Performance Enhancement of End Milling Process Using Particle Swarm Optimization

Muhammed Althaf, Muhammed Hashir O A*, Muhammed Mubeen, Dr. Sivaprasad P V

Department of Mechanical Engineering, Adi Shankara Institute of Engineering and Technology, A.P.J. Abdul Kalam Technological University, Kerala, India

* Corresponding author: muhammedhashiroa@gmail.com
doi: https://doi.org/10.21467/proceedings.7.5.14

ABSTRACT

End milling is a crucial machining process that has a significant impact on manufacturing productivity and part quality. Although gun metal alloys are widely used in industry, there is little research on their machinability. This work focuses on concurrently optimizing two machining responses; material removal rate and surface roughness while end milling gun metal. Particle Swarm Optimization (PSO) algorithm has used for this purpose. A structured methodology is proposed, which integrates Grey Relational Analysis to convert multiple responses to single response, trailed by PSO optimization of cutting speed, feed rate, and coolant application. This two-way tactic is used to recognize best machining parameters that boost productivity while sustaining better surface finish. The PSO-based technique fruitfully detects better machining parameters, as evidenced by experimental results. The study offers practical insights on selecting the optimal cutting conditions for industrial gun metal machining applications, which can lead to improved process efficiency and part quality.

Keywords: Gun metal, Milling, GRA, Modelling, PSO

1 Introduction

The end milling process is a key machining technique widely used in modern manufacturing industries to produce components with high precision and intricate geometries. This method comprises the use of a rotating cutting tool, known as a milling cutter, to remove material from a workpiece. Unlike other machining processes, end milling is particularly effective for creating parts with complex shapes and fine surface finishes, making it an indispensable tool in various advanced manufacturing applications. The primary goal of the end milling process is to shape the workpiece to meet the desired design specifications, which often involves operations like:

Profile milling, Slot milling, Face milling, 3D Contouring, etc. Multi criteria decision making (MCDM) techniques are mainly used in the scenarios where multiple outputs were assessed simultaneously. Using these techniques, multiple responses were converted to a single output. This single output is then evaluated. The effect of this single output actually explains the effects of the multiple outputs that it comprises of. Grey relational analysis (GRA) is one such technique. Particle Swarm Optimization (PSO) is a nature-inspired optimization algorithm that mimics the social behaviour of birds flocking or fish schooling. It is a population-based search algorithm used to find an optimal solution by adjusting the positions of individual particles based on their own experience and the experience of other particles in the swarm. PSO is widely used for solving continuous optimization problems, and its simplicity, ease of implementation, and effectiveness have made it a popular choice in many fields, including machine learning, control systems, robotics, and engineering. In this study, the effect of various end milling process parameters while machining gun metal were studied.

Specimen and Tool

In this work, commercially available gun metal, which is shown as figure 1 is taken for the study. An HSS end mill tool; shown as figure 2 has been used for the milling operation.

Proceedings DOI: 10.21467/proceedings.7.5; Series: AIJR Proceedings; ISSN: 2582-3922; ISBN: 978-81-989164-7-1

Figure 1: *Gun metal*

Figure 2: *HSS End mill*

2 Experimental Procedure

This study has been carried out in the following way. Initially, pilot studies have been made and decided the major process parameters affecting milling operation. After that, trial runs were made with gun metal to fix the parameters and the levels. Once the levels fixed, a design matrix has been designed to start the experiments. The fixed levels of each parameter are tabulated in Table 1. Taguchi's L_9 Orthogonal Array concept has been utilized for this. Two sets (two replications) of experiments were done in random order to minimize the machine errors. The developed design matrices are tabulated as Table 2 & 3 respectively. After the machining has done, outputs (material removal rate & surface roughness) were measured and start the analysis. The Minitab® statistical software is used for this purpose. The significance of individual process parameters can be assessed from the statistical analysis. Also the multi criteria decision making (MCDM) approcah, Grey Relational Analysis (GRA) which applied to identify the local best combination of the best process parameters. A linear regression model developed from the Minitab®, can be used for future prediction of the similar milling operation. As the experiments were restricted to minimal, the optimal results obtained from the GRA may be the local optima. Inorder to identify the global optima, an evolutionary optimization algorithm method named, Particle Swarm Optimization (PSO) has also been employed. Similar type of works were also done previously with only two process parameters [1], whereas in this work three machining parameters were taken for experimentation. Very few works have considered three parameters in which only one output response has been evaluated at a time. Studies were also analysed the roughness on various machining factors [2], yet they only focused on the single output optimization and not any simultaneous optimization of multiple responses. Multiple outputs were not evaluated simulataneously for more number of inputs [3], which is evaluated here. The convergence plots of the PSO technique were drawn to identify the optimal point corresponding the best machining process parameter combination globally.

2.1 Selected Machining Parameters

Table 1: *Process Parameters Selected for the Experiments*

Parameter	Level 1	Level 2	Level 3
Feed (mm/min) - F	50	100	150
Speed (RPM) - S	500	2000	4000
Coolant Type - C	Dry	Mist	Flood

Proceedings DOI: 10.21467/proceedings.7.5
ISBN: 978-81-989164-7-1

Series: AIJR Proceedings
ISSN: 2582-3922

An Orthogonal Array (OA) is a statistical design used to plan experiments. It allows you to explore multiple factors (independent variables) simultaneously while minimizing the number of experimental runs. The key characteristic of orthogonal arrays is that they ensure all combinations of factor levels are tested in a balanced way, allowing for efficient estimation of factor effects.

2.2 Design Matrix for Experimentation – 1

Table 2: *Design of Experiment (Set 1)*

Exp. No.	Exp. Order	Speed (RPM)	Feed (mm/min)	Coolant Type	Machining Time (min)
1	7	500	50	Dry	3:09
2	5	500	100	Mist	1:43
3	1	500	150	Flood	1:03
4	8	2000	50	Mist	3:05
5	3	2000	100	Flood	1:31
6	6	2000	150	Dry	1:06
7	4	4000	50	Flood	3:05
8	9	4000	100	Dry	1:38
9	2	4000	150	Mist	1:04

Desing matrix for experimentation -2

Table 3: *Design of Experiment (Set 2)*

Exp. No.	Exp. Order	Speed (RPM)	Feed (mm/min)	Coolant Type	Machining Time (min)
1’	5	500	50	Dry	3:02
2’	7	500	100	Mist	1:37
3’	3	500	150	Flood	1:04
4’	4	2000	50	Mist	3:03
5’	8	2000	100	Flood	1:37
6’	6	2000	150	Dry	1:04
7’	2	4000	50	Flood	3:05
8’	9	4000	100	Dry	1:33
9’	1	4000	150	Mist	1:04

2.3 Machining Conditions

Three different coolant conditions are used here. These types are illustrated as Figure 3-5.

Figure 3: *Dry*

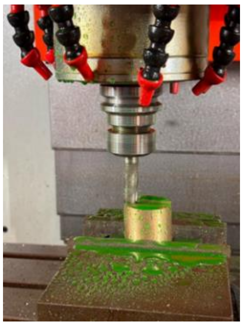

Figure 4: *Mist*

Figure 5: *Flood*

3 Results and Discussion

The material removal rate and the surface roughness measured from each machined sample were tabulated below in Table 4 & 5.

3.1 Material Removal Rate (MRR) -1 & 2

Table 4: *MRR (Set 1)*

Specimen No.	Weight before machining (gm)	Weight after machining (gm)	Time for the machining (min)	Material Removal Rate (MRR) gm/min
1	695	657.67	3:09	12.08
2	720	680.94	1:43	27.31
3	730	687.98	1:03	40.79
4	925	882.21	3:05	14.02
5	785	739.43	1:31	34.78
6	720	681.51	1:06	36.31
7	685	647.91	3:05	12.16
8	675	634.26	1:38	29.52
9	685	635.32	1:04	47.76

Table 5: *MRR (Set 2)*

Specimen No.	Weight before machining (gm)	Weight after machining (gm)	Time for the machining (min)	Material Removal Rate (MRR) gm/min
1'	657.67	618.74	3:02	12.89
2'	680.94	641.09	1:37	29.08
3'	687.98	648.00	1:04	38.44
4'	882.21	843.34	3:03	12.82
5'	739.43	699.19	1:37	29.37
6'	681.51	644.09	1:04	35.98
7'	647.91	605.92	3:05	13.76
8'	634.26	596.70	1:33	28.24
9'	635.32	595.72	1:04	38.07

Proceedings DOI: 10.21467/proceedings.7.5
ISBN: 978-81-989164-7-1

Series: AIJR Proceedings
ISSN: 2582-3922

3.2 Surface Roughness (SR)

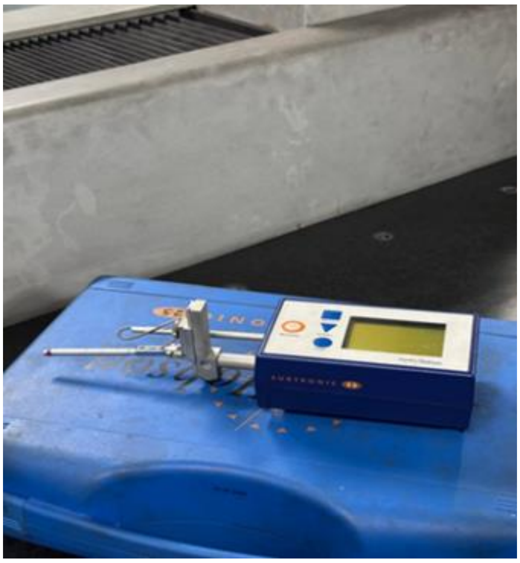

Figure 6: *Perthometer, Surtronic 25*

The Perthometer Surtronic 25 is a precision surface roughness measuring device; shown in Figure 6, is broadly used in manufacturing, electronics, motorized, and aerospace industries. It measures the texture of surfaces using a stylus that moves across the surface, detecting height variations to determine roughness parameters like Ra (average roughness), Rz (average maximum height), and Rt (total height). The surface roughness of the machined samples was measured using this instrument. All measurements were done twice and the average of these values were tabulated in the Table 6 & 7 given below.

SR value for the Matrix 1 & 2

Table 6: *SR (Set 1)*

Specimen No.	Speed (RPM)	Feed (mm/min)	Coolant	Avg. Surface Roughness
1	500	50	Dry	1.4 µ
2	500	100	Mist	1.86 µ
3	500	150	Flood	4.1 µ
4	2000	50	Mist	0.46 µ
5	2000	100	Flood	0.58 µ
6	2000	150	Dry	1.32 µ
7	4000	50	Flood	0.64 µ
8	4000	100	Dry	0.38 µ
9	4000	150	Mist	0.64 µ

Table 7: *SR (Set 2)*

Specimen No.	Speed (RPM)	Feed (mm/min)	Coolant	Avg. Surface Roughness
1’	500	50	Dry	1.36 µ
2’	500	100	Mist	2.01 µ
3’	500	150	Flood	3.84 µ
4’	2000	50	Mist	0.67 µ
5’	2000	100	Flood	0.49 µ
6’	2000	150	Dry	1.47 µ
7’	4000	50	Flood	0.68 µ
8’	4000	100	Dry	0.45 µ
9’	4000	150	Mist	0.86 µ

Analysis using MINITAB

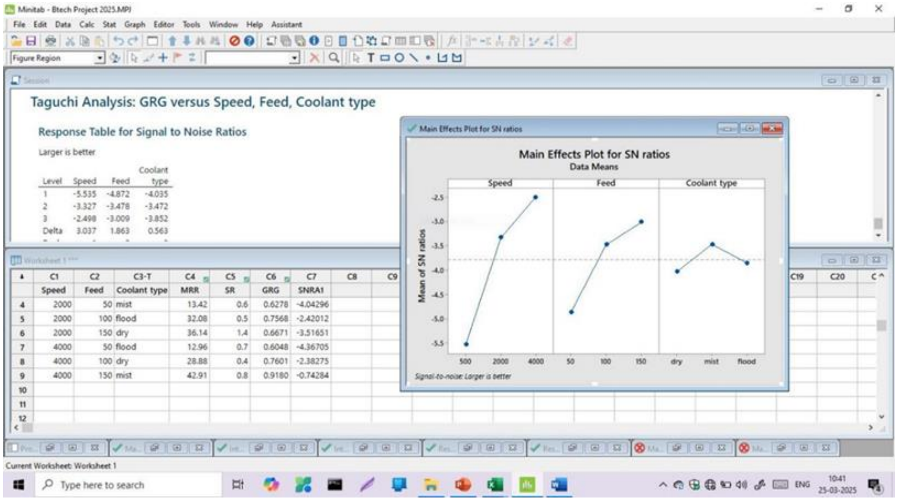

Figure 7: *Window of Minitab*

Minitab® is a powerful statistical analysis software widely used for data analysis, quality improvement, and decision-making in several industries, as well as manufacturing, healthcare, education, and research. Initially developed in 1972, Minitab simplifies complex statistical methods, making them accessible to users with varying levels of expertise. The software offers a comprehensive suite of tools for performing statistical analysis, counting regression analysis, hypothesis testing, ANOVA, control charts, and time series analysis, etc. Figure 7 illustrates the window of the software executing statstical analysis. All the analysis plots were developed from this software.

S/N Ratio Plot and Explanation:

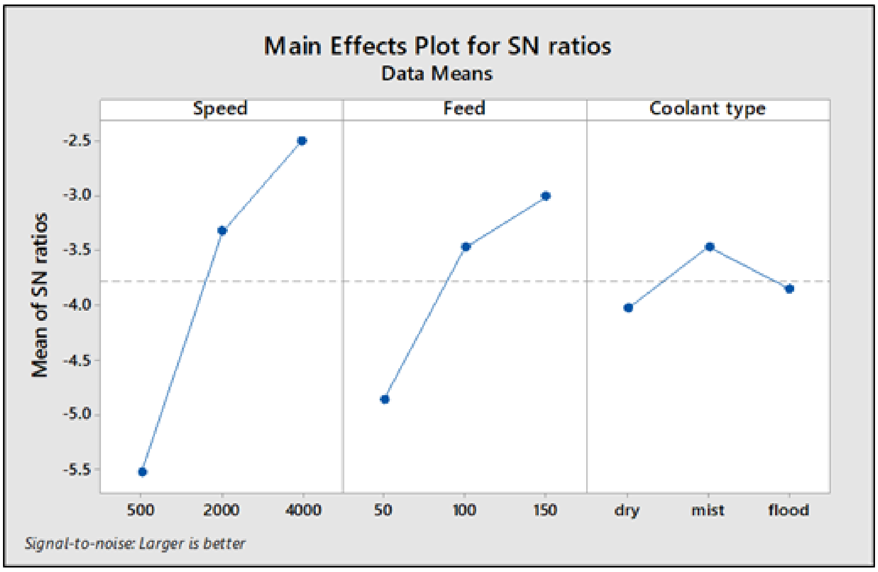

Figure 8: *S/N ratio*

The Main Effects Plot for Signal-to-Noise (SN) Ratios is depicted in figure 8. This plot is an essential tool used to evaluate the impact of various machining inputs on performance, particularly when following the 'Larger is Better' criterion. This particular criterion is used when the objective is to maximize the output. In this analysis, it was observed that as 'speed' increases from 500 to 4000, the SN ratio improves, suggesting that higher speeds lead to better machining. This indicates that increasing the speed has a positive effect on the output, consistent with the principle that larger values of certain factors generally lead to improved results. Similarly, the SN ratio also improves as 'feed' increases from 50 to 150, highlighting that a higher feed rate enhances performance. This further supports the 'Larger is Better' approach, where higher values

Proceedings DOI: 10.21467/proceedings.7.5
ISBN: 978-81-989164-7-1

Series: AIJR Proceedings
ISSN: 2582-3922

of feed are associated with better outputs**.** Table 8 provides the parameter effect(influence) over the output. It can be seen that, speed having the highest delta (difference) value is the most influencing parameter among the three. This is followed by 'feed' and then the least significant factor, 'type of coolant.' The coolant has not made any much changes to the output if we change it in to different levels. This means, at present set of experiments, these coolant levels are not sufficient enough to explain any variations in outputs. May be with a greater number of experiments (instead of 9 fractional factorial experiments, 27 full factorial experiments) can be dealt with the coolant's effect clearly.

Table 8: *Response Table for Signal to Noise Ratios - ('Larger is Better' Criteria)*

Level	Speed	Feed	Coolant type
1	-5.535	-4.872	-4.035
2	-3.327	-3.478	-3.472
3	-2.498	-3.009	-3.852
Delta	**3.037**	**1.863**	**0.563**
Rank	**1**	**2**	**3**

3.3 GRA Steps

Grey Relational Analysis (GRA) is a method within Multi-Criteria Decision-Making (MCDM) that is particularly effective for problems involving uncertain, incomplete, or vague data (referred to as 'grey' information). It helps assess the relationships between different decision-making alternatives and desired outcomes by quantifying the strength of the relationship between them. Other studies were happened using advanced neural network models and modelling [4-5] demands high computational efficiency and large set of data; which is a time consuming and cost involving process. GRA on the other hand, often used for small or large set of data with minimal computational and cost requirements. GRA involves normalizing the data to a comparable scale, calculating Grey Relational Coefficients (GRCs) to measure how closely each alternative matches the ideal solution, and determining the Grey Relational Grade (GRG) by averaging the GRCs across all criteria. The sequence in finding the GRG is as follows:

Step 1: Normalisation

For larger-the-better characteristic of Material Removal Rate (MRR), the output (MRR) value can be normalized as follows:

$$nMRR = \frac{MRR\ value - \ MRRmin}{MRRmax - MRRmin}$$

Similarly, for smaller-the – better characteristics of the Surface Roughness (SR), the original output can be normalized as follows:

$$nSR = \frac{SRmax - \ SRvalue}{SRmax - SRmin}$$

Step 2: Calculate Deviation Sequence

Deviation sequence for Material Removal Rate (MRR) = dMRR = 1 – nMRR

Deviation sequence for Surface Roughness (SR) = dSR = 1- nSR

Step 3: Calculate Grey Relational Coefficient

$$\text{GRC} = \frac{\Delta\min + \zeta\Delta max}{\Delta 0i(k) + \zeta\Delta max}$$

' ζ ' is distinguishing or identification coefficient, the value of 'ζ' is taken as 0.5 (Equal weightage for both outputs).

Step 4: Calculate Grey Relational Grade

Grey Relational Grade (GRG) = Average of gMRR and gSR

The value of GRG varies between 0-1. Table 9 indicates the GRG calculation result and grading of each GRG. Value near to 1 indicates, optimum.

Table 9: *GRG Calculations*

	Process Parameters			**Outputs**				
Sl. No:	Speed	Feed	Coolant	MRR	SR	GRG	Rank	
1	500	50	Dry	12.49	1.38	0.4895	9	
2	500	100	Mist	28.20	1.94	0.5230	8	
3	500	150	Flood	39.62	3.97	0.5775	7	
4	2000	50	Mist	13.42	0.57	0.6278	5	
5	2000	100	Flood	32.08	0.54	0.7568	3	
6	2000	150	Dry	36.15	1.40	0.6671	4	
7	4000	50	Flood	12.96	0.66	0.6048	6	
8	4000	100	Dry	28.88	0.40	0.7601	2	
9	4000	150	Mist	42.92	0.75	0.9180	**1**	**Optimal**
			Max	42.915	3.97	**0.9180**		
			Min	12.485	0.4			

So, we optimized that, max GRG value is 0.9180 which led to better performance.

Regression Model Equation

GRG = 0.3717+ 0.000065.S + 0.001468.F

0.3717: This is a constant or baseline value, which might represent the initial value of the GRG without considering the effects of speed and feed.

(0.000065 x Speed): This term indicates that the speed parameter has a positive effect on GRG. For every unit increase in speed, the GRG increases by 0.000065.

(0.001468 x Feed): This term shows that feed also positively influences the GRG. For every unit increase in feed, the GRG increases by 0.001468.

We can predict value with putting appropriate speed and feed in the regression model equation. The prediction accuracy is tabulated below in Table 10. It is clear from the table that, the prediction accurateness is decent for the given set of data with this model. Yet, for out of the design matrix data, it is found to be less. It is mainly because of the smaller number of data sets that used for formulating the model. Full factorial experiments will provide more relative accuracy for out of the box data sets also.

Table 10: *Prediction Accuracy*

S	R*sq*	R*sq(adj)*	R*sq(pred)*
0.07680	75.55%	67.40%	41.77%

3.4 Optimization using PSO

Particle Swarm Optimization (PSO) is a computational way stirred by the societal behaviour of birds flocking or fish schooling, applied for solving optimization problems by iteratively refining candidate solutions. In PSO, a populace of candidate solutions, termed particles, explores the problem space to find optimal solutions. Each particle alters its position based on its own knowledge and that of its neighbours, guided by simple mathematical rules. This technique mainly applied to identify the best optimal solution in the global search space (all possible process parameter combinations) very effectively (with a smaller number of computation iterations). The performance of PSO was proven to be superior to other soft computing techniques like genetic algorithm (GA) & simulated annealing (SA). PSO was utilized by most researchers due to its simplicity, and it also has the features of both GA and evolution strategies. [6]. A simple flowchart of the PSO algorithm is illustrated as figure 9.

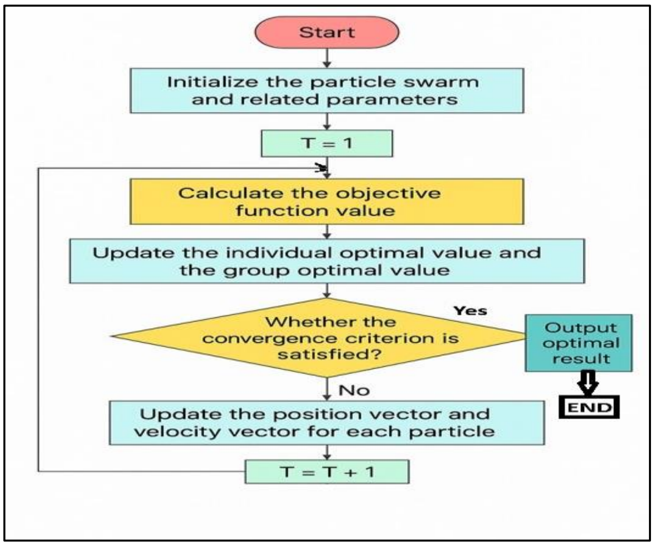

Figure 9: *Flow chart of PSO*

The simplest structure of the pseudocode for the PSO algorithm is:

1. For each iteration (i = 1 to max_iter):

2. For each particle (j = 1 to number of particles):

a. Evaluate fitness of current position of particle j:

fitness_j = evaluate_fitness(X[j])

b. If fitness_j is better than Pbest[j] (the personal best position):

Update Pbest[j] = X[j]

c. If fitness_j is better than Gbest (the global best position):

Update Gbest = X[j]

d. Update the velocity (V) of particle j:

*V[j] = w * V[j] + c1 * random() * (Pbest[j] - X[j]) + c2 * random() * (Gbest - X[j])*

e. Update the position (X) of particle j:

X[j] = X[j] + V[j]

3. Output: Return the global best position (Gbest) as the optimal solution.

The PSO code written for the current work is given below:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
# GRG function definition
def grg(speed, feed):
    return 0.3717 + 0.000065 * speed + 0.001468 * feed
# PSO parameters
num_particles = 30
max_iterations = 100
w = 0.7  # inertia weight
c1 = 1.5  # cognitive coefficient
c2 = 1.5  # social coefficient
# Problem parameters
speed_bounds = (500, 4000)
feed_bounds = (50, 150)
dimensions = 2  # speed and feed
# Initialize particles
particles = np.zeros((num_particles, dimensions))
velocities = np.zeros((num_particles, dimensions))
personal_best_positions = np.zeros((num_particles, dimensions))
personal_best_scores = np.full(num_particles, -np.inf)
global_best_position = np.zeros(dimensions)
global_best_score = -np.inf
# Initialize particles randomly within bounds
particles[:, 0] = np.random.uniform(speed_bounds[0], speed_bounds[1], num_particles)
particles[:, 1] = np.random.uniform(feed_bounds[0], feed_bounds[1], num_particles)
personal_best_positions = particles.copy()
# Initialize velocities
velocities[:, 0] = np.random.uniform(-speed_bounds[1], speed_bounds[1], num_particles)
velocities[:, 1] = np.random.uniform(-feed_bounds[1], feed_bounds[1], num_particles)
# Track convergence
convergence_curve = []
# PSO main loop
for iteration in range(max_iterations):
    for i in range(num_particles):
        # Evaluate current particle
        current_speed = particles[i, 0]
        current_feed = particles[i, 1]
        # Apply bounds
```

```
            current_speed = np.clip(current_speed, speed_bounds[0], speed_bounds[1])
            current_feed = np.clip(current_feed, feed_bounds[0], feed_bounds[1])
            particles[i, :] = [current_speed, current_feed]
            current_score = grg(current_speed, current_feed)
            # Update personal best
            if current_score > personal_best_scores[i]:
                personal_best_scores[i] = current_score
                personal_best_positions[i] = particles[i].copy()
            # Update global best
            if current_score > global_best_score:
                global_best_score = current_score
                global_best_position = particles[i].copy()
        convergence_curve. Append(global_best_score)
        # Update velocities and positions
        for i in range(num_particles):
            r1, r2 = np.random.random(2)
            velocities[i] = (w * velocities[i] +
                            c1 * r1 * (personal_best_positions[i] - particles[i]) +
                            c2 * r2 * (global_best_position - particles[i]))
            particles[i] += velocities[i]
        # Early stopping if convergence is reached
        if iteration > 10 and abs(convergence_curve[-1] - convergence_curve[-10]) < 1e-6:
            break
# Results
print(f"Optimal Speed: {global_best_position[0]:.2f} rpm")
print(f"Optimal Feed: {global_best_position[1]:.2f} mm/min")
print(f"Maximum GRG: {global_best_score:.6f}")
# Convergence plot
plt.figure(figsize=(10, 6))
plt.plot(convergence_curve, linewidth=2)
plt.title('PSO Convergence Curve', fontsize=14)
plt.xlabel('Iteration', fontsize=12)
plt.ylabel('Best GRG Score', fontsize=12)
plt.grid(True)
plt.show()
# 3D Surface plot
speed_values = np.linspace(speed_bounds[0], speed_bounds[1], 50)
feed_values = np.linspace(feed_bounds[0], feed_bounds[1], 50)
speed_grid, feed_grid = np.meshgrid(speed_values, feed_values)
grg_values = grg(speed_grid, feed_grid)
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
# Plot surface
surf = ax.plot_surface(speed_grid, feed_grid, grg_values, cmap='viridis', alpha=0.8)
# Plot optimal point
ax.scatter(global_best_position[0], global_best_position[1], global_best_score,
          color='red', s=100, label='Optimal Point')
# Labels and title
```

```
ax.set_xlabel('Speed (rpm)', fontsize=12)
ax.set_ylabel('Feed (mm/min)', fontsize=12)
ax.set_zlabel('GRG', fontsize=12)
ax.set_title('GRG Surface with Optimal Point', fontsize=14)
ax.legend()
# Add colorbar
fig.colorbar(surf, ax=ax, shrink=0.5, aspect=5)
plt.tight_layout()
plt.show()
```

The convergence plot shows the optimal score the iteration number and the 3D plot provide a better visual analysis on the best input parameters and output (GRG). Figure 10 illustrate the convergence plot and the figure 11 is the developed 3D plot.

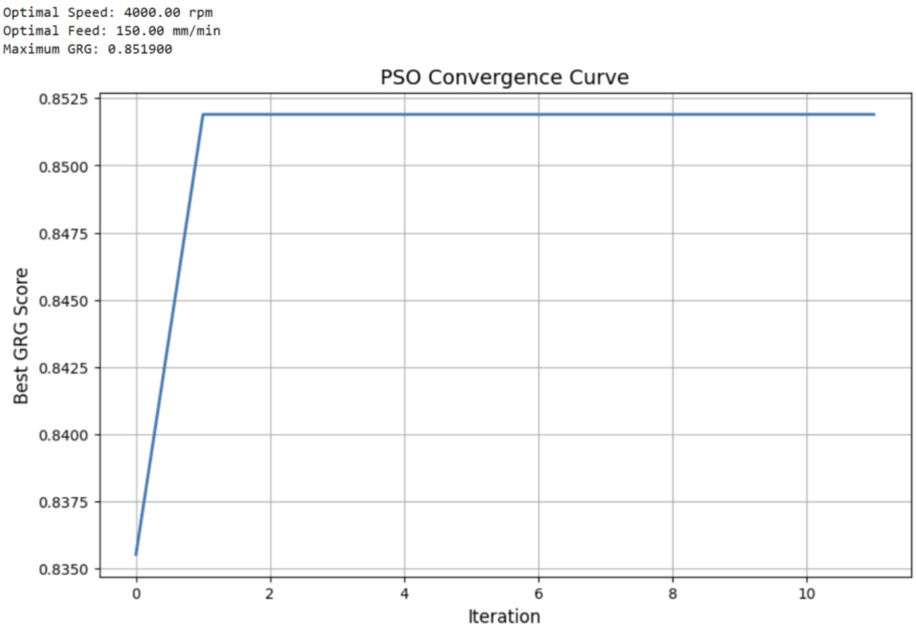

Figure 10: *PSO Convergence Plot*

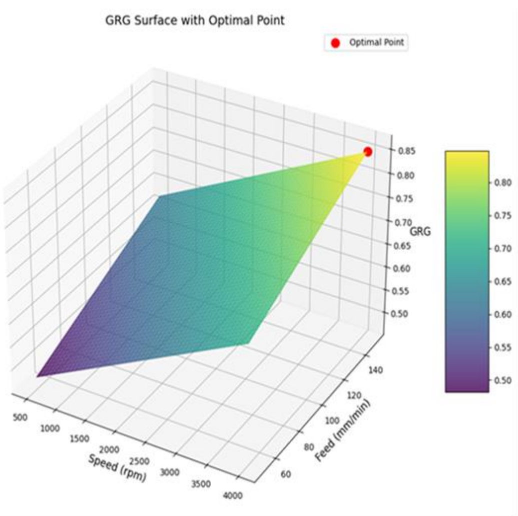

Figure 11: *3D Plot*

4 Conclusions

Both the local and global optima are found to be the same, it is not necessary to validate the parameter combination again. The best optimal parameters identified are spindle speed 4000 rpm, feed rate 150 mm/min, and Mist as the coolant (S4000/F150/C Mist). Grey Relational Analysis (GRA) has been employed to assess the collective effect of multiple machining parameters on various output responses. The outputs were combined in to one single output called as the GRG which used for further analysis. Among all tested combinations, the condition S4000/F150/C Mist showed the highest Grey Relational Grade (GRG) value of 0.918 in the Signal-to-Noise (S/N) ratio plot, indicating superior machining performance. Coolant type has been excluded from being a significant factor in the optimization process due to its relatively low influence on the output. This has been identified from the SN ratio plot (Figure 8) and the response table (Table 8). The Particle Swarm Optimization (PSO) algorithm was selected for this study owing to its high convergence efficiency and ability to reach optimal solutions in fewer iterations. In this case, minimal iteration only taken to reach the convergence and it is found that, the local and the global optimal sticks with the same parameter combination. It is always appropriate to have large number of experiments to achieve a better result.

5. Publisher's Note

AIJR remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

How to Cite

Althaf *et al.*, "Performance Enhancement of End Milling Process Using Particle Swarm Optimization", *AIJR Proc.*, vol. 7, no. 5, pp. 111-123, Sep. 2025. doi: https://doi.org/10.21467/proceedings.7.5.14

References

[1]. V. K. Tandon, Hazim El-Mounayri, and H. A. Kishawy, "NC end milling optimization using evolutionary computation," vol. 42, no. 5, pp. 595–605, Apr. 2002, doi: https://doi.org/10.1016/s0890-6955(01)00151-1.

[2]. X. Wu and X. Yin, "Surface Roughness Analysis and Parameter Optimization of Mold Steel Milling," *Procedia CIRP*, vol. 71, pp. 317–321, 2018, doi: https://doi.org/10.1016/j.procir.2018.05.032.

[3]. S. Bharathi Raja and N. Baskar, "Application of Particle Swarm Optimization technique for achieving desired milled surface roughness in minimum machining time," Expert Systems with Applications, vol. 39, no. 5, pp. 5982–5989, Apr. 2012, doi: https://doi.org/10.1016/j.eswa.2011.11.110.

[4]. M. S. J Hossain, "Cutting Parameter Optimization for End Milling Operation Using Advanced Metaheuristic Algorithms," *International Journal of Advanced Robotics and Automation*, vol. 2, no. 2, pp. 1–12, Jun. 2017, doi: https://doi.org/10.15226/2473-3032/2/2/00125.

[5]. M. Soori, B. Arezoo, and R. Dastres, "Machine Learning and Artificial Intelligence in CNC Machine Tools, A Review," Sustainable Manufacturing and Service Economics, p. 100009, Jan. 2023, doi: https://doi.org/10.1016/j.smse.2023.100009.

[6]. N. Yusup, A. M. Zain, and S. Z. M. Hashim, "Overview of PSO for Optimizing Process Parameters of Machining," *Procedia Engineering*, vol. 29, pp. 914–923, 2012, doi: https://doi.org/10.1016/j.proeng.2012.01.064.