Chapter-2
Chapter 2

PSO/Self-adaptive EP Based Hybridized Techniques For Solution of Economic Load Dispatch Problems

2.1 Introduction

Economic dispatch (ED) is an important optimization task in power system operation for allocating generation among the committed units such that the cost of production is minimized and at the same time constraints imposed are satisfied. Improvements in scheduling the unit outputs can lead to significant savings in the cost of producing the required energy. Conventional classical dispatch algorithms employing the lambda-iteration method, the base point and participation factors method, and the gradient method [1] [3] require the approximation of incremental cost curves with monotonically increasing or piece-wise linear curves. However, most of the modern units exhibit highly non-linear input-output characteristics because of valve point loadings, rate limits, prohibiting operating zones etc resulting in multiple local minima in the cost function.

Typically, the valve point effects, due to wire drawing, as each steam admission valve starting to open produce ripple like heat rate curve of the type shown in figure 2.1. Besides, a power generation system with combined-cycle unit serving
2.1. Introduction

Figure 2.1: Input-Output curve with five valve points. A, B, C, D and E show Operating points for admission valve.

For peak load comprises of series of single-cycle gas turbines in conjunction with some heat-recovery steam generators (HRSG) which lead to non-linearities in cost function.

Figure 2.2: Heat rate curve for combined cycle unit with three gas turbines

Figure 2.2 shows the net heat-rate curve of a combined cycle unit with three gas turbines. The other important factor that may introduce discontinuity in the cost function is the prohibited operating zone. Units may even develop such zones due to faults in the machines, boilers, feed pumps etc., leading to instabilities in certain ranges. Figure 2.3 shows the cost function with prohibited operating zones. The above considerations are inherently suggestive of highly non-convex input-output characteristics of modern units which may result in multiple local
2.1. Introduction

Prohibited operating zones

Power Output

Figure 2.3: Input-Output Characteristics of generators (units) with prohibited operating zones

minima in the cost function. Figure 2.4 shows a multi-modal objective function with number of local minima. Hence, optimization techniques that require convex characteristics may not be smart enough in solving these types of problems. So, their characteristics have to be approximated to linearize them to meet the requirements of the classical dispatch algorithms. However, such approximations may lead to huge loss of revenue over the time.

Figure 2.4: A multimodal objective function

Consideration of highly nonlinear characteristics of the units demand for solution techniques, which have no restrictions on the shape of the fuel cost curves. The classical gradient-based techniques fail in solving these types of problems.
2.2. Self-adaptive Evolutionary Programming

Unlike the conventional algorithms, though Dynamic Programming (DP) [1] imposes no restrictions on the nature of the cost curves and hence can solve the ED problems with inherently nonlinear and discontinuous cost curves but proves to suffer from intensive mathematical computations.

In this respect, stochastic random search techniques like evolutionary computational techniques are reported to have the capability of tackling very highly non-linear optimization problems. Evolutionary computational techniques like genetic algorithm (GA), evolutionary strategy (ES), evolutionary programming (EP), simulated annealing (SA), particle swarm optimization (PSO) and differential evolution (DE) are reported to be very efficient in solving highly non-convex optimization problems with no restriction on the shape of the objective functions. Although these heuristic methods do not always guarantee the globally optimal solution, they generally provide a fast and reasonable solution (sub-optimal near globally optimal). Their reported excellent records on benchmark pure mathematical problems have tempted us to apply them to highly non-convex economic load dispatch (ELD) problems in power systems.

2.2 Self-adaptive Evolutionary Programming

Self-adaptive EP technique has been widely used in recent years for highly non-linear optimization problems. The strategy parameters, also called step size, play a very significant role in determining the new population of the objective variables in the evolutionary process. In the widely used self-adaptation scheme of EP, this parameter, rather than being manually fixed, is evolved along with the objective variables.

Angeline [13] [14] defines this self-adaptation as an evolutionary computation, which evolves the values of the adaptive parameters. He suggested two methods to categorize adaptive evolutionary optimizations. The first method classifies self-adaptation schemes into "absolute" and "empirical" ones. An absolute update rule applies the predefined heuristic or statistical patterns from a set of generations or populations to modify the adaptive parameters. An empirical update rule uses specified mutation function to evolve the adaptive parameters during the evolutionary process. If the modified adaptive parameter in an offspring re-
2.2. Self-adaptive Evolutionary Programming

suits in higher fitness, it will survive to the next generation and proliferate in the population.

The second method classifies the adaptation schemes into three different levels, e.g., (i) population level, (ii) individual level, (iii) component level. The population-level adaptations have the effect on the whole population e.g. “1/5 success rule” that modifies the global mutation variance according to the ratio of successful mutations to the unsuccessful mutations. The individual-level adaptation parameters are associated with single individuals. The component-level adaptive parameters are only used to modify specified components during reproduction. In EP and ES component-level adaptive parameters have been extensively applied to various function optimizations. In the present work component-level adaptation with empirical learning rate has been considered for adaptation.

The standardized problem for an evolutionary computation is given a function \( \Gamma : \nu \rightarrow \mathbb{R} \) to find a structure, \( \nu_0 \), such that \( \Gamma \) returns a value at or near a desired extrema in its range. \( \Gamma \) is called the fitness function for the problem and represents the metric for evaluation and often the only specific information of the problem. Without loss of generality, the function is defined as, \( \Gamma : \nu \rightarrow \mathbb{R} \) which is the vector version of \( \Gamma \) that simply applies to each member of a vector of structures and returns the vector of fitness values. The vector of structures, \( \Gamma \), represents the evolving population for the problem. With these definitions, an evolutionary computation can be defined as an iteration of the following equation:

\[
\nu_{i+1} = \beta(\nu_i, \Gamma(\nu_i))
\]  

where, \( \beta : \nu \times \mathbb{R} \rightarrow \nu \) is the function that creates a new vector of structures, i.e., a new population, from the old. The parameters of \( \beta \) are determined by the style of evolutionary computation being used and include components that select population members to be parents for the next generation and the operators that manipulate them. A single iteration of equation (2.1) corresponds to a single generation in an evolutionary computation and is typically iterated until some predefined stopping criteria is met.

An adaptive evolutionary computation can be defined mathematically as follows:

\[
\nu_{i+1} = \beta_\delta(\nu_i, \Gamma(\nu_i), \bar{\kappa}_i)
\]  

(2.2)
where, $\beta : \mathbb{R} \times \mathbb{R} \times \delta \rightarrow \mathbb{R}$ is the adaptive evolutionary computation that takes on additional argument, $\delta$, often called the adaptive parameters or the strategy parameters. Strategy parameters are often numerical values of symbolic structures used by the reproduction operators (e.g. the frequency of crossover or severity of mutation). Ideally, strategy parameters are intended to take on values that supply a bias in the creation of offspring towards more fit solutions in the solution space. A second equation associated with an adaptive evolutionary computation defines the progression of values for the strategy parameters:

$$\delta_{i+1} = \Delta(\delta_i, \nu, \Gamma(\nu_i))$$

where, $\Delta : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ maps the current vector of strategy parameters into those to be used during the next reproduction using the current population and the fitness of each number as additional inputs. $\Delta$ is typically called the adaptive heuristic or the parameter update rule.

The ultimate efficiency of an adaptive evolutionary computation is directly dependent on the choice of parameters and the method used for their updation. While using an empirical update rule, the benefit of a particular modification is determined by its persistence in the face of natural dynamics of evolutionary method rather than its agreement with any empirically determined absolute heuristic. The natural dynamics of an evolutionary computation encourages the promotion of those structures that quickly lead to more fit individuals. If the modification made to a strategy parameter in a particular individual is comparatively beneficial (or neutral) to offspring development, it will be persistent in the population along with the lineage of the individual. If, on the other hand, it is detrimental, then the individual’s offspring will eventually be supplanted by those with strategy parameters more beneficial in creating increasingly fit offspring.

A number of self-adaptation strategies have been proposed for EP. The basic self-adaptation strategies are:

1. **Gaussian Self-Adaptation**

   The strategy parameters are adapted using the relation
   $$S'(i) = S(i) + S(i)N_i(0, 1)$$

   any $S'(i)$ that go negative are set to a positive small value $\epsilon$. $S$ is strategy
2.3. PSO Technique

parameter and $N_i$ is normally distributed random number generated anew for each $i$.

2. Lognormal self-adaptation

The strategy parameters are adapted using the relation

$$S'(i) = S(i) \exp(\tau N_i(0, 1) + \tau' N(0, 1))$$

where $\tau$ and $\tau'$ are called empirical leaning rates. In this work lognormal self adaptation strategy is used.

2.3 PSO Technique

Kennedy and Eberhart [10] were the first to propose PSO technique as one of the modern heuristic algorithms. It was developed under emulation of a simplified social system, and has been found to be very efficient in solving continuous nonlinear optimization problems. PSO as an optimization tool provides a population-based search procedure in which individuals, called particles, change their positions (states) with time. In a PSO system, particles fly around in a multidimensional search space. During flight, each particle adjusts its position according to its own experience, and the experience of neighboring particles, making use of the best position encountered by itself and its neighbors. The PSO technique can generate high-quality solution within shorter calculation time and more stable convergence characteristics than other stochastic methods [11]. Many researches are still in progress for proving its potential in solving complex power system problems.

2.3.1 Creation of Offsprings

Let $P_{i,j}$ and $v_j$ denote the coordinates and the corresponding flight speed of the particle $(i, j)$ respectively in a search space. The particle positions are manipulated according to the following equation (2.4):

$$v_{i,j} = w \cdot v_{i,j} + c_1 \cdot rand_1 \cdot (P_{gbest,j} - P_{i,j}) + c_2 \cdot rand_2 \cdot (P_{pbest_{i,j}} - P_{i,j}) \quad (2.4)$$

$$P'_{i,j} = P_{i,j} + v_{i,j} \quad (2.5)$$
where \( w \) = inertia weight factor, in our work \( w = 0 \)
\( c_1, c_2 \) = acceleration constant in general.
\( rand_1 \) = random number in the range \([0,1]\)
\( rand_2 \) = random number in the range \([0,1]\)
\( P_g\text{best} \) = the best particle among all individuals in the population.
\( P_p\text{best} \) = the best history position of particle \( P_{i,j} \).

2.4 A Brief literature survey on previous works

Prior to 1930, various methods were in practice such as (a) base point method, where the most efficient unit is loaded to its maximum capacity; then the second most efficient unit is loaded etc; (b) “Best point loading”, where the units are successively loaded to their lowest heat rate point, beginning with the most efficient unit and working down to the least efficient unit etc. It was recognized as early as in 1930 that the incremental method, later known as the equal incremental method, yielded the most economic results. The theoretical work on optimal dispatch later led to the development of analog computers for properly solving the coordination equations in a dispatching environment. A transmission loss penalty factor computer was developed in 1954 and was used by AEP in conjunction with an incremental loading slide rule for producing daily generation schedules in a load dispatching office. An electronic differential analyzer was developed for use in economic scheduling for off-line or on-line use by 1955. The use of digital computers for obtaining loading schedules was investigated in 1955 and this had revolutionized the developments in optimization. A breakthrough in the mathematical formulation of the ELD problem was proposed by Carpentier [15] whose general formulation of the ELD problem was based upon the Kuhn-Tucker theorem of non-linear programming. Happ [16] in his paper concluded that the classical method based on coordination equations, is as good as any other rigorous method for the ELD problems.

The dynamic programming (DP) method as proposed by Wood and Wollenberg [1] [3] provides a general solution to the ED problem and imposes no restrictions on the characteristics of the generating units. But, it suffers from the
curse of dimensionality. Both the storage requirements and the execution time of the DP algorithm increases by leaps and bounds with increase in system size and accuracy requirement. Chowdhury and Rahman [17] have presented a review of the advances in ED up to 1990. In the paper, they have also proposed some advanced techniques for solving the ELD problem.

Stochastic search algorithms like genetic algorithm (GA) [4][18-25], evolutionary strategy (ES) [26], evolutionary programming (EP) [8-9][26-36], particle swarm optimization (PSO) [10-12][37-39] and simulated annealing (SA) [4] [5], which have no restrictions on the shape of the objective functions, may prove to be very efficient in solving highly nonlinear ELD problems. Although these heuristic methods do not always guarantee the globally optimal solution, they generally provide a fast and reasonable solution (sub-optimal near globally optimal). The main drawback of SA is the difficulty in determining an appropriate annealing schedule, otherwise the solution achieved may still be a locally optimal one. Most recent trends for research, therefore, have been directed towards application of efficient and near optimal evolutionary algorithms i.e. GA, ES and EP. These evolutionary algorithms (EAs) are search algorithms based on the simulated evolutionary process of natural selection and genetics. EAs are more flexible and robust than conventional calculus based methods. Due to its high potential for global optimization, GA has received great attention in solving ED problems. Walters and Sheble [18] reported a GA model that employed units' output as the encoded parameter of chromosome to solve an ED problem with valve-point discontinuities. To enhance the performance of GA, Yalcinoz et al. [24] have proposed the real-coded representation scheme, arithmetic crossover, mutation, and elitism in the GA to solve more efficiently the ED problem which can obtain a high-quality solution with less computation time. Though the GA methods have been employed successfully to solve complex optimization problems, recent research has identified some deficiencies in GA performance. This degradation in efficiency is apparent especially with epistatic objective functions. EP differs from GA in two aspects: EP uses the control parameters (real values), but not their codings as in GAs; the operators in EP are mutation and competition (selection), but not reproduction, mutation and crossover as in GAs. Hence considerable computation time may be
saved in EP. It has been reported [23] that EP outperforms GAs. EP has seen a lot of developments for over four decades in terms of faster convergence rate and solution quality. Mutation in EP is often implemented by adding a random number or a vector from a certain distribution [e.g., a Gaussian distribution in the case of classical EP (CEP)] to a parent. The degree of variation of the Gaussian mutation is controlled by its standard deviation, which is also known as a "strategy parameter" in evolutionary search. In the self-adaptation scheme of EP, this parameter is not prefixed; rather, it is evolved along with the objective variables. Experiments with self-adaptive EP have indicated efficient convergence to quality solutions [8][26-30][32][35].

Very recent addition to these types of search techniques is PSO [10-12]. In comparison to population based evolutionary algorithms, PSO is computationally inexpensive in terms of memory and speed. Constructive cooperation rather than survival of the fittest is the fundamental principle of PSO technique. Therefore, the optimal solution is reached by cooperation of all of the individuals within the population.

In pursuit of enhancing the convergence rate and efficiency of stochastic algorithms, it is felt to combine self-adaptive EP and PSO techniques. While hybridizing the two techniques, naturally the following questions regarding the nature of hybridization arise:

(i) Should the swarm direction of PSO be embedded into mutation (strategy parameter) of self-adaptive CEP, or

(ii) Should the strategy parameter of CEP be embedded into swarm directions of PSO; or

(iii) Should there be two stage hybridization wherein new solutions are created first with CEP and better individuals are chosen for finding new swarm directions of PSO in the second stage.

In view of the above, the main objectives of the present chapter are:

(1) To develop a program based on floating point GA and study its performance in solving the non-convex ELD problem.
(2) To develop a program based on self-adaptive CEP technique and study its performance in solving the same problem as in 1.

(3) To develop a program based on PSO technique and study its performance in solving the same problem as in 1.

(4) To develop programs based on three types of PSO and CEP hybridization techniques and study their performance as compared to those developed under steps 1, 2 and 3 above.

(5) To develop a program based on hybridization of PSO and improved fast EP (IFEP) using the best of three hybridization techniques under step 4 and study its performance as compared to that with the best developed under step 4.

2.5 ELD Problem Formulation

The ELD problem is an important optimization task in power system operation for allocating amount of power to be generated among the committed units. In ELD problem, the objective function is to minimize the total cost of production of all generators as follows,

\[
\min(TFC) = \sum_{j=1}^{n} F(P_j)
\]

where \( FC \) is the Total Fuel Cost of all the generators, \( F(P_j) \) is the fuel cost of the \( j \)th unit and is a function of the Power generated by the \( j \)th unit, \( P_j \).

The problem at hand is to minimize the equation (2.6) satisfying the equality and in-equality constraints as detailed below:

1. Power Balance Constraints:

\[
\sum_{j=1}^{n} P_j - P_{\text{loss}} - P_D = 0
\]

where \( P_D \) is the system load demand and \( P_{\text{loss}} \) is the transmission loss

2. Generation Capacity Constraints:

\[
P_{j,\text{min}} \leq P_j \leq P_{j,\text{max}} \quad \text{for} \quad j = 1, 2, ..., n
\]
where $P_{\text{min}}$ and $P_{\text{max}}$ are the minimum and maximum power outputs of the $j$th unit. In other words, a generator has to be scheduled such that the power output is within these bounds.

The fuel cost function referred in (2.6), considering valve points loadings of the generating units are given as

$$F(P_j) = a_j + b_j P_j + c_j P_j^2 + \left| e_j \sin(f_j(P_{\text{min}} - P_j)) \right|$$

(2.9)

where $a_j$, $b_j$ and $c_j$ are the fuel cost coefficients of the $j$th unit and $e_j$ and $f_j$ are the fuel cost coefficients of the same with valve points effects.

The generating units with multi-valve steam turbines exhibit a greater variation in the fuel cost functions. The valve-points introduce ripples in the heat rate curves.

### 2.6 Self-Adaptive CEP Based Economic Dispatch

#### 2.6.1 Initialisation

Let $p_i = [P_1, P_2, \ldots, P_j, \ldots, P_n]$ be a trial vector denoting the $i$th individual of a population to be evolved. The elements, $P_j$s ($j = 1, 2, \ldots, N$), of the trial vector $p_i$ are the real power outputs (objective variables) of the committed $n$ generating units which are subjected to their respective capacity constraints as in (2.8). An initial population of $N$ individuals is taken as a pair of real valued vectors $(p_i, s_i), \forall_i, i \in \{1, 2, \ldots N\}$. The components $P_j$s are determined as below

$$P_j \sim U(P_{\text{min}}, P_{\text{max}}) \text{ for } j = 1, 2, \ldots, n, j \neq d$$

where, $U(P_{\text{min}}, P_{\text{max}})$ denotes a uniform random variable ranging over $(P_{\text{min}}, P_{\text{max}})$ and $s_i$'s are standard deviations for Gaussian mutation (also known as strategy parameters in the self-adaptive evolutionary algorithms). Here, $s_i$ is initialized to a suitable value after tuning. $d$ is the index of the dependent generating unit as discussed below.

To meet the exact load demand in (2.7), a dependent unit is arbitrarily selected from among the committed $n$ units. The power output of the dependent unit, $P_d$ can be calculated by

$$P_d = P_D + P_{\text{loss}} - \sum_{j=1, j \neq d}^{n} P_j$$

(2.10)
In our work the power loss is not considered. However, it may be calculated by an iterative algorithm or by directly using B-loss matrix of the power system.

### 2.6.2 Creation of Offspring

Self-adaptive CEP based ED with adaptation of strategy parameter using empirical learning rate is proposed by Bäck and Schwefel [26]. The mutation scheme as explained below has been employed for creation of offspring:

**By Gaussian mutation (in CEP)**

An offspring vector is created from each parent by

\[
S'_{ij} = S_{ij} \exp\{\tau'N(0,1) + \tau N_j(0,1)\} \\
P'_{ij} = P_{ij} + S'_{ij} N_j(0,1)
\]

where \(S_{ij}, S'_{ij}, P_{ij}\) and \(P'_{ij}\) denote the \(j\)th component of vectors \(s_i, s'_i, p_i\) and \(p'_i\) respectively. \(N(0,1)\) denotes a normally distributed random number with mean 0 and standard deviation 1. And \(N_j(0,1)\) denotes the random number generated anew for each value of \(j\). The factors \(\tau\) and \(\tau'\) are called learning rates and commonly set to \((\sqrt{\frac{2}{\pi n^2}})^{-1}\) and \((\sqrt{\frac{2}{\pi n}})^{-1}\) respectively; where \(n\) is the number of objective variables.

### 2.6.3 Fitness Evaluation

Evaluate the fitness scores for each individual \((p_i, s_i)\), \(\forall i, i \in \{1, 2, \ldots N\}\), of the population based on the fitness function.

The fitness function, which is the sum of production cost and penalty for constraint violation, can be calculated for each individual of the parent population as

\[
FIT_i = TFC_i + \sum_{z=1}^{N_c} PF_z 
\]

where \(PF_z = \lambda_z |VIOL_z|^2\) and \(VIOL_z\) is the violation of constraint \(z\), \(\lambda_z\) is the penalty multiplier and \(N_c\) is the number of constraints. In the present case, the limits of power outputs of the dependent unit are the constraints.
2.6.4 Competition and Selection

Each individual in the combined population of $N$ parent trial vectors and their corresponding $N$ offsprings has to compete with $R$ number of individuals (Tournament size), randomly chosen from the combined population, to have a chance to survive to the next generation. The value of $R$ may be at best equal to the population size of the parent population. A weight value $\text{win}_r$ is assigned to the $i^{th}$ individual on competition with each of the random chromosome, $r$, as per the following equation:

$$\text{win}_r = \begin{cases} 1 & \text{if } u_1 > \frac{FIT_i}{FIT_r + FIT_i} \\ 0 & \text{otherwise} \end{cases}$$

Now the sum of all the wins of $i^{th}$ individual, over randomly selected ones are added to get a score of the $i^{th}$ individual as follows:

$$w_i = \sum_{r=1}^{R} \text{win}_r$$

(2.14)

where $R$ is the number of competitors; $FIT_i$ is the fitness value of $i^{th}$ individual, $p_i$; $FIT_r$ is the fitness value of selected competitor from $2N$ trial solutions based on $r = [2Nu_2 + 1]$; $u_1$ and $u_2$ are uniform random numbers ranging over $[0, 1]$. 

When all individuals obtain their competition scores, they will be ranked in descending order of their corresponding score, $w_i$. The first $N$ individuals are selected and transcribed along with their corresponding fitness values $FIT_i$ to be the parents in the next generation.

2.6.5 Stopping Rule

The iterative procedure of generating new trials by selecting those with minimum objective function values from the competing pool is terminated when there is no significant improvement in the solution. It can also be terminated when a given maximum number of generations (iterations) are reached. In the present work the latter method is employed.
2.7 Hybrid PSO/Self Adaptive CEP Based ED

To exploit the faster convergence capabilities of PSO technique and better exploration features of EP, it is felt to develop solution techniques through hybridization of self-adaptive EP and PSO techniques. Different variants of hybridization is implemented in the way new offsprings are created.

2.7.1 Creation of Offsprings

The hybridization is carried out and offsprings can be created in three ways, which are described as below:

(i) By PSO-CEP technique

The offspring under this technique can be created by embedding swarm directions into CEP as below:

\[ v_{i,j} = c_1 \cdot rand_1 \cdot (P_{best,j} - P_{i,j}) \]
\[ + c_2 \cdot rand_2 \cdot (P_{pbest_{i,j}} - P_{i,j}) \]  \hspace{1cm} (2.15)

\[ P'_{i,j} = P_{i,j} + v_{i,j} \]  \hspace{1cm} (2.16)

\[ S'_{i,j} = S_{i,j} \exp \{\tau'N(0,1) + \tau N_j(0,1)\} \]  \hspace{1cm} (2.17)

\[ P''_{i,j} = P'_{i,j} + S'_{i,j}N_j(0,1) \]  \hspace{1cm} (2.18)

(ii) By CEP-PSO technique

An offspring is created by embedding strategy parameter of CEP into the swarm direction of PSO as follows:

\[ S'_{i,j} = S_{i,j} \exp \{\tau'N(0,1) + \tau N_j(0,1)\} \]  \hspace{1cm} (2.19)

\[ P'_{i,j} = P_{i,j} + S'_{i,j}N_j(0,1) \]  \hspace{1cm} (2.20)

\[ v_{i,j} = c_1 \cdot rand_1 \cdot (P_{best,j} - P_{i,j}) \]
\[ + c_2 \cdot rand_2 \cdot (P_{pbest_{i,j}} - P_{i,j}) \]  \hspace{1cm} (2.21)

\[ P''_{i,j} = P'_{i,j} + v_{i,j} \]  \hspace{1cm} (2.22)
2.8. Algorithm for Hybrid PSO/CEP Technique for the test case

(iii) By CEP-PSO' technique

Creation of offspring is carried out in two steps as follows:

First new population is created by self-adaptive CEP

\[ S'_{i,j} = S_{i,j} \exp\{\tau'N(0,1) + \tau N_j(0,1)\} \] (2.23)

\[ P'_i = P_{i,j} + S'_{i,j}N_j(0,1) \] (2.24)

Next, the new population is compared with their parent population and better individuals are chosen as parents \( P^N \). Now the population is evolved by PSO technique as below:

\[ v_{i,j} = c_1 \cdot \text{rand}_1 \cdot (P_{\text{gbest}_j} - P'_{i,j}) + c_2 \cdot \text{rand}_2 \cdot (P_{\text{pbest}_{i,j}} - P'_{i,j}) \] (2.25)

\[ P''_{i,j} = P^N_{i,j} + v_{i,j} \] (2.26)

Evaluation, competition and selection functions are the same as in section 2.6.

2.8 Algorithm for Hybrid PSO/CEP Technique for the test case

• Step-1: The problem variables to be determined are represented as a \( n \)-dimensional trial vector, where each vector is an individual of the population to be evolved.

• Step-2: An initial population of parent vectors, \( Q_i \), for \( i = 1, 2, \ldots, N \), is selected at random from the feasible range in each dimension. The distribution of these initial parent vectors is uniform.

• Step-3: Calculate the fitness value of each individual in the population using the evaluation function given by (2.13).

• Step-4: An offspring \( Q'_i \) is generated from each parent by using Eqs. (2.15) to (2.18), Eqs. (2.19) to (2.22) or Eqs. (2.23) to (2.26) depending on the type of hybridization required viz. PSO-CEP, CEP-PSO or CEP-PSO' respectively.
2.9. Numerical Tests

- Step-5: Fitness function, \( F_{IT} \), is evaluated for each individual of both parent and child populations as per eq. (2.13). Compare each individual’s fitness value with that of \( P_{pbest} \). The best fitness value among the \( P_{pbests} \) is denoted as \( P_{gbest} \).

- Step-6: A competitor is chosen randomly from the combined population of \( 2N \) trial solutions (\( N \) parents and \( N \) offspring) and stochastic competition is performed based on the value of fitness function where each individual in the competing pool compete against other members for survival as shown in equation (2.14).

- Step-7: After the competition is over, the \( 2N \) trial solutions in the competing pool are sorted in descending order of their scores. Thereafter the first \( N \) trial solutions are selected as the new parent vectors for the next generation.

- Step-8: If current generation number is greater than or equal to the maximum generation, print the result and stop; otherwise go to step 3.

2.9 Numerical Tests

The performance of the proposed algorithms are experimented on test systems of 15 and 40 units separately, with valve point loadings in both the cases. The input data for 15 and 40 units are given in table 2.1 and 2.3 respectively. In these tables, \( Unit_j \) is the \( j \)th generator number, \( P_{min} \) and \( P_{max} \) are min and max power output of \( j \)th unit expressed in MW (Mega watts). The performance of the proposed EP algorithms are verified on a test system which has been adapted from [22] with modifications to incorporate the effects of valve point loadings.

General parameters used in the algorithms:

Number of generating units = \( n = 15 \) and 40 (Two cases are considered)
Population Size = \( N = 60 \) Maximum
Iterations = 500 in case of 15 unit and 400 in case of 40 unit.
PSO Parameters used in the algorithms:
Maximum Velocity, $V_{\text{max}} = \frac{1}{2}P_{\text{max}}, \ j = 1,2, \ldots , n$
Minimum Velocity, $V_{\text{min}} = \frac{1}{2}P_{\text{min}}, \ j = 1,2, \ldots , n$
Inertia Weight, $w = 0.0$
Constants, $c1 = c2 = 2.0$

Features of floating point GA (GAF) used:
(i) Heuristic crossover
(ii) Non-uniform mutation
(iii) Normalized geometric select function.

The GA optimization toolbox GAOT in Matlab proposed by C. R. Houck, J. Joines, and M. Kay [40] is used after minor modification for solving the problem. The binary GA and floating point GA (GAF) with different mutation and crossover functions available in the toolbox were tried on this problem. The GAF with heuristic crossover and non-uniform mutation appeared to perform better than GAFs with other type of crossover and mutation functions and hence considered here for comparison. Non-uniform mutation [40] changes one of the parameters of the parent based on a non-uniform probability distribution. Non-uniform mutation randomly selects a variable $i$ and sets it equal to a non-uniform random number:

$$x'_i = \begin{cases} 
  x_i + (b_i - x_i)f(G) & \text{if } r_1 < 0.5 \\
  x_i - (x_i + a_i)f(G) & \text{if } r_2 \geq 0.5 \\
  x_i & \text{otherwise}
\end{cases}$$

where $r_1, r_2$ = a uniform random number (0,1)
$c$ = shape parameter
$i$ = $i^{th}$ individual
$x_i$ and $x'_i$ = old/new value of $i^{th}$ variable respectively
$a_i, b_i$ = Lower and Upper bounds of $x_i$ respectively
$G$ = Current generation index
$G_{\text{max}}$ = Max. number of generations
$f(G) = (r_2(1 - \frac{G}{G_{\text{max}}}))^c$
Heuristic crossover [40] performs extrapolation along the line formed by the two parents outward in the direction of the better fit parent. It utilizes the fitness information. New individuals $X'$ and $Y'$ are created using Eqs. (2.27) and (2.28) respectively where $r = U(0,1)$ and the parent $X$ is better than $Y$ in terms of fitness.

$X' = X + r(X - Y)$ \hspace{1cm} (2.27)

$Y' = X$ \hspace{1cm} (2.28)

The convergence characteristics of the algorithms are shown in fig. 2.5 and 2.6 for 15 unit test case. Similarly, fig. 2.9 and 2.10 show the convergence characteristics of the algorithms for 40 unit case. Investigation of these four convergence characteristics reveals that all the hybrid algorithms posses better convergence rate than any of the individual algorithms (viz. GAF, SA-CEP, PSO). Algorithms CEP-PSO and PSO-CEP do not appear to have significant improvements over the other, though the former has marginally better results.

To investigate the effects of initial trial solutions, all the algorithms were run with 50 different initial trial solutions and the results are reported in fig. 2.7, 2.8 and in table 2.2 for 15 unit test case. Similarly, the case of 40 unit has been shown in fig. 2.11, 2.12 and in table 2.4.

In these results, the average cost is the prime indicator of competence of an algorithm in finding better quality solutions. It is observed that PSO is more competent than GAF and SA-CEP but hybrid algorithms are observed to possess better capability than any of the individual algorithms. Amongst the hybrid algorithms, CEP-PSO$'$ appears to be the most efficient in terms of faster convergence rate and quality of solution, which makes it very efficient in finding the global optimum.

Table 2.5 shows the average solution times for different algorithms for 40 unit test case. The solution times are the times taken by different algorithms in achieving solutions of nearly similar quality. It is worth to be mentioned herein that success rate of other algorithms in achieving solution qualities similar to that of CEP-PSO$'$ is significantly less. It is evident that CEP-PSO$'$ is the most efficient in finding better quality solutions.
2.9. Numerical Tests

### Table 2.1: Units' data for the (15 units) test case with load 2650 MW (considering valve point loadings)

<table>
<thead>
<tr>
<th>Unit_j</th>
<th>Output Limits</th>
<th>Fuel coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>1</td>
<td>15</td>
<td>55</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>455</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>130</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>130</td>
</tr>
<tr>
<td>5</td>
<td>150</td>
<td>470</td>
</tr>
<tr>
<td>6</td>
<td>135</td>
<td>460</td>
</tr>
<tr>
<td>7</td>
<td>135</td>
<td>465</td>
</tr>
<tr>
<td>8</td>
<td>60</td>
<td>300</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
<td>162</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>160</td>
</tr>
<tr>
<td>11</td>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>12</td>
<td>20</td>
<td>80</td>
</tr>
<tr>
<td>13</td>
<td>25</td>
<td>85</td>
</tr>
<tr>
<td>14</td>
<td>15</td>
<td>55</td>
</tr>
<tr>
<td>15</td>
<td>150</td>
<td>455</td>
</tr>
</tbody>
</table>

### Table 2.2: Statistical test results of 50 runs with different initial solutions (with cost curves considering valve point loading effect) for 15 units test case

<table>
<thead>
<tr>
<th>Method</th>
<th>Average</th>
<th>Maximum</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAF</td>
<td>34013.56</td>
<td>34601.26</td>
<td>33343.80</td>
</tr>
<tr>
<td>SA-CEP</td>
<td>33601.65</td>
<td>33926.78</td>
<td>33228.77</td>
</tr>
<tr>
<td>PSO</td>
<td>33359.58</td>
<td>33873.03</td>
<td>33045.77</td>
</tr>
<tr>
<td>PSO-CEP</td>
<td>33434.76</td>
<td>33777.76</td>
<td>33137.02</td>
</tr>
<tr>
<td>CEP-PSO</td>
<td>33441.09</td>
<td>33815.56</td>
<td>33066.23</td>
</tr>
<tr>
<td>CEP-PSO'</td>
<td>33074.36</td>
<td>33572.02</td>
<td>32822.35</td>
</tr>
</tbody>
</table>

Table 2.1: Units' data for the (15 units) test case with load 2650 MW (considering valve point loadings)

Table 2.2: Statistical test results of 50 runs with different initial solutions (with cost curves considering valve point loading effect) for 15 units test case
2.9. Numerical Tests

Figure 2.5: The convergence nature of GAF, SA-CEP, PSO and PSO-CEP algorithms for 15 units test case
Figure 2.6: The convergence nature of PSO-CEP, CEP-PSO and CEP-PSO' algorithms for 15 units test case
Figure 2.7: Cost distribution of optimal solutions with 50 different initial trials using GAF, SA-CEP and PSO methods for 15 units test case.
Figure 2.8: Cost distribution of optimal solutions with 50 different initial trials using PSO-CEP, CEP-PSO and CEP-PSO’ methods for 15 units test case.
### Table 2.3: Units' data for the (40 units) test case with load 10500 MW (considering valve point loadings)

<table>
<thead>
<tr>
<th>Unit</th>
<th>Output Limits</th>
<th>Fuel coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
</tr>
<tr>
<td>1</td>
<td>36</td>
<td>114</td>
</tr>
<tr>
<td>2</td>
<td>36</td>
<td>114</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>120</td>
</tr>
<tr>
<td>4</td>
<td>80</td>
<td>190</td>
</tr>
<tr>
<td>5</td>
<td>47</td>
<td>97</td>
</tr>
<tr>
<td>6</td>
<td>68</td>
<td>140</td>
</tr>
<tr>
<td>7</td>
<td>110</td>
<td>300</td>
</tr>
<tr>
<td>8</td>
<td>135</td>
<td>300</td>
</tr>
<tr>
<td>9</td>
<td>135</td>
<td>300</td>
</tr>
<tr>
<td>10</td>
<td>130</td>
<td>300</td>
</tr>
<tr>
<td>11</td>
<td>94</td>
<td>375</td>
</tr>
<tr>
<td>12</td>
<td>94</td>
<td>375</td>
</tr>
<tr>
<td>13</td>
<td>125</td>
<td>500</td>
</tr>
<tr>
<td>14</td>
<td>125</td>
<td>500</td>
</tr>
<tr>
<td>15</td>
<td>125</td>
<td>500</td>
</tr>
<tr>
<td>16</td>
<td>125</td>
<td>500</td>
</tr>
<tr>
<td>17</td>
<td>220</td>
<td>500</td>
</tr>
<tr>
<td>18</td>
<td>220</td>
<td>500</td>
</tr>
<tr>
<td>19</td>
<td>242</td>
<td>550</td>
</tr>
<tr>
<td>20</td>
<td>242</td>
<td>550</td>
</tr>
<tr>
<td>21</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>22</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>23</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>24</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>25</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>26</td>
<td>254</td>
<td>550</td>
</tr>
<tr>
<td>27</td>
<td>10</td>
<td>150</td>
</tr>
<tr>
<td>28</td>
<td>10</td>
<td>150</td>
</tr>
<tr>
<td>29</td>
<td>10</td>
<td>150</td>
</tr>
<tr>
<td>30</td>
<td>47</td>
<td>97</td>
</tr>
<tr>
<td>31</td>
<td>60</td>
<td>190</td>
</tr>
<tr>
<td>32</td>
<td>60</td>
<td>190</td>
</tr>
<tr>
<td>33</td>
<td>60</td>
<td>190</td>
</tr>
<tr>
<td>34</td>
<td>90</td>
<td>200</td>
</tr>
<tr>
<td>35</td>
<td>90</td>
<td>200</td>
</tr>
<tr>
<td>36</td>
<td>90</td>
<td>200</td>
</tr>
<tr>
<td>37</td>
<td>25</td>
<td>110</td>
</tr>
<tr>
<td>38</td>
<td>25</td>
<td>110</td>
</tr>
<tr>
<td>39</td>
<td>25</td>
<td>110</td>
</tr>
<tr>
<td>40</td>
<td>242</td>
<td>550</td>
</tr>
</tbody>
</table>
Figure 2.9: The convergence nature of GAF, SA-CEP, PSO and PSO-CEP algorithms for 40 units test case.
Figure 2.10: The convergence nature of PSO-CEP, CEP-PSO and CEP-PSO' algorithms for 40 units test case
### Table 2.4: Statistical test results of 50 runs with different initial solutions (with cost curves considering valve point loading effect) for 40 units test case

<table>
<thead>
<tr>
<th>Method</th>
<th>Cost in Rs.</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>Maximum</td>
<td>Minimum</td>
</tr>
<tr>
<td>GAF</td>
<td>134128.78</td>
<td>138432.17</td>
<td>130521.54</td>
</tr>
<tr>
<td>SA-CEP</td>
<td>125942.35</td>
<td>127212.12</td>
<td>124979.31</td>
</tr>
<tr>
<td>PSO</td>
<td>125237.08</td>
<td>127300.11</td>
<td>123368.78</td>
</tr>
<tr>
<td>PSO-CEP</td>
<td>124085.37</td>
<td>125210.80</td>
<td>123278.09</td>
</tr>
<tr>
<td>CEP-PSO</td>
<td>124022.33</td>
<td>125741.11</td>
<td>123198.32</td>
</tr>
<tr>
<td>CEP-PSO'</td>
<td>123120.03</td>
<td>128522.14</td>
<td>122281.99</td>
</tr>
</tbody>
</table>

### Table 2.5: Average Solution times of different algorithms for 40 units test case

<table>
<thead>
<tr>
<th>Method</th>
<th>Average solution time (in Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAF</td>
<td>716.35</td>
</tr>
<tr>
<td>SA-CEP</td>
<td>607.90</td>
</tr>
<tr>
<td>PSO</td>
<td>465.11</td>
</tr>
<tr>
<td>PSO-CEP</td>
<td>405.70</td>
</tr>
<tr>
<td>CEP-PSO</td>
<td>392.18</td>
</tr>
<tr>
<td>CEP-PSO'</td>
<td>178.94</td>
</tr>
</tbody>
</table>
Figure 2.11: Cost distribution of optimal solutions with 50 different initial trials using GAF, SA-CEP and PSO methods for 40 units test case.
2.9. Numerical Tests

Figure 2.12: Cost distribution of optimal solutions with 50 different initial trials using PSO-CEP, CEP-PSO and CEP-PSO' methods for 40 units test case
2.10 Hybridization of PSO and improved fast EP (IFEP)

Out of all the EPs improved fast EP (IFEP) [8] is reported to have outperformed all other methods. Hence, in the pursuit of enhancing the convergence rate and efficiency of stochastic algorithms further, it is felt to combine self-adaptive improved EP (IFEP) and PSO techniques. It has been observed in the previous part of this chapter that the algorithm with mutation technique using two stage hybridization in between CEP and PSO (CEP-PSO') is superior to the other two hybridizations. Hence, it is felt to have two stage hybridization between IFEP and PSO to develop an algorithm with further enhanced convergence rate and efficiency. In view of the above, the further targeted objectives are:

(i) To develop a program based on two stage hybridization between CEP and PSO and study its performance in solving the non-convex ELD problem;

(ii) To develop a program based on two stage hybridization between IFEP and PSO and compare its performance with that developed in step (i) in solving the same above problem.

2.10.1 Creation of Offspring

The mutation scheme as explained below has been employed for creation of offspring under the new hybridization technique.

New solutions (offsprings) in IFEP technique are created using better of two offsprings generated from each parent, one by Gaussian mutation eq. (2.30) and the other by Cauchy mutation eq. (2.31), as below:

\[
S'_{i,j} = S_{i,j} \exp\{\tau'N(0,1) + \tau N_j(0,1)\} \tag{2.29}
\]

\[
P'_{i,j} = P_{i,j} + S'_{i,j} N_j(0,1) \tag{2.30}
\]

\[
P''_{i,j} = P_{i,j} + S'_{i,j} C_j(0,1) \tag{2.31}
\]

where \(S_{ij}, S'_{ij}, P_{ij}, P'_{ij}\) and \(P''_{ij}\) denote the \(j\)th component of vectors \(s_i, s'_i, p_i, p'_i\) and \(p''_i\) respectively. \(N(0,1)\) denotes a normally distributed random number with mean 0 and standard deviation 1. And \(N_j(0,1)\) denotes the random number generated.
2.11. Numerical Test on hybrid PSO and IFEP Based Technique

 anew for each value of $j$ and $C_j(0, 1)$ denotes the Cauchy random variable with scale parameter $t = 1$ and generated anew for each value of $j$. The factors $\tau$ and $\tau'$ are called learning rates and commonly set to $(\sqrt{2\sqrt{n}})^{-1}$ and $(\sqrt{2n})^{-1}$ respectively; where $n$ is the number of objective variables.

By Hybridized Mutation for IFEP-PSO' Technique

Creation of offspring under IFEP-PSO' is carried out in two steps as follows: The new offsprings $P_{ij}'$ and $P_{ij}''$ are created first by self-adaptive IFEP using equations (2.29) to (2.31). The objective function values of both the offsprings are evaluated. The newly created offsprings are compared with their parent and best of the three individuals is chosen. This method is applied to all the individuals of parent population and the new population $P^C$ is obtained. Now this population is evolved by PSO technique for next generation as below:

$$v_{i,j} = c_1 \times \text{rand}_1 \times (P_{gbest,j} - P_{i,j}^C)$$
$$+ c_2 \times \text{rand}_2 \times (P_{pbest,i,j} - P_{i,j}^C) \quad (2.32)$$

$$P_{ij}^C = P_{ij}^C + v_{i,j} \quad (2.33)$$

Evaluation, competition and selection functions are the same as in section 2.6.

2.11 Numerical Test on hybrid PSO and IFEP Based Technique

The performance of the proposed algorithms are experimented on a test system of 15 units with valve-point loadings. The input data for 15 units are given in table 2.1.

General parameters used in the algorithms:

- Number of generating units = $n = 15$
- Population Size = $N = 60$ Maximum
- Iterations = 500
2.11. Numerical Test on hybrid PSO and IFEP Based Technique

<table>
<thead>
<tr>
<th>Method</th>
<th>Cost in Rs.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
</tr>
<tr>
<td>CEP-PSO'</td>
<td>33073.00</td>
</tr>
<tr>
<td>IFEP-PSO'</td>
<td>33008.00</td>
</tr>
</tbody>
</table>

Table 2.6: Statistical test results of 50 runs with different initial solutions (with cost curves considering valve point loading effect) for 15 units test case.

<table>
<thead>
<tr>
<th>Method</th>
<th>Average solution time (in Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEP-PSO'</td>
<td>4.918</td>
</tr>
<tr>
<td>IFEP-PSO'</td>
<td>8.938</td>
</tr>
</tbody>
</table>

Table 2.7: Average Solution times with different algorithms for 15 units test case.

PSO parameters used in the algorithms:
- Maximum Velocity, $V_{max} = \frac{1}{2}P_{j, max}$, $j=1,2,\ldots,n$.
- Minimum Velocity, $V_{min} = \frac{1}{2}P_{j, min}$, $j=1,2,\ldots,n$.
- Inertia Weight, $w = 0.0$
- Constants, $c_1 = c_2 = 2.0$

The convergence characteristics of the algorithms are shown in fig. 2.13. Investigation of the figure reveals that IFEP-PSO' possesses better convergence rate than that of CEP-PSO'.

To investigate the effects of initial trial solutions, both the algorithms were run with 50 different initial trial solutions and the results are reported in fig. 2.14 and in table 2.6. In these results, the average cost is the prime indicator of competence of an algorithm in finding better quality solutions. It is observed that IFEP-PSO' is more competent than CEP-PSO'.

Table 2.7 shows the average solution times for both the algorithms, CEP-PSO' and IFEP-PSO'. Though the latter has taken more time as it uses both Gaussian and Cauchy mutations in creating offspring, comparing them and selecting the best, it has converged at faster rate and achieved better quality solutions as compared to that of CEP-PSO'. It is evident that IFEP-PSO' is more efficient in finding better quality solutions.
Figure 2.13: The convergence nature of IFEP-PSO' and CEP-PSO' algorithms for 15 units test case
Figure 2.14: The optimum costs obtained at different run in respect of IFEP-PSO' and CEP-PSO' for 15 units test case
2.12 Conclusion

Algorithms based on floating point GA (GAF), self-adaptive CEP, PSO, PSO-CEP, CEP-PSO and CEP-PSO' were developed and their performances are tested on a moderately large non-convex ELD problem with valve point loading effects. Results show that all the hybridized algorithms are more competent to individual algorithms. Amongst the hybrid algorithms, CEP-PSO' appears to be the most efficient in achieving better quality solutions at faster convergence rate. There is not much significant difference in performance between CEP-PSO and PSO-CEP while the former appears to have slight improvement over the latter. Solutions with different random trials also proved higher efficiency of the CEP-PSO' technique. Hence, CEP-PSO' technique can be considered as a very fast and reliable algorithm to solve non-convex ELD problems.

Moreover, algorithms based on CEP-PSO' and IFEP-PSO' were developed and their performances were tested on a test case of non-convex ELD problem with valve point loading effects. Results reveal that IFEP-PSO' appears to be more efficient in achieving better quality solutions at faster convergence rate. Solutions with different random trials also proved better performance of IFEP-PSO' technique over that of CEP-PSO' in finding quality solutions. Hence, IFEP-PSO' technique can be considered as a very fast and reliable algorithm to solve highly nonlinear non-convex ELD problems. Research works for further improvement of the algorithm can be carried out by considering a fuzzy adaptive strategy parameter.