Chapter 6 - Particle Swarm based machining process optimization

Eberhart and Kennedy [92] suggested a particle swarm optimization (PSO) based on the analogy of swarm of bird and school of fish. The PSO mimics the behavior of individuals in a swarm to maximize the survival of the species. In PSO, each individual makes his decision using his own experience together with other individuals’ experiences [93]. The algorithm, which is based on a metaphor of social interaction, searches a space by adjusting the trajectories of moving points in a multidimensional space. The individual particles are drawn stochastically toward the position of present velocity of each individual, their own previous best performance, and the best previous performance of their neighbors [94]. The main advantages of the PSO algorithm are summarized as: simple concept, easy implementation, robustness to control parameters, and computational efficiency when compared with mathematical algorithm and other heuristic optimization techniques.

PSO have been successfully applied to various fields of power system optimization [95], reactive power and voltage control [96, 97, 98, 99]. The original PSO mechanism is directly applicable to the problems with continuous domain and without any constraints. Therefore, it is necessary to revise the original PSO to reflect the equality/inequality constraints of the variables in the process of modifying each individual’s search. Yoshida et al. [100] suggested a modified PSO to control reactive power and voltage considering voltage security assessment. Since the problem was a mixed-integer nonlinear optimization problem with inequality constraints, they applied the classical penalty method to reflect the constraint-vviolating variables. Abido [94] developed a revised PSO for determining the optimal values of lag-lead design parameters of multi-machine power system stabilizers. In this study, the velocity of each parameter is limited to a certain value to reflect the inequality constraint problem in the dynamic process.
6.1. **PSO in Machining Parameter Optimization**

Particle Swarm Intelligent technique combines social psychology principles in socio-cognition human agents and evolutionary computations. PSO has been motivated by the behavior of organisms, such as fish schooling and bird flocking. Generally, PSO is characterized as a simple concept, easy to implement, and computationally efficient. Unlike the other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities. Thus, a PSO algorithm can be employed to solve an optimization problem.

The PSO coding scheme is to be defined and the initial population is produced. The computation with particle swarm intelligent operators is used to evaluate fitness with respect to the objective function. Fig 6.1 shows the PSO based optimization procedure.

The Swarm Intelligent is designed for optimization of four inputs, the feed (F), speed (V) depth of cut (D) and tool nose radius (R) and surface roughness (Ra) as output.

Accordingly in the proposed approach each particle (agent) represents a possible solution to the optimization task at hand. Initially a random set of 20 population is created for the particles to be optimized (i.e. feed, speed, depth of cut and nose radius). Each particle accelerates in the direction of its own personal best solution found so far during each iteration cycle, as well as in the direction of the global best position discovered so far by any of the particles in the swarm. If a particle discovers a promising new solution, all the other particles will move closer to it, exploring the region more thoroughly in the process. From this grouped population equal numbers of new populations are generated.
6.2. Swarm Intelligent Optimization

From a view of social cognition, each individual in PSO can benefit from both its own experience and group findings. In its theoretical base, some factors [101,102] are included: i) evaluation of stimulation; ii) influence to its behavior hereafter by its own experience; iii) influence to its behavior by other particles’ experience. The principle of PSO algorithm is as
follows [103]. Let \( X \) and \( V \) denote the particle’s position and its corresponding velocity in search space respectively. The PSO algorithm models the exploration of a problem space by a population of individuals; individual’s successes influence their searches and those of their peers.

The PSO algorithm searches in parallel using a group of individuals similar to other AI-based heuristic optimization techniques. An individual in a swarm approaches to the optimum or a quasi-optimum through its present velocity, previous experience, and the experience of its neighbors. In a physical \( n \)-dimensional search space, the position and velocity of individual \( i \) are represented as the vectors \( X_i=(x_{i1}, \ldots, x_{in}) \) and \( V_i=(v_{i1}, \ldots, v_{in}) \), respectively, in the PSO algorithm. Let \( P_{best_i}=(x_{i1}^{P_{best}}, \ldots, x_{in}^{P_{best}}) \) and \( G_{best}=(x_{i1}^{G_{best}}, \ldots, x_{in}^{G_{best}}) \), respectively, be the best position of individual and its neighbors’ best position so far. Using the information, the updated velocity of individual \( i \) is modified under the following equation in the PSO algorithm:

\[
V_{k+1}^i = wV_k^i + c_1 \text{rand}_1^* (P_{k}^{best_i} - X_k^i) + c_2 \text{rand}_2^* (G_{k}^{best} - X_k^i) \tag{6.1}
\]

where

\( V_k^i \): velocity of individual at iteration ;

\( W \): weight parameter;

\( c_1, c_2 \): weight factors;

\( \text{rand}_1, \text{rand}_2 \): random numbers between 0 and 1;

\( X_k^i \): Position of individual at iteration \( k \);

\( P_{k}^{best_i} \): Best position of individual until iteration \( k \);
The steps involved in PSO algorithms are:

1. Initialize an array of particles with random positions and velocities on D dimensions (parameters to be optimized i.e F,D,V,R).

   Set constants $k_{\text{max}}$, $C_1$, $C_2$.  

   Initialize particle position $X_0^i = X_{\text{min}} + \text{rand}(X_{\text{max}} - X_{\text{min}})$  

   Initialize particle velocity $V_0^i = X_{\text{min}} + \text{rand}(X_{\text{max}} - X_{\text{min}})$  

   Set $k=1$

2. Evaluate the desired minimization function ($F_k^i$).

   (a) If $F_k^i \leq F_{\text{best}}^i$, then $F_{\text{best}}^i = F_k^i$, $P_k^i = X_k^i$.

   (b) If $F_k^i \leq F_{\text{best}}^g$, then $F_{\text{best}}^g = F_k^i$, $P_k^g = X_k^i$.

   (c) If stopping condition is satisfied then go to 3.

3. Update particle velocities $V_{k+1}^i$

   $$ V_{k+1}^i = w V_k^i + c_1 \text{rand}_1 \ast (P_k^{\text{best}_i} - X_k^i) + c_2 \text{rand}_2 \ast (G_k^{\text{best}_i} - X_k^i) $$

4. Update particle positions $X_{k+1}^i$

   $$ X_{k+1}^i = X_k^i + V_{k+1}^i $$
5. Evaluate the objective function \( F_k^i \).

\[
\text{(a) \quad \text{If} \quad F_k^i \leq F_{\text{best}}^i \quad \text{then} \quad F_{\text{best}}^i = F_k^i, P_k^i = X_k^i}
\]

\[
\text{(b) \quad \text{If} \quad F_k^i \leq F_{\text{best}}^g \quad \text{then} \quad F_{\text{best}}^g = F_k^i, P_k^g = X_k^i}
\]

(c) If stopping condition is satisfied then go to 7, otherwise go to 5.

6. Increment K

7. Go to next iteration.

8. Terminate.
The search mechanism of the PSO using the modified velocity and position of individual based on (6.2), (6.3), (6.6) and (6.7) is illustrated in Fig. 6.2.

![Diagram of the search mechanism of the particle swarm optimization.](image)

Fig. 6.2 The search mechanism of the particle swarm optimization.

(X and Y axis represents direction of motion of particle)

**Stopping criteria**

There are many number of stopping criteria reported such as Maximum number of functional evaluation, Convergence criteria, computation time etc. In this work, Maximum number of functional evaluations has been used as stopping criteria. The detailed flow chart of the proposed PSO design is shown in Fig. 6.3.
Fig 6.3. Flowchart of the PSO design.
In order to optimize the present problem using PSO, the following parameters have been selected to obtain optimal solutions with less computational effort.

- No. of iterations-1000
- \( c_1 = 2 \)
- \( c_2 = 2 \)
- \( w = 0.5 \)

From the observed data for surface roughness, the response function has been determined using RSM and fitness function, defined as Minimize,

\[
R_a = -4.89 + 2.49F - 38.0D + 0.599V + 3.27R - 5.38F \times D + 0.0140F \times V - 18.2F \times R + 0.0097D \times V + 15.8D \times R - 0.232V \times R + 80.5F^2 + 16.5D^2 - 0.00318V^2
\]  

subject to

- \( 39.269 \text{ m/min} \leq V \leq 94.247 \text{ m/min} \)
- \( 0.059 \text{ mm/rev} \leq F \leq 0.26 \text{ mm/rev} \)
- \( 0.4 \text{ mm} \leq D \leq 1.2 \text{ mm} \)
- \( 0.4 \text{ mm} \leq R \leq 1.2 \text{ mm} \)

### 6.3. Simulation Studies and Performance Evaluation

The PSO code was developed using MATLAB. The input machining parameter levels were fed to the PSO program. It is possible to determine the conditions at which the turning operation has to be carried out in order to get the optimum surface finish. The fitness evaluation is described in figure 6.4. Table 6.1 shows the performance of surface roughness with respect to input machining parameters for PSO.
Table 6.1. Output values of the PSO with respect to input machining parameters

<table>
<thead>
<tr>
<th>Machining Parameters</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feed, F(mm/rev)</td>
<td>PSO</td>
</tr>
<tr>
<td></td>
<td>0.127395</td>
</tr>
<tr>
<td>Depth of cut, D(mm)</td>
<td>PSO</td>
</tr>
<tr>
<td></td>
<td>0.718475</td>
</tr>
<tr>
<td>Cutting Velocity, (m/min)</td>
<td>PSO</td>
</tr>
<tr>
<td></td>
<td>43.8783</td>
</tr>
<tr>
<td>Nose Radius, R(mm)</td>
<td>PSO</td>
</tr>
<tr>
<td></td>
<td>0.941211</td>
</tr>
<tr>
<td>Min. Surface Roughness, Ra(microns)</td>
<td>PSO</td>
</tr>
<tr>
<td></td>
<td>1.9938*10^{-7}</td>
</tr>
</tbody>
</table>

Fig.6.4. Performance of PSO
6.4. Summary

Based on this PSO algorithm the following conclusions may be drawn from the optimization results of the PSO program. Particle Swarm Optimization (PSO) is a relatively recent heuristic search method whose mechanics are inspired by the swarming or collaborative behavior of biological populations. PSO is more computationally efficient (uses less number of function evaluations). The basic PSO algorithm consists of three steps, namely, generating particles’ positions and velocities, velocity update, and finally, position update. Here, a particle refers to a point in the design space namely feed, speed, depth of cut and nose radius, that changes its position from one move (iteration) to another based on velocity updates. Even though PSO gives better result than SA, for improving the output results different algorithms named CGA and IGA is applied, which is discussed in the coming chapter.