@lsi.cefetmg.br

ABSTRACT

This paper presents a method based on Neural Networks and Evolutionary Algorithms to solve the Hydroelectric Unit Commitment Problem. A Neural Network is used to model the production function and a novel approach based on movable partitions is proposed, which makes it easier to model the desired power output equality constraint in the optimization modeling. Three evolutionary algorithms are tested in order to find optimized operation points: differential evolution DE/best/1/bin, a balanced version of DE and Particle Swarm Optimization algorithm (PSO). The results show that the proposed method is effective in terms of water consumption, reaching in some cases more than 1% of economy whether compared to the traditional commitment strategy.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: [Constrained optimization]
; I.2.6 [**Learning**]: [Connectionism and neural nets]; J.2 [**Phisical Sciences and Engineering**]: [Engineering]

Keywords

Optimization algorithms, Operations research, Neural systems, Energy generation and storage, Renewable energy

1. INTRODUCTION

The hydro unit commitment (UC) problem aims to determine, for a time stage, the status of the generating units and their respective dispatch (in MW) in a hydroelectric generation plant, so that the total power demand is satisfied, the operating cost is minimized and the set of constraints is guaranteed. This work presents an optimum dispatch model which adopts, as the performance criteria, the minimum consumption of hydro resources. The main idea behind this study is that the water is a scarce resource, for which the generation units are competing.

In Brazil, the hydro generation is the main source of electric energy, corresponding to 81.7% of the total production. The energy generation grid is predominantly connected. Only 3.4% of its electric production capacity is out of the national system. In this way, there are centers which are responsible to imply the operation points in MW for each generation plant. From that, each plant must commit its generation units in order to meet the required demand.

The dispatch can be divided into long, medium and short-term planning [1]. The focus here will be in the short-term planning, which involves, in this case, to find for a specific time slice the respective dispatch for each generation unit in order to produce the desired total power output. Normally in a hydroelectric plant, the control system performs an equal division of the power demand in MW among its generation units. However, this approach does not take into consideration if each unit is operating at its particular efficient point. Considering the non-linearity of the production function and the high number of continuous and discrete constraints involved, this problem can be considered complex. In this way, many approaches have been developed around the world trying to solve it.

Lagrangian Relaxation (LR) is one of the most widely used methods to solve the UC problem, mainly over thermal plants [3] [15] [8]. This approach is very attractive, specially because of the possibility to relax the problem constraints, such as the spinning reserve and demand supply requirements, which makes possible to solve multiple independent subproblems, one for each generating unit, without any explicit coupling between them. However, the solution to UC problem by LR is sensitive to changes of the Lagrangian multipliers and highly dependent upon the size of the duality gap [11], which in large power systems such as the Brazilian one, can increase dramatically.

Another possible approach is to use decomposition techniques. Finardi and Silva proposed a solution in which two subproblems were created [7]. The first one corresponding to a linear programming which main purpose was to establish the total volume of water to be discharged by each hydro plant, in each time stage, without considering the units individually. The second problem, with respect to the nonlinearity, is associated with the hydro units and the combinatorial nature of the commitment. Forbidden zones constraints

GECCO'13, July 6–10, 2013, Amsterdam, The Netherlands.

were considered and the Branch and Bound method was used to determine feasible states. Thus, due to the various steps needed, this strategy becomes complex and harder to employ in systems that require small response times.

Considering the difficulties stated before, an alternative approach consists of heuristic methods since they have the merits of being more flexible while handling nonlinear and discrete variables, in addition to easy coding and implementation whether compared to classical optimization methods. Pancholi e Swarup [14] used the Particle Swarm Optimization (PSO) evolutionary technique to solve the economic dispatch problem taking into consideration security constraints. The results were compared to other techniques such as linear programming and quadratic programming achieving reduced computational costs.

Yalcinoz and Short [17] presented a solution for the economic dispatch problem based on a modified Hopfield Neural Network to deal with transmission capacity constraints. The proposed method achieved efficient solutions to power systems up to 120 units. When compared to a classical optimization program based on quadratic programming, the solution obtained the same results (operation cost [$ / h]) while demanding less time to run.

Liu and Li [11] proposed an approach that establishes the hydro dispatch for T time stages. It was taken into consideration the desired power demand, operation regions and costs to start and to shutdown turbines. An enhanced version of the PSO was also used to search for optimal solutions. According to the results, the method outperformed the classical PSO algorithm and other methods such as genetic algorithms and dynamic programming in relation to the water consumption. However, the modelling applied did not explain in details the relation among the involved variables to establish the production function.

This paper presents a method based on a Neural Network to model the production function and the novel movable partition approach which makes it easier to model the desired power output equality constraint in the optimization problem. Three evolutionary algorithms are tested in order to find optimized operation points: differential evolution DE/best/1/bin, a balanced version of DE and Particle Swarm Optimization algorithm (PSO).

This paper is organized as follows. In section 2 the applied model is presented. Section 3 presents the evolutionary algorithms which will be applied. Section 4 shows the application of the proposed solution, the results obtained and some analysis among the algorithms tested and section 5 concludes the paper.

2. MODELING THE PROBLEM

Hydropower plants use the hydraulic potential to generate energy. A common installation is composed by a dam, responsible to hold back water, creating the reservoir. The gravity pulls the water to the penstock, a pipeline that leads to the turbine. Water builds up pressure as it flows through this pipe. Then the water strikes and turns the large blades of a turbine, which is attached to a generator above it by way of a shaft. Figure 1 shows a schematic representation of a hydroeletric plant.

The vertical distance between the forebay level and the tailrace level (intake to turbine) influences the resulting pressure at the bottom. This measure is called gross head of the turbine. However, when water is flowing, the pressure

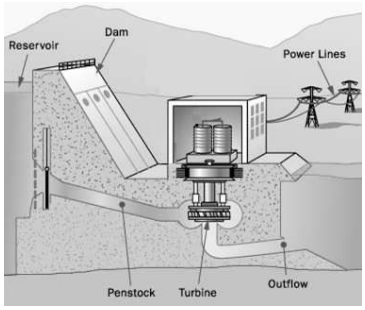

Figure 1: Hydroelectric plant representation

at the bottom of the penstock will be always less than the gross head due to energy losses within the pipeline. This measure, which takes into consideration the penstock losses, is called net water head [8].

Different generation units even operating with the same kind of turbine can have different net water heads associated when discharging the same amount of water. This happens mainly since their penstocks have different geometries, which leads to different losses, having a direct influence on the unit efficiency.

The production function of a hydro generating unit is dependent on the turbine-generator efficiency, the net water head, and the turbine water discharge. Both net water head and efficiency are described as nonlinear functions, which are dependent on the control variables (turbine water discharge and spillage) and the state variables (volumes at the beginning and end of each stage) [7]. The relationship among these variables can be described by curves known as Hill Diagram.

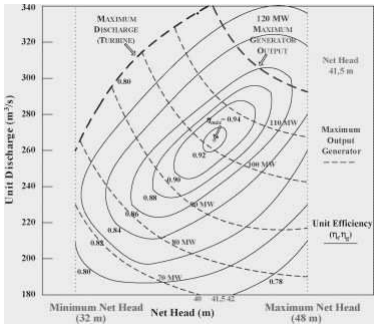

Figure 2: Hill Diagram Example. Adapted from [7].

As can be seen in Figure 2, the Hill Diagram provides basically two kinds of curves for a generation unit: one representing the relationship among the net water head, discharge and the efficiency, represented in %, and the other representing the relationship among the net water head, discharge and power output, normally represented in MW. Both curves are related since it is possible to define power output from the turbine efficiency [1].

A common approach to estimate the production function is to use the information provided by the Hill Diagram. Thus, since the Hill Diagram only provides specific curves to represent the relationships among the involved variables, further techniques have to be implemented to create a continuous function. Finardi represented this function using a

polynomial function of degree seven. The coefficients were obtained using a multivariable nonlinear regression technique [7]. The problem of this technique is that losses of precision can occur during the definition of a specific degree for the polynomial and during the process of estimation of the coefficients.

In this paper, an Artificial Neural Network (ANN) approach is proposed with the same purpose. The major benefit of a ANN is its ability to learn and therefore to generalize. Generalization refers to the capacity of producing reasonable outputs for inputs not encountered during training (learning) phase [9]. This ability make it possible for neural networks to solve complex problems. From the extracted points in the Hill Diagram a multilayer perceptron was used to estimate the production function. The objective of this ANN is to estimate the relationship among variables net water head, power output and unit discharge. Considering that it is necessary to calculate the water discharge for each unit, the production function has to be represented in terms of this last variable.

The set of available points was separated randomly in two subsets: 80% for training and 20% for validation [9]. Then the points were normalized in the interval [-1, 1]. The network with 2 hidden layers having 4 and 7 neurons respectively was trained using the Levenberg-Marquadt method [2], with fixed learning rate 0.3 and momentum constant 0.6.

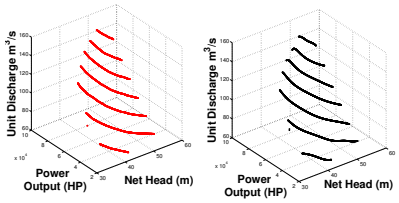

Figure 3: Hill Diagram and Neural Network Points

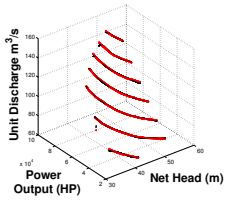

Figure 4: Neural Network Recall over Hill Diagram Points

Figure 3 shows the graphical representation of the original points extracted from the Hill Diagram and the corresponding points obtained through the neural network recall. Figure 4 shows both sets of points plotted together. Testing the validation set the mean error encountered was 0.3182 m^3/s with standard deviation 0.0910 m^3/s, which is very acceptable considering the precision of the hydro plants control systems. The mean squared error is 0.1934 m^3/s.

As mentioned before, the problem addressed in this paper can be stated as: given a power demand to be met by the hydro plant p, the units must be committed so that the total hydro plant discharge is minimized. Thus, each unit has power output forbidden zones which must be avoided during the dispatch. In addition, some plants need to establish that each unit and the plant as a whole are discharging water within a predetermined range. Formally, the problem can be mathematically modelled as follows:

$$Minimize\ C = \sum_{i=1}^{N(p)} q_i \tag{1}$$

$$q_i = Q_{NN}(p_i, hl_i) \tag{2}$$

subject to:

$$P^t = \sum_{i=1}^{N(p)} p_i \tag{3}$$

$$p_i^{min} \leq p_i \leq p_i^{max} \tag{4}$$

$$q_i^{min} \leq q_i \leq q_i^{max} \tag{5}$$

$$Q_p^{min} \leq \sum_{i=1}^{N(p)} q_i \leq Q_p^{max} \tag{6}$$

in which:

$N(p)$	number of generation units in the hydro plant p;
q_i	water discharge of unit i (m^3/s);
Q_{NN}	water discharge function calculated using the proposed artificial neural network (m^3/s);
p_i	production of unit i (MW);
hl_i	net water head of unit i (m);
P^t	required power demand for plant p (MW);
p_i^{min}	minimum power output allowed to unit i (MW);
p_i^{max}	maximum power output allowed to unit i (MW);
q_i^{min}	minimum discharge allowed to unit i (m^3/s);
q_i^{max}	maximum discharge allowed to unit i (m^3/s);
Q_p^{min}	minimum discharge allowed to hydro plant p (m^3/s);
Q_p^{max}	maximum discharge allowed to hydro plant p (m^3/s).

3. EVOLUTIONARY ALGORITHMS

3.1 Overview of Classical DE Algorithm

Differential evolution (DE) has recently proven to be an efficient method for optimizing real-valued multi-modal objective functions. Besides its good convergence properties and suitability for parallelization, DE's main assets are its conceptual simplicity and ease of use. The first written article on DE appeared as a technical report by R. Storn and K. V. Price in 1995 [16]. One year later, the success of DE was demonstrated at the First International Contest on Evolutionary Optimization in May 1996, which was held in conjunction with the 1996 IEEE International Conference on Evolutionary Computation (CEC). Thereafter, DE emerged as a very competitive form of evolutionary computation, turning out as one of the best among important competitions [4].

DE searches for a global optimum point in a D-dimensional real parameter space. The classical algorithm begins with a randomly initiated population of D-dimensional real-valued

parameter vectors of size NP. In one of the simplest forms of DE-mutation, to create the donor vector for each ith target vector from the current population $X_t = \{x_{t,i}; i = 1, ..., NP\}$, three other distinct individuals, say $\vec{x}_{r_1^i}$, $\vec{x}_{r_2^i}$ and $\vec{x}_{r_3^i}$ are sampled randomly. The indices r_1^i, r_2^i and r_3^i are mutually exclusive integer random numbers from the range [1, NP]. Then, a differential vector is created, which is the difference between $\vec{x}_{r_2^i}$ and $\vec{x}_{r_3^i}$ scaled by a number F (that typically lies in the interval [0.4, 1]). The differential vector is then added to $\vec{x}_{r_1^i}$, that is named base vector [4]. This generates a mutant vector. The process at the generation t can be expressed as follows:

$$\vec{v}_{t,i} = \vec{x}_{t,r_1^i} + F(\vec{x}_{t,r_2^i} - \vec{x}_{t,r_3^i}) \tag{7}$$

The mutant population generated $V_t = \{v_{t,i}; i = 1, ..., NP\}$ is then combined to the current population producing the offspring $U_t = \{u_{t,i}; i = 1, ..., NP\}$. In the classic version of the algorithm, for each dimension j of each individual i of the population a discrete recombination with probability C $\in$ [0, 1] is applied, as follows:

$$u_{t,i,j} : \begin{cases} v_{t_{i,j}}, & \text{if } rand_{i,j}[0,1] \leq C \vee j = \delta_i \\ x_{t,i,j}, & \text{otherwise} \end{cases} \tag{8}$$

in which $rand_{i,j}[0, 1]$ is a uniformly distributed random number, which is called anew for each jth component of the ith generated individual. $\delta_i \in$ [1, 2, ..., D] is a randomly chosen index, which ensures that U_t gets at least one component from V_t.

The next step of the algorithm calls for selection to determine whether the parent or the mutant vector survives to the next generation, i.e., at $T = T + 1$. The selection operation is described as:

$$\begin{aligned} \vec{X}_{i,T+1} &= \vec{U}_{i,T} \quad \text{if } f(\vec{U}_{i,t}) \leq f(\vec{X}_{i,t}) \\ &= \vec{X}_{i,T} \quad \text{if } f(\vec{U}_{i,t}) > f(\vec{X}_{i,t}) \end{aligned} \tag{9}$$

in which $f(\vec{X})$ is the objective function to be minimized (minimization is considered). As each generated vector only replaces the corresponding parent vector in the next generation if its fitness is better, then it's guaranteed that population either gets better (with respect to the minimization of the objective function) or remains the same in fitness status, but never deteriorates [4].

3.2 Balanced Approach of the DE Algorithm

DE has shown to be effective on a large range of classical optimization problems, and it showed to be more efficient than other well known evolutionary algorithms [4] [16]. In addition to classical DE, many other mutation operators have been proposed to improve its convergence capabilities. However, not all of the DE mutation operators are equally efficient. Some favor the exploration of the search space, while some other operators favor its fast exploitation. Explorative mutation operators have a greater possibility of locating the minima of the objective function, but generally need more iterations (generations). On the other hand, the exploitive mutation operators rapidly converge to a minimum of the objective function. In this case, there exists the risk of premature convergence to a suboptimal solution.

Epitropakis et al proposed a hybrid approach that makes a linear combination between an explorative and exploitive mutation operator [12]. More specifically, the mutant individual $\vec{v}_{t,i}$ is generated using the following equation:

$$\vec{v}_{t,i} = \xi.v_{t,i}^a + (1 - \xi).v_{t,i}^b \tag{10}$$

in which ξ determines the influence of the explorative over the exploitive mutation operator, $v_{t,i}^a$ denotes a explorative mutation operator, while $v_{t,i}^b$ denotes a exploitive mutation operator. For example, $v_{t,i}^a$ can represent the mutation operator of the algorithm DE/rand/1/bin, $(\vec{x}_{t,r_1^i} + F(\vec{x}_{t,r_2^i} - \vec{x}_{t,r_3^i}))$ and $v_{t,i}^b$ can represent the mutation operator of the algorithm DE/best/1/bin $(\vec{x}_{t,best} + F(\vec{x}_{t,r_3^i} - \vec{x}_{t,r_2^i}))$. The parameter ξ can be calculated using one of the following equations:

$$\xi_1 = (1 + \rho).2^{-c.g}, \tag{11}$$

$$\xi_2 = \rho.2^{-c.g}, \tag{12}$$

$$\xi_3 = \frac{\xi_1 + \xi_2}{2}, \tag{13}$$

where g is the current generation, $\rho \in [0, 1]$ is a random number from the uniform distribution, and c, the noise decay constant, is calculated as one tenth of the maximum allowed number of generations. According to the author, this scheme can lead to reliable optimization of unknown objective functions, since it alleviates problems generated by poor selection of the user defined parameters, such as decreased rate of convergence, or even divergence and premature saturation [12].

3.3 Particle Swarm Optimization (PSO)

PSO was introduced by Kennedy and Eberhart in 1995. It was inspired by the swarm behaviour as is displayed by a flock of birds, a school of fish, or even human social behaviour being influenced by other individuals [10].

PSO consists of a swarm of particles moving in an N-dimensional, real-valued search space of possible problem solutions. Every particle has a position vector $\vec{x}$ encoding a candidate solution to the problem, a velocity vector $\vec{v}$ and a small memory that stores its own best position so far $\vec{p}$. Moreover, a global best position among the swarm $\vec{g}$ is stored. In the classical version of the algorithm the velocity and position of the particles are updated as follows:

$$\begin{aligned} v_{i,k} = \omega.v_{i,k-1} + c_1.R_1^k.[p_{i,k-1} - x_{i,k-1}] + \\ c_2.R_2^k.[g_{k-1} - x_{i,k-1}], \end{aligned} \tag{14}$$

$$x_{i,k} = v_{i,k} + x_{i,k-1}, \tag{15}$$

in which ω is a inertia weight factor, $v_{i,k}$ is the velocity of particle i at kth iteration, c_1 is the cognitive factor, c_2 is the social factor, R_1^k and R_2^k are two random numbers uniformly distributed in the range of [0, 1], $x_{i,k}$ is the current position of particle i at iteration k, $p_{i,k}$ is the position of particle i at kth iteration and g_k is the best position in the swarm at kth iteration.

3.4 Problem Model

As described before, the proposed model used in this paper defines four constraints: one regarding to the plant to

meet the desired power output, one to ensure that each unit is respecting its output limits, one to evaluate if each unit is demanding discharges in accordance to the plant capabilities and the last one to evaluate if the total discharge is within a predetermined range.

The most popular range constraint handling strategy in evolutionary algorithms is to use a penalty function (PF) added to the individuals fitness values. Obviously, the individuals will avoid getting close to prohibited areas due to bigger fitness values that they reach when they do not meet the constraints established.

3.4.1 Movable Partition Approach

It is also possible to use the PF approach to satisfy equality constraints. However, it can make it harder for the optimization technique in the process of finding global minimum values because the fitness of the individuals can change very quickly for each difference between the current value of the equality variable and the required constant for it.

Here a movable partitions approach is proposed to deal with the desired power output equality constraint. This modelling consists of, instead of using a variable to represent the power output for each one of the $N(p)$ production units, define a compartment with fixed size representing the total power output for the plant separated by $N(p)-1$ movable partitions. Using this approach, in addition to eliminate the equality constraint of the problem, one less dimension is required to represent the plant what improves the search capabilities of the algorithm. Figure 5 illustrates this technique.

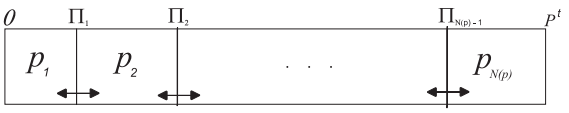

Figure 5: Movable Partitions Approach illustration

Observing Figure 5, $N(p)-1$ partitions are used to determine $N(p)$ unit power outputs. In this way, the objective of the algorithm is to move these $N(p)-1$ partitions over the space to find optimized solutions. The conversion calculus from partitions to power output is performed by a simple subtraction, as below:

$$p_i = \Pi_i - \Pi_{i-1}, \quad 1 \leq i \leq N(p), \tag{16}$$

in which Π_i represents the partition i, $\Pi_0 = 0$ and $\Pi_{N(p)} = P^t$.

From this approach, the partition overpassing problem can occur. If one partition overpasses others during the algorithm processing then the power demand dispatched would be different from the desired. In this case, a possible solution is to sort the partitions according to their new indexes to ensure the commitment to this constraint, what is simple to implement and does not cause any issues to the algorithm behavior.

3.4.2 Fitness Function

After the range constraints be treated by PF method, the equality constraint be solved using the movable partition approach and based on the fitness function proposed by Liu and Li [11] the initial objective (Equation 1) can be turned into an unconstrained optimization problem, which can be formulated as:

$$Minimize\ F = \sum_{i=1}^{N(p)} \{\ q_i + \sum_{j=1}^{N(c)} [\ |\varphi_{i,j}|\ .\ \lambda_j\ ]\ \} \tag{17}$$

- $\varphi_{i,1} = \max(p_i^{max}, p_i^t) - p_i^{max}$
- $\varphi_{i,2} = \min(p_i^{min}, p_i^t) - p_i^{min}$
- $\varphi_{i,3} = \max(q_i^{max}, q_i) - q_i^{max}$
- $\varphi_{i,4} = \min(q_i^{min}, q_i) - q_i^{min}$
- $\varphi_{i,5} = \max(Q_p^{max}, \sum_{i=1}^{N(p)} q_i) - Q_p^{max}$
- $\varphi_{i,6} = \min(Q_p^{min}, \sum_{i=1}^{N(p)} q_i) - Q_p^{min}$

in which F is the fitness value of the individuals, $N(c)$ is the number of constraints (in this case 6), λ_j is the penalty operator of jth constraint and $\varphi_{i,j}$ represents the jth constraint according to the individual i.

4. APPLICATION OF THE OBTAINED SOLUTION

4.1 Plant and Algorithms Information

The proposed model was applied to the hydroelectric plant of Três Marias in Brazil. This plant has 6 units on the upstream of the São Francisco River but only 5 are fully operational at the moment. The power output interval tested for each unit was (35 to 61)MW which enables the power rating of 305MW (5×61MW). The turbines are identical, but they have different net heads due to their different geometries in the penstocks. The gross head used for the the units was 56m. The water dispatch interval allowed for each turbine used was (60 to 150)m^3/s and the total plant dispatch interval used was (320 to 700)m^3/s. Six different typical power demands for the plant were tested: 210MW, 230MW, 250MW, 270MW, 290MW and 300MW.

In order to optimize the proposed model, three evolutionary algorithms, the DE/best/1/ bin (DE_1), the balanced version of the DE proposed by Epitropakis ($DE_{2,1}$) and the classical Particle Swarm Optimization (PSO) algorithm, were tested. The only stop criteria used was number of generations which was equals to 50. It was chosen the DE/rand/1/bin and DE/best/1/bin as the explorative and exploitive mutation operations, respectively, in the balanced version of the DE algorithm. The parameter ξ_3 was used to perform the balancing. All the other parameters used in this algorithm were the same used in the DE_1 mentioned above. The PSO used a self-adaptive decreasing inertia weight schema [11]. The number of particles used was also 50. The parameters cognitive factor c_1, social factor c_2, maximum inertia weight w_{max}, minimum inertia weight w_{min}, as well as DE parameters are shown in table 1.

Table 1: Algorithms Parameters

DE Parameters			
F	C	NP	-
0.4	0.6	50	-
PSO Parameters			
c_1	c_2	w_{max}	w_{min}
2.0	3.5	0.9	0.4

The constraint penalties adopted and net water heads for each unit are contained in the table 2. They were found empirically.

Table 2: Model Parameters

Constraint Penalties Values					
λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
6.0	6.0	5.0	5.0	5.0	5.0
Net Water Heads (hl_i)					
hl_1	hl_2	hl_3	hl_4	hl_5	-
52.5	54.0	55.0	54	52.5	-

4.2 Results

Each algorithm tested was executed 30 times for each power demand. Table 3 shows the best solutions encountered by each algorithm through the executions. In addition, it is shown the equal division of the power demand among the units available, what is the common practice of the plant studied. As can be seen in the table 3 all algorithms reached the same best fitness value. The best discharge gain happened for the demand of 230MW, reaching 4.95 less m^3/s what represents 1.05% of water economy.

Table 3: Unit Commitment Results

Load Demand (MW)	Method	Unit Power Output (MW)					Water Disch.
		#1	#2	#3	#4	#5	(m^3/s)
210	DE_1	39.8	40.0	50.3	40.0	39.8	426.45
	$DE_{2,1}$	39.8	40.0	50.3	40.0	39.8	426.45
	PSO	39.8	40.0	50.3	40.0	39.8	426.45
	=	42.0	42.0	42.0	42.0	42.0	427.85
230	DE_1	40.3	53.6	54.8	40.8	40.3	465.40
	$DE_{2,1}$	40.3	40.8	54.8	53.6	40.3	465.40
	PSO	40.3	53.6	54.8	40.8	40.3	465.40
	=	46.0	46.0	46.0	46.0	46.0	470.35
250	DE_1	40.5	56.2	56.2	56.2	40.5	505.23
	$DE_{2,1}$	40.5	56.2	56.2	56.2	40.5	505.23
	PSO	40.5	56.3	56.2	56.3	40.5	505.23
	=	50.0	50.0	50.0	50.0	50.0	508.57
270	DE_1	40.7	57.1	56.8	57.1	58.0	546.87
	$DE_{2,1}$	58.0	57.1	56.8	57.2	40.7	546.87
	PSO	58.1	57.1	56.8	57.15	40.7	546.87
	=	54.0	54.0	54.0	54.0	54.0	547.67
290	DE_1	58.5	57.7	57.3	57.7	58.5	588.64
	$DE_{2,1}$	58.5	57.7	57.3	57.7	58.6	588.64
	PSO	58.5	57.7	57.3	57.7	58.6	588.64
	=	58.0	58.0	58.0	58.0	58.0	588.73
300	DE_1	60.5	59.8	59.3	59.8	60.5	613.19
	$DE_{2,1}$	60.5	59.8	59.3	59.8	60.4	613.19
	PSO	60.5	59.8	59.3	59.8	60.4	613.19
	=	60.0	60.0	60.0	60.0	60.0	613.74

4.3 Algorithm Analysis

As said before, all algorithms reached the same best solution at least once during their executions. In this section, the three algorithms will be compared in order to determine which of them is the most suitable to optimize the optimization problem.

4.3.1 Results Analysis

In order to analyse the results found for each algorithm during their executions, a box plot was generated for each power demand tested. Box plot is a compact representation that encodes the minimum, maximum, mean, median, and quartile information of a distribution [5]. The outliers are the observations that present greater distances to the rest of the distribution. Figure 6 shows the results distribution box plot for the the 290MW power demand. The x-axis represents the three algorithms tested and the y-axis represents the fitness value. The filled point is the mean of the distribution and the unfilled points are the outliers. The other box plots will not be shown in this paper because they had similar forms.

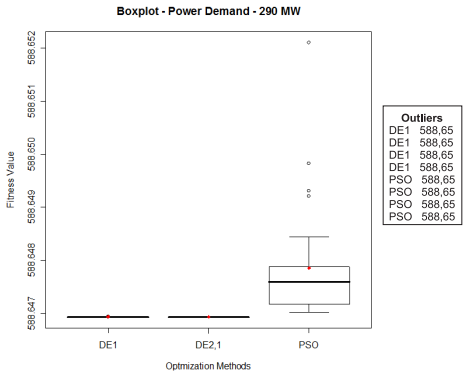

Figure 6: Box Plot - Demand 290MW

Analysing that box plot, it is possible to say that the algorithms had similar results. It is valid to notice that y-axis scale precision in Figure 6 is 10^{-3}, what makes the differences among the samples very discrete. Only a line was observed for the algorithms DE_1 and $DE_{2,1}$ in the box plot because the quartiles are very close to the medians. In the PSO algorithm, the difference between the upper and lower quartiles are also very subtle. All the outliers are very close to the rest of the distribution (differences less than 10^{-1}). Hence, there are indications that the algorithms are executing with robustness and usually find similar results during all the executions.

To testify the conclusions provided by the box plots, a Permutation Test was also applied to the algorithm results. Permutation tests are non-parametric statistical methods which estimate a reference distribution by calculating all possible values (or at least a considerably large set) of a statistic test rearrangements of the labels on a set of observed data points [13]. Any performance index can be chosen as the test statistic object. For the comparison of each pair of different algorithms, the following null and alternative hypothesis are formulated:

- Null hypothesis H_0: there is no performance difference between the two algorithms;
- Alternative hypothesis H_1: there is a performance difference between the two algorithms.

In other words, the null hypothesis states that *There is no evidence that one of the algorithms is intrinsically better than the other one, since the difference between their performance indices could be explained as being a realization of an underlying probability distribution of the test object that is neutral (the distribution does not favor any algorithm).* The central assumption of the method is that, if the observed result has arisen by chance, then this value will not seem unusual in a distribution of results obtained through many random relabellings of the samples.

Permutation tests can be used for building a reference distribution in which it is possible to determine if the differ-

ence which has been observed between the sample means is enough to reject H_0. That test can be performed as follows [6].

1. Find the sample means $\overline{A}$ and $\overline{B}$.
2. Find the observed difference $d = \overline{A} - \overline{B}$.
3. Arrange the observations of A and B in a single vector $S = [a_1,\ a_2,\ \ ...\ a_{n_A},\ b_1,\ b_2,\ \ ....\ b_{n_B}]$.
4. Make i $\leftarrow = 1$.
5. For a high pre-determined number of times:
   a) Shuffle the elements of S to create S'.
   b) Calculate $\overline{x_a} = \frac{1}{n_A} \sum_{i=1}^{n_A} S'_i$.
   c) Calculate $\overline{x_b} = \frac{1}{n_B} \sum_{i=n_A+1}^{n_B} S'_i$.
   d) Make $X_i \leftarrow \overline{x_a} - \overline{x_b}$.
6. Sort X in ascendant order .
7. Calculate $p_{value}(d) = P(X \leq d)$.
8. If $p_{value}(d) \leq \frac{\alpha}{2}$ or $p_{value}(d) \geq 1 - \frac{\alpha}{2}$ then reject H_0 and accept H_1. Otherwise, keep H_0 as a non-discarded possibility.

In step 8, the parameter α is an user-defined confidence level for the rejection of the null hypothesis.

In the algorithms results case, A stands for the results distribution of one algorithm and B stands for the results distribution of another algorithm. It is worthwhile to notice that the Permutation Test is applied between two distributions. H_0 indicates that there are no sufficient statistic evidences that enable the rejection of the null hypothesis, which means that it can not be evaluated that the algorithm A is better or worse than the algorithm B. H_1 indicates that the results are different. In this case, if d is positive then it can be noticed that B has better results on average (the problem stands for minimization). The same way, if d is negative then it can be noticed that A has better results on average. The confidence level defined α is 0.05 (95%) and the number of times that the statistical object were performed (step 5) is 5000. The Permutation Test was performed for all the power demands tested using the following A vs. B: DE_1 vs. $DE_{2,1}$, DE_1 vs. PSO and $DE_{2,1}$ vs. PSO.

In the first part of the test, the difference of the objective function means was used as the statistical object. For all the tests performed, the hypothesis H_0 was true, indicating that there is no evidence that one of the algorithms is intrinsically better than the other one. Figure 7 shows the histogram of the test for the 210MW power demand, DE_1 vs. PSO. Notice that the observed result is not statistically significant since it is located inside the confidence interval. The other histograms will not be shown in this paper because they had similar forms.

4.3.2 Convergence Analysis

Figure 8 shows the mean convergence lines for each power demand tested. Each graph shows, for all evolutionary algorithms, the mean of the fitness value encountered at each generation among the 30 executions.

Table 4 shows the mean number of generations that each algorithm executed until find the best fitness value. It was just considered fitness value improvements higher than 5×10^{-3} in relation to the best previous fitness value encountered.

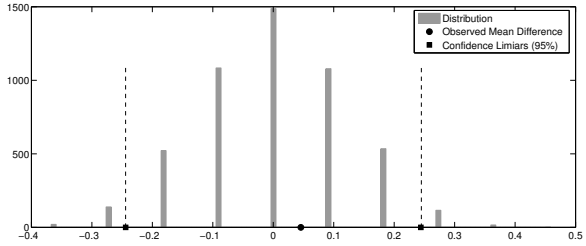

Figure 7: Permutation Test Histogram - DE_1 vs. PSO - 270MW

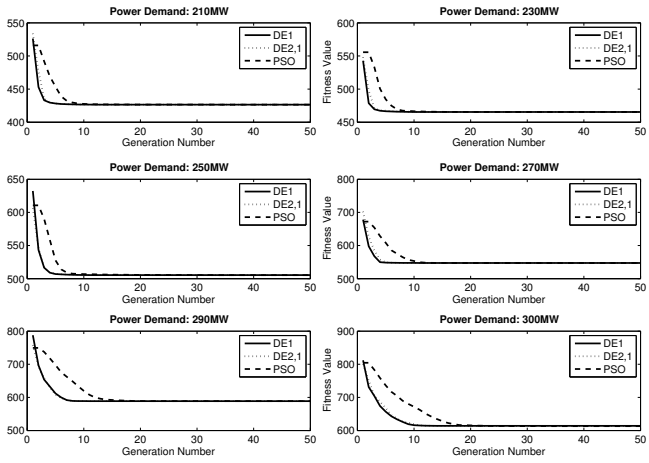

Figure 8: Mean Convergence Lines

Table 4: Mean Number of Generations to Converge

Method	Load Output (MW)	Mean # of Gen.	Stand. Dev.	General Mean # of Gen.	General Stand. Dev.
DE_1	210	14.0	1.9	14.8	2.4
	230	13.4	1.1		
	250	15.3	1.3		
	270	13.4	2.5		
	290	14.9	1.8		
	300	17.8	2.4		
$DE_{2,1}$	210	15.4	1.6	16.4	2.5
	230	15.2	1.5		
	250	16.7	1.5		
	270	16.2	3.1		
	290	15.7	1.6		
	300	19.7	2.2		
PSO	210	23.9	7.0	25.6	8.8
	230	23.8	4.8		
	250	24.7	7.8		
	270	18.1	5.8		
	290	24.0	5.2		
	300	39.1	5.1		

The algorithm DE_1 demanded less generations on average to find the results than the other methods tested, but its results were close to $DE_{2,1}$ algorithm. To better evaluate the convergence of the algorithms, another Permutation Test was applied. In this case, the number of generations required to algorithm convergence was used as statistical object. In this case, H_0 indicates that there are no sufficient statistic evidences to reject the hypothesis that the algorithm A converges demanding less generations than the algorithm B and vice versa. H_1 with d negative indicates that the algorithm A converges faster with 95% confidence

level. The same way, H_1 with d positive indicates that the algorithm B converges faster with the same confidence level. Table 5 shows the results for the disputes A vs. B.

Table 5: Permutation Test Applied to the Algorithms Convergence

Power Output (MW)	DE_1 vs. $DE_{2,1}$	DE_1 vs. PSO	$DE_{2,1}$ vs. PSO
210	H_0	H_1-	H_1-
230	H_1-	H_1-	H_1-
250	H_1-	H_1-	H_1-
270	H_1-	H_1-	H_0
290	H_0	H_1-	H_1-
300	H_1-	H_1-	H_1-

At Table 5 H_1- indicates that d has a negative value. According to the results, it can be said that H_1 is predominant. DE_1 converged before that $DE_{2,1}$ in 4 of the 6 tested power demands and before the PSO for all power demands. Furthermore, $DE_{2,1}$ converged faster than PSO for 5 of the 6 tested demands. Figure 9 shows the histograms for DE_1 vs $DE_{2,1}$ for the demands 210 and 230 MW, which means H_1- and H_0, respectively.

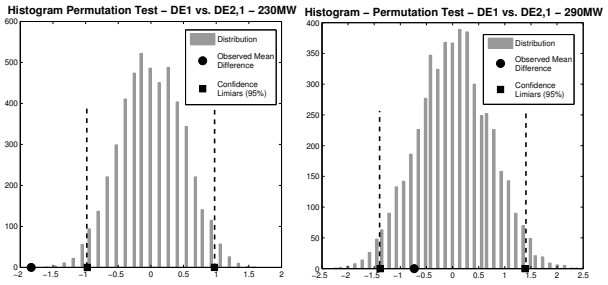

Figure 9: Permutation Results - Convergence

5. CONCLUSIONS

This paper presented a proposal for optimizing the unit commitment of hydroelectric generating units of a single plant. The proposed method is based on a Neural Network to model the discharge function and the novel movable partitions approach to meet the required power demand, which avoids an equality constraint in the model and requires one less dimension to represent it. Three evolutionary methods were tested to optimize the objective function modeled: DE/best/1/bin, a balanced version of DE and the PSO algorithm. All the three algorithms encountered similar best fitness values for all the demands tested and it was not possible to say if an algorithm is better than another. From the convergence test analysis, the algorithm DE/best/1/bin presented a higher performance for this problem comparing to the other two, in terms of number of generations required to reach the best solution. The balanced version of DE also manifested faster convergence when compared to PSO in most of cases.

6. REFERENCES

[1] T. O. A. Arce and S. Soares. Optimal dispatch of generating units of the itaipú hydroelectric plant. *IEEE Transactions on Power Systems*, 17:154–158, Feb 2002.

[2] N. Ampazis and S. Perantonis. Levenberg-marquadt algorithm with adaptive momentum for the efficient training of feedforward networks. In *IEEE International Joint Conference on Neural Networks*, volume 1, pages 126–131, Jul 2000.

[3] A. Conejo and N. J. Redondo. Short-termhydro-thermal coordination by relaxation lagrangian: Solution of the dual problem. *IEEE Trans on Power Systems*, 14(1):89–95, February 1999.

[4] S. Das and P. N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15:4–31, Feb 2011.

[5] J. L. D. David Kao, Alison Luo and A. Pang. Visualizing spatially varying distribution data. *Proceedings of the Sixth International Conference on Information Visualisation*, pages 219–225, Nov 2002.

[6] E. F. W. Eduardo G. Carrano and R. H. C. Takahashi. A multicriteria statistical based comparison methodology for evaluating evolutionary algorithms. *IEEE Transactions of Evolutionary Computation*, 15:848–870, Dec 2011.

[7] E. C. Finardi and E. L. da Silva. Unit commitment of single hydroelectric plant. *Electric Power Systems Research*, 75:116–123, May 2005.

[8] E. C. Finardi and E. L. da Silva. Solving the hydro unit commitment problem via dual decomposition and sequential quadratic programming. *IEEE Transactions on Power Systems*, 21(2):835–844, May 2006.

[9] S. Haykin. *Neural Networks - A Comprehensive Foundation*. Pearson Prentice Hall, 2nd edition, 1999.

[10] J. Kennedy and R. C. Eberhart. Particle swarm optimization. *Proceedings of the 1995 IEEE International Conference on Neural Networks*, 4:1942–1948, 1995.

[11] S. Liu and X. Li. Hydroelectric unit commitment by enhanced pso. In *IEEE E-Product E-Service and E-Entertainment*, pages 1–4, 2010.

[12] V. P. M.G Epitropakis and M. Vrahatis. Balancing the exploration and exploitation capabilities of the differential evolution algorithm. In *IEEE Congress on Evolutionary Computation*, pages 2686–2693, Jun 2008.

[13] D. S. Moore and G. P. McCabe. *Introduction to the Practice of Statistics*. Freeman, 5th edition, 2005.

[14] R. K. Pancholi and K. S. Swarup. Particle swarm optimization for security constrained economic dispatch. In *IEEE Intelligent Sensing and Information Processing*, pages 7–12, 2004.

[15] X. G. Q. Zhai and J. Cui. Unit commitment with identical units: Successive subproblem solving method based on lagrangian relaxation. *IEEE Trans on Power Systems*, 17(4):1250–1257, November 2002.

[16] R. Storn and K. V. Price. Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. *ICSI, USA Technical Report TR-95-012*, Mar 1995.

[17] T. Yalcinoz and M. Short. Neural networks approach for solving economic dispatch problem with transmission capacity constraints. *IEEE Transactions on Power Systems*, 13:307–313, May 1998.