PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY SAAD DAHLEB OF BLIDA

Faculty of Sciences and Technology

Department of Mechanics

End Of Study dissertation

For obtaining the MASTER's degree

Option: Mechanical Construction

Presented by :

Redouan BELAL
Mohamed FELLAH

Theme

Prediction of surface roughness using artificial intelligence in multi-axis machining

Defended publicly on june 23, 2024. Befor the jury members:

President	MELZI Nesrine	MCB	USDB, Blida
Examiner	MEGHATRIA M'Hamed	MAA	USDB, Blida
Supervisor	BOUHADJA Khadidja	MRB	CDTA, Baba Hassen

june 2024

Acknowledgements

Praise be to the Almighty Allah who has given us faith, courage, and patience to carry out this work.

We would like to express our sincere gratitude to all those who have contributed to the completion of this thesis.

First and foremost, We are deeply grateful to our supervisors khadidja BOUHADJA , zahida TCHANTCHANE and mohammed OUALI for their invaluable guidance, encouragement, and support throughout this research endeavor. Their expertise, patience, and constructive feedback have been instrumental in shaping the direction and content of this thesis.

We would like to extend our appreciation to the faculty members of department of Mechanics [BLida1], whose lectures, seminars, and discussions have provided us with a solid foundation of knowledge and skills essential for this project.

Special thanks are due to our colleagues and friends for their camaraderie, encouragement, and assistance during challenging times. Their support has been a constant source of motivation and inspiration.

Lastly, We would like to express our gratitude to all the individuals [Chouaib], and institutions [CDTA] whose contributions, whether direct or indirect, have contributed to the successful completion of this thesis.

Thank you all for your support, encouragement, and belief in us.

Dedication

I dedicate this work to my parents:

May they find here the testimony of my deep gratitude and acknowledgment

To my brothers and my sisters, my grandparents and my family who give love and liveliness.

To all those who have helped me - directly or indirectly - and those who shared with me the emotional moments during the accomplishment of this work and who warmly supported and encouraged throughout my journey.

To all my friends who have always encouraged me, and to whom I wish more success.

Thanks!

Mohamed

Dedication

I would like to thank ALLAH first and for all. My mother my heart my soul and everything i got in this life ,my father,the beautiful lovely flowers in my garden,special thanks to you for being through it all.

My grandmother, aunts and uncles and every single person who helped in my career whether you were family members,friends or colleagues know that all your favors are engraved in mind, may god almighty help me repay you all.

My colleague and partner in research and our guiding supervisors for taking us into this academic journey.

Last but not least our Muslim brothers and sisters in Palestine and around the world.

Redouan

ملخص

يقدم هذا البحث تحقيقاًا شاملاً في التنبؤ والتحسين لخشونة السطح في عمليات التصنيع الميكانيكية. تلعب جودة السطح دوراً حاسماًا في ضمان الوظائف والجماليات للأجزاء الميكانيكية، مما يجعل التنبؤ والتحكم الدقيق في خشونة السطح أمراً حيوياًا في العمليات التصنيعية.

تبدأ الدراسة بنظرة عامة على جودة السطح، لتأكيد أهمية تحقيق أبعاد دقيقة وتشطيبات سطحية في تصنيع الأجزاء الميكانيكية. بعد ذلك، يتم إجراء استعراض لأحدث التقنيات لتوقع خشونة السطح، والتي تغطي النهج القائمة على نظرية التصنيع، النهج القائمة على التجارب المصممة والنماذج القائمة على الذكاء الاصطناعي.

ثم يتحول التركيز إلى تنفيذ نموذج الشبكة العصبية الاصطناعية - خوارزمية حركة الجزيئات $(ANN - PSO)$ لتوقع خشونة السطح. يتم تفصيل عملية التدريب وهندسة النموذج وتحسين معلمات النموذج وعرض النتائج في الفصول التالية. من خلال التجارب المنهجية وتحسين المعلمات، يظهر نموذج $ANN - PSO$ تحسيناًا ملحوظاًا في دقة التنبؤ بخشونة السطح عبر شروط التشغيل المختلفة.

تؤكد النتائج التي تم الحصول عليها من التجارب فعالية نموذج $ANN - PSO$ في التنبؤ بخشونة السطح، مع ملاحظة تحسينات كبيرة في دقة التنبؤ مقارنة بالتنبؤات الأولية. تلخص الخاتمة نتائج الدراسة، وتناقش تداعياتها، وتوضح سبل البحث المحتملة للمستقبل في مجال تنبؤ وتحسين جودة السطح في عمليات التصنيع الميكانيكي.

بشكل عام، يسهم هذا البحث في تقدم الفهم والسيطرة على خشونة السطح في عمليات التصنيع الميكانيكي، مما يمهد الطريق لزيادة كفاءة التصنيع وجودة المنتج في مختلف الصناعات.

كلمات مفتاحية: التحسين، التنبؤ، الذكاء الاصطناعي، التصنيع، خشونة السطح.

Abstract

This thesis presents a comprehensive investigation into the prediction and optimization of surface roughness in mechanical machining processes. Surface quality plays a critical role in ensuring the functionality and aesthetics of mechanical parts, making accurate prediction and control of surface roughness imperative in manufacturing operations.

The study begins with an overview of surface quality, emphasizing the importance of achieving precise dimensions and surface finishes in mechanical part manufacturing. Subsequently, a review of state-of-the-art techniques for predicting surface roughness is conducted, covering machining theory-based approaches, designed experiments-based methodologies, and artificial intelligence-based models.

The focus then shifts to the implementation of an Artificial Neural Network-Particle Swarm Optimization (ANN-PSO) model for surface roughness prediction. The training process, model architecture, optimization of model parameters, and presentation of results are detailed in the subsequent chapters. Through meticulous experimentation and parameter optimization, the ANN-PSO model demonstrates improved predictive accuracy in surface roughness estimation across various machining conditions.

Results obtained from experiments validate the efficacy of the ANN-PSO model in predicting surface roughness, with significant improvements observed in predictive accuracy compared to initial predictions. The conclusion summarizes the findings of the study, discusses their implications, and outlines potential avenues for future research in the field of surface quality prediction and optimization in mechanical machining processes.

Overall, this thesis contributes to advancing the understanding and control of surface roughness in mechanical machining processes, paving the way for enhanced manufacturing efficiency and product quality in various industries.

Key words: *Prediction, Optimization, Surface roughness, Machining, Artificial intelligence.*

Résumé

Cette thèse présente une enquête approfondie sur la prédiction et l'optimisation de la rugosité de surface dans les processus d'usinage mécanique. La qualité de surface joue un rôle crucial dans la garantie de la fonctionnalité et de l'esthétique des pièces mécaniques, rendant la prédiction précise et le contrôle de la rugosité de surface impératifs dans les opérations de fabrication.

L'étude commence par un aperçu de la qualité de surface, en mettant l'accent sur l'importance de l'obtention de dimensions précises et de finitions de surface dans la fabrication de pièces mécaniques. Ensuite, une revue des techniques de pointe pour la prédiction de la rugosité de surface est effectuée, couvrant les approches basées sur la théorie de l'usinage, les méthodologies basées sur les expériences conçues et les modèles basés sur l'intelligence artificielle.

L'accent est ensuite mis sur la mise en œuvre d'un modèle Réseau de Neurones Artificiels-Optimisation par Essaim Particulaire (ANN-PSO) pour la prédiction de la rugosité de surface. Le processus de formation, l'architecture du modèle, l'optimisation des paramètres du modèle et la présentation des résultats sont détaillés dans les chapitres suivants. Grâce à une expérimentation méticuleuse et à l'optimisation des paramètres, le modèle ANN-PSO démontre une amélioration de la précision prédictive dans l'estimation de la rugosité de surface dans diverses conditions d'usinage.

Les résultats obtenus à partir des expériences valident l'efficacité du modèle ANN-PSO dans la prédiction de la rugosité de surface, avec des améliorations significatives observées dans la précision prédictive par rapport aux prédictions initiales. La conclusion résume les résultats de l'étude, discute de leurs implications et esquisse des avenues potentielles pour la recherche future dans le domaine de la prédiction et de l'optimisation de la qualité de surface dans les processus d'usinage mécanique.

Dans l'ensemble, cette thèse contribue à faire progresser la compréhension et le contrôle de la rugosité de surface dans les processus d'usinage mécanique, ouvrant la voie à une efficacité de fabrication accrue et à une qualité de produit améliorée dans diverses industries.

Mots clés : *Prédiction, Optimisation, Rugosité, Usinage, Intelligence artificielle.*

Contents

List of Figures

List of Tables

List of Abbreviations

RSM:	Response Surface Methode
SQ:	Surface Quality
SR:	Surface Roughness
DoE:	Design of Experiment
OAs:	Orthogonal Arrays
ANOVA:	Analyse of Variance
ML:	Machine Learning
ANN:	Artificial Neural Network
ANFIS:	Adaptive Neuro-Fuzzy Interface System
SVM:	Support Vector Machine
GPR:	Gaussian Process Regression
HSDM:	High Speed Dry Milling
CNC:	Computer Numerical Control
GA:	Genetic Algorithm
MAE:	Mean Absolute Error
MSE:	Mean Square Error
RMSE:	Root Mean Square Error
HS:	Harmony Search

General introduction

1. Context

In practical manufacturing, mechanical parts often cannot be produced to the exact dimensions required by the designer due to inherent inaccuracies in the manufacturing processes. These imperfections are acknowledged as manufacturing defects. To minimize these defects and manufacture high-quality parts within reduced timeframes, machining is performed using high-speed multi-axis machines. The objective is to select appropriate machining conditions to achieve the desired finished surface using optimization tools and artificial intelligence.

In this study, we employed the ANN-PSO model to predict surface roughness.

2. Problem Statement

In the manufacturing industry, achieving ideal dimensions for mechanical parts poses a significant challenge due to the inherent limitations of fabrication processes. These limitations result in imperfections known as manufacturing defects. To mitigate these defects and produce high-quality parts efficiently, machining is conducted using high-speed multi-axis machines. The aim is to identify suitable machining conditions to attain the desired surface finish through the utilization of optimization tools and artificial intelligence techniques. In this context, the prediction of surface roughness plays a crucial role in ensuring the quality of machined components.

3. Objectives

The primary objective of this study is to enhance the quality of machined surfaces through the prediction of surface roughness using the ANN-PSO model. By employing advanced computational techniques, we aim to optimize machining parameters and leverage artificial intelligence to achieve the desired surface finish. Through this approach, we seek to minimize manufacturing defects and improve the overall quality of mechanical parts within a reduced time frame.

4. Organization of the Document

This thesis is organized into several chapters:

- The first chapter provides an overview of surface quality, highlighting the importance of achieving precise dimensions and surface finishes in mechanical parts manufacturing.
- The second chapter delves into the state-of-the-art techniques for predicting surface roughness, including machining theory-based approaches, designed experiments-based approaches, and artificial intelligence-based approaches.
- The third chapter details the implementation of the ANN-PSO model for surface roughness prediction, including the training process, model architecture, optimization of model parameters, and presentation of results.

The concluding chapter summarizes the findings of the study, discusses their implications, and outlines potential avenues for future research in the field of surface quality prediction and optimization in mechanical machining processes.

Chapter 1

Overview of surface quality

1.1 Introduction

In modern industry, the goal is to produce high-quality products at low cost and in a short time. Automated and flexible manufacturing systems are used for this purpose, as well as machine tools capable of achieving high accuracy and very low processing time [Choudhury and El-Baradie, 1997].

In recent years, the industrial sector, particularly the machine tool manufacturing sector, has made significant progress. The machines have diversified both in terms of their structure and the types of machining offered, both conventional and unconventional. In addition, the emergence of new high-performance tools and innovative machining strategies meets the growing needs in terms of quality and complexity of mechanical parts. At the same time, in today's consumer environment, the pressure to reduce production costs while ensuring increased part quality has become unavoidable.

1.2 Surface Quality

Producing good quality, appropriate surface finish, and geometry are important for the machined workpiece. The surface finish or surface texture based on [Abernethy et al., 1985] is defined as geometrical irregularities of solid materials surface while surface roughness is defined as the finer irregularities of the surface texture, usually resulting from the inherent action of the production process, such as feed marks produced during machining. The surface roughness is commonly indicated by parameters such as average roughness (Ra) or root mean

square roughness (Rq) and calculated by Eqts. 1.1 and 1.2, respectively.[Subbiah, 2014]

$$R_a = \frac{1}{L}\int_0^L [Y(x)]\,dx \tag{1.1}$$

$$R_q = \sqrt{\frac{1}{L}\int_0^L [Y(x)]^2\,dx} \tag{1.2}$$

where:

L is the sampling length

Y(x) is the ordinate of the profile curve

1.2.1 Importance of surface quality

At one time measurement of the surface was considered largely irrelevant but it soon became apparent that the finish on the surface was extremely sensitive to any changes in the process. Hence it became logical to assume that measurement of the surface could be used to control the process of manufacture [Whitehouse, 1997].

The geometric and material properties of surfaces have a significant impact on friction, wear, fatigue [Ben Mhenni, 2007], corrosion, as well as electrical and thermal conductivity [Serope, 1928]. An evaluation of the surface texture is essential for product quality control.

1.2.2 Problems related to surface quality

The machining surface concept was developed with the aim of improving the quality of the machined surfaces by combining a surface representation with the machining paths. The qualitative advantages result from the integration of design functional constraints into the construction of the machining surface, thus enabling the machined surface to meet the requirements of the designer. From the point of view of trajectory generation, the improvements arise from the continuous surface representation of the tool path, as opposed to conventional approaches where the path is a discrete representation.

Every tool geometry and for every type of machining, whether 3 or 5 axes, end or sidewall, corresponds to a more precise definition of the machining surface. In three-axis milling with a hemispherical tool, the definition of the machining surface corresponds to the parallel surface if the fixed point under consideration is the center of the sphere modeling the active part of the tool.

The generation of trajectories using the parallel surface has already been the subject of work [Kim, 1995]. Among the problems encountered, the most restrictive are the problems

of loops [Maekawa et al., 1997] and precision [Farouki, 1986]. The problem of loops or self-intersection of parallel surfaces arises from using a tool whose radius is greater than the concave radius of curvature of the surface. Thus, to avoid loop problems, tools with a radius smaller than the smallest concave radius of the surface to be machined are used. This seems consistent in the context of finish machining to generate iso-peak paths. Accuracy problems arise from the model of representation of parallel surfaces. Indeed, in the majority of cases, it is not possible to model these surfaces by a parameterized surface of type NURBS without approximation.

1.2.3 Surface finish and roughness

Surface finish refers to the surface quality of a part or material after a machining or manufacturing process. It is evaluated in terms of roughness, flatness, surface defects, and other characteristics that define the texture and appearance of the surface. In the field of machining, surface finish is often measured in terms of parameters such as roughness, the dimensional and geometric quality of the part produced, and the quality of the surfaces obtained [Pateloup S, 2011].

The surfaces, regardless of their methods of production, have defects compared to the prescribed theoretical surface. The latter are of different orders ranging from a shape defect to a crystalline defect of interatomic dimension [Bhushan, 2012]. The surface finish is observable through micro and macro geometric defects identifiable at different scales (orders) and decomposable into several cumulative orders (Figure 1.1) . Macrogeometric defects correspond to orders 0 to 2, while roughness (microgeometric defects) is at the level of orders 3 to 6. Surface finish defects, in mechanical engineering, essentially refer to machining by material removal [Davim, 2010]. A first orientation defect, macro geometric (order 0), appears in relation to the theoretical profile. It corresponds to a geometrical defect of the NCM. A macro-geometric (order 1) defect in form is compounded by the previous defect in orientation. It is related to the machine (quality of the kinematic structure, elastic deformations) and to poor fixings of the tool or workpiece. The last macro-geometric defect is the undulation (order 2). It is mainly caused by eccentric tool rotation, vibration, severe tool wear and/or inhomogeneous machined material.

A periodic roughness defect, microgeometric (order 3), is superimposed on the undulation defect (macrogeometric defect). The latter is formed by short-wavelength fluctuations of the surface characterized by dips and protrusions. The periodic roughness comes mainly from the machine kinematics and the morphology of the chips. This is followed by an aperiodic

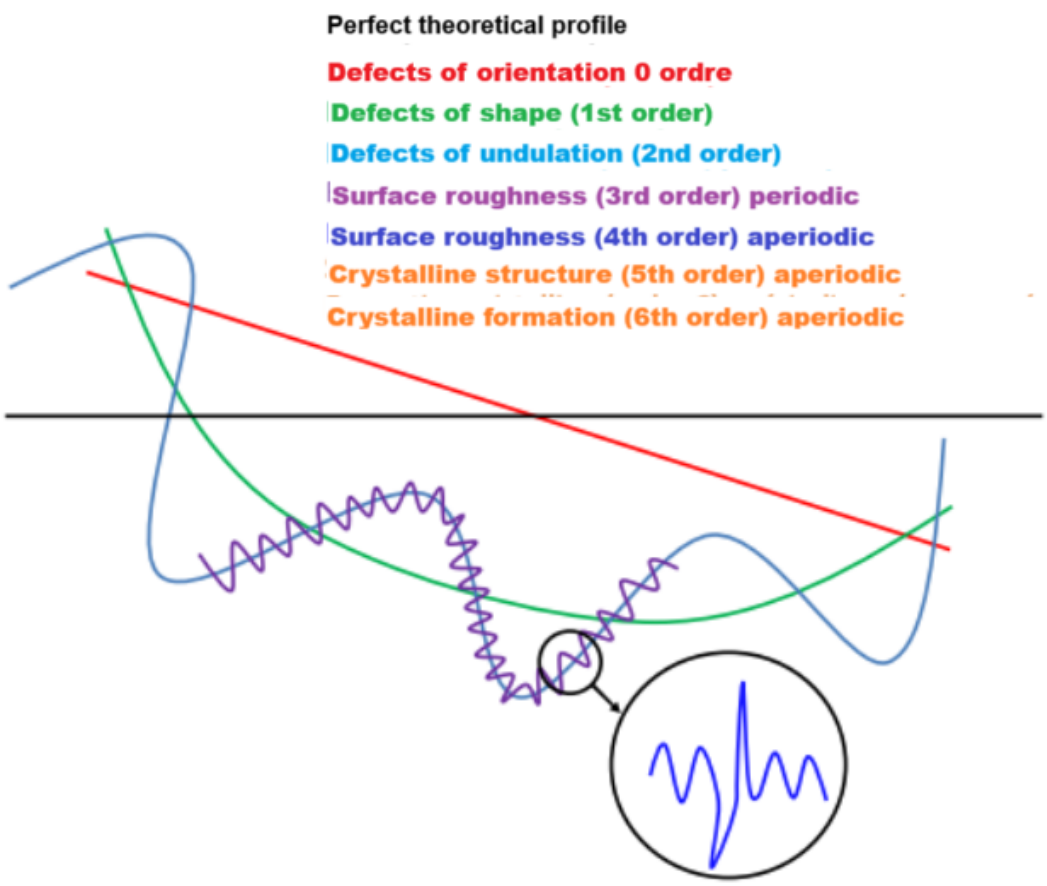


Figure 1.1: The different scale orders of surface finish defects. [Lebon, 2017]

roughness (order 4) due to wear of the tool tip and the way the chips are formed. At smaller scales, there are aperiodic defects in structure and crystal formation (orders 5 and 6), resulting from the decrystallization mode, irregularities related to chemical reactions, deformations of the crystal networks, physical and chemical alterations occurring in the fine structure of the material. In addition to the previously defined micro and macro geometric orders, the [ISO4287 98] standard associates three prefixes to the basic parameters. Prefix parameters P: These are parameters calculated on the primary profile, a general profile corresponding to orders 2 to 4.Prefix parameters W: These are parameters calculated on the wave profile corresponding to order 2.Prefix parameters R: These are parameters calculated on the surface roughness profile corresponding to orders 3 and 4. The R profile is calculated by applying a filter that removes the wavelength elements from the W profile. The R profile is therefore an intentional modification of the P profile. This is the profile used for the roughness study.

1.2.3.1 Definition of profilometry (2D and 3D)

According to [Métrologie, 1999], profilometry consists of characterizing the geometry of a surface by focusing only on the variations of the side Z(X, Y) locally normal to the mean surface as a function of the X and Y parameters of position on the surface, at different scales.

Two scales can be studied:

- The scale of the entire part (macroscopic scale): Examine the deviations of the average surface area from a perfect surface of simple shape (for example, plane, sphere, cylinder, or cone). In this type of study, the roughness is determined by specifying a local average surface area.
- The microscopic scale (a few micrometers or a few tens of micrometers in X and Y): this is a question of what is called roughness, which will generally not be studied over the entire surface, but on a few intelligently distributed samples.

Two types of profilometry can be defined:

- 2D profilometry: Consists of analyzing a single profile of the sample Z(X) surface. In mathematics, it is a one-dimensional (1D) analysis that allows the height to be represented at any point with respect to a reference line.
- 3D profilometry: Consists of analyzing many parallel profiles Z(X, Y). In mathematics or signal processing, it corresponds to a 2D analysis (2D image). This analysis makes it possible to identify the three-dimensional topography of a surface.[Lebon, 2017]

The [ISO4287 98] [Iso-4287, 1998] standard defines 14 2D parameters divided into 4 classes (amplitude parameters, spacing parameters, hybrid parameters, curves and associated parameters). Similarly, the [ISO25178-2 12] [Iso-25178-2, 2012] standard defines 32 3D parameters divided into 5 classes (height parameters, spacing parameters, hybrid parameters, curves and associated parameters, miscellaneous parameters) (Table 1.1). There is a similarity between the definitions of the two classes 2D and 3D, as well as an equivalence between the 2D and 3D parameters. I.e., when possible, there is a 3D parameter homologous to the 2D parameter.

The arithmetic mean deviation of the evaluated profile (Ra) corresponds to the arithmetic mean of the absolute values of the y-intercepts within a base length defined by the standard [ISO4287 98]. It is almost the only roughness parameter involved in the evaluation of surface integrity for several authors [Yin L, 2003] [Yin L, 2006b] [Yin L, 2006a]

Table 1.1: Summary of standard roughness parameters

14 2D Profile Roughness Parameters	
Amplitude parameters	Rp, Rv, Rz, Rc, Rt, Ra, Rq, Rsk, Rku
Spacing parameter	RSm
Hybrid Parameter	Rq
Curves and Associated Parameters	Rmr(c), Rc, Rmr
32 3D Surface Roughness Parameters	
Amplitude parameters	Sq, Ssk, Sku, Sp, Sv, Sz, Sa
Spacing parameter	Sal, Str
Hybrid Parameter	Sdq, Sdr
Curves and Associated Parameters	Smr(c), Smc(mr), Sk, Spk,Svk, Smr1, Smr2, S, Spq, Smq, Sxp, Svs(c), Srel(c), Svfc, Safc, Vv, Vvv, Vvc, Vm(p), Vmc
Miscellaneous parameter	Std

[Klopfstein M. J, 2011] [Institute, 1986]. The characterization of roughness by this parameter is also widely established in dentistry. The major criticism of the Ra parameter is that it doesn't distinguish between dips and protrusions. There are roughness parameters that are very sensitive to surface dips and protrusions and especially to extreme values. Of these, the first, Rt, corresponds to the total height of the profile. It is the sum of the greater of the profile protrusion heights and the greater of the profile trough depths within the evaluation length defined by the standard [ISO4287 98]. The second, Rz, is the maximum height of the profile. It is the sum of the greater of the profile projection heights and the greater of the profile trough depths within a base length (different from the evaluation length) defined by the [ISO4287 98] standard. For 3D parameters, the parameter Sa (arithmetic mean height of the limited-scale surface) is the counterpart of Ra. This is the arithmetic average of the absolute value of the ordinates within a definition area defined by the standard [ISO25178-2, 12]. The Sz parameter (maximum surface height at limited scale) characterizes peaks and valleys in 3D. This is the homologous parameter of Rz and corresponds to the sum of the greatest peak heights and the greatest pit depths within a definition zone.

The Sq parameter (RMS Height of the Limited Scale Surface) is the root mean square (RMS) of the y-values within a definition box. The calculation of Sq is equivalent to a standard deviation calculation of roughness. So, for a given Sa, a surface with a few large

peaks/pits spread over a smoother nominal surface will have a higher Sq.

1.2.3.2 Correlations between Functionalities and roughness parameters

Since roughness has been identified as an essential component of IS in the representativeness of features, it is necessary to deepen it. The choice of roughness parameters to best represent the expected functionalities is essential. Each selected parameter must be in agreement with the functionality to be represented (Petropoulos et al) [Petropoulos G. P, 2010] identified the following correlations between the functionalities and the 2D roughness parameters (Tab.2).

Table 1.2: Significant features of 2D roughness parameters [Petropoulos G. P, 2010]. (**pronounced influence *normal influence)

Functionalities	Ra, Rq	Rp, Rpm	Rt, Rz	Rsk	Rku	RSm	Wa
Friction	*	**	*	*	*	*	*
Fatigue (cracks)	*	*	**	*			
Thermal conductivity	*	**	**	*			
Electrical conductivity	*	*	*				*
Light Reflection	**						
Usury	*	**	**	**	*	*	*
Lubrification	*	*	**	**	*	**	**
Airtightness	*	**	**	**			**
Corrosion	*	*	*	*			
Assemblage	*	**	**				**

It appears that the Ra/Rq parameters have an impact on 90% of the identified functionalities. Secondly, the Rt/Rz parameters have a pronounced influence on 80% of the selected features. In most cases, a set of parameters is required to define the roughness of a surface representing a feature. In the same way as associating components with a feature, it is possible to associate multiple parameters with a component. Each parameter has a different level of representativeness (relative weight) of functionality. The same parameter can be assigned different weights when it occurs in multiples components. A set of components or a set of parameters is often required to define a feature. However, many authors still evaluate surface integrity 38 through the independent measurement of parameters from each other, without weighting the parameters in line with the importance given to the functionality(s)

they represent. "For example, from a dental point of view, it can be interesting to give more weight to the quality of the edge of a crown (chipping) rather than to the roughness of the surface when machining fragile materials. This weighting will make it possible to choose the most appropriate process parameters." [Lebon, 2017]

1.2.4 Surface Finish Analysis

1.2.4.1 Context

Surface roughness refers to the evolution of the surface area compared to the average surface. The DIN4760 standard defines deviation orders [Fuer, 1982]. [Benardos and Vosniakos, 2003] describe these deviation orders. They indicate that:

The first and second orders of deviation relate respectively to shape, i.e. flatness, roundness, etc., and to undulation. They are caused by machine tool errors, part deformation, misconfigurations, clamping, vibration and material inhomogeneity of the workpiece.

The third and fourth order of deviation refer to periodic grooves and cracks and squandering that are related to the shape and condition of the cutting edges, chip formation and process kinematics.

The fifth and sixth order of deviation referring to the material structure of the part which is related to physico-chemical mechanisms that act on the grains and the lattice scale (slippage, diffusion, oxidation, residual stress, etc.)

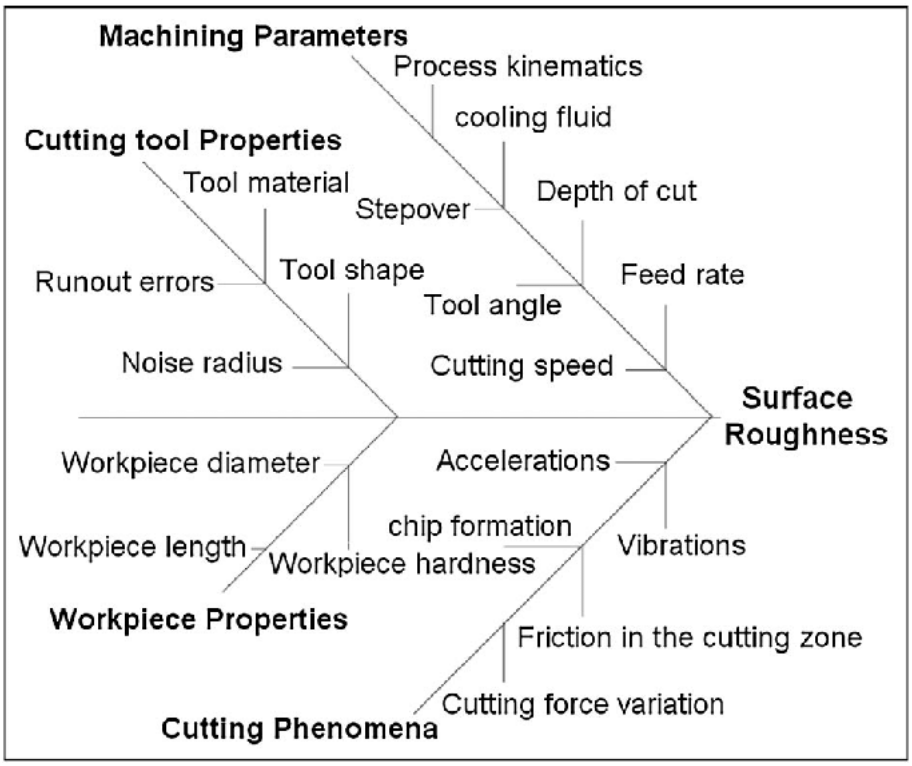


Figure 1.2: Diagram showing the parameters influencing surface roughness.

Numerous parameters related to the machining processes and/or the properties of the parts influence the surface roughness. Figure 1.2 shows a non-exhaustive diagram of these parameters.

1.2.4.2 Three-dimensional analysis

1.3 Multi-axis machining

the pressure to reduce production costs while ensuring increased part quality have led to the widespread adoption of multi-axis machining and the development of High Speed Machining (HSC) with the emergence of new roughing and finishing machining strategies, whether in 2D1/2 or 3D.

Technological advances in the field of machine tools have been remarkable, especially with regard to numerical controls and drive axes. The travel and rotation axes have become much more dynamic, especially thanks to the use of high-torque motors. These advances have also been accompanied by significant improvements in cutting tools, which has enabled a major evolution of HSC, resulting in a radical reduction in production times.

Today, the industry has a wide range of machine tools, whether flexible or specialized, small or large, with or without HSC, with 3-axis, 5-axis or more. However, despite this diversity, the literature presents different structures of multi-axis machine tools, with generally two types of machines mainly used: those with 3 axes and those with 5 axes. [Karlo, 2009]

1.3.1 Fundamentals of multi-axis machining

Multi-axis machining refers to a machining process that involves moving the cutting tool along multiple axes simultaneously to create complex shapes and surfaces [Lee R, 1997]. It is particularly useful for achieving geometric variety, high precision, and machining of sculpted surfaces. Geometric errors in multi-axis machining have a direct impact on the position of the tool tip, leading to a reduction in machining accuracy, making it crucial to identify and correct these errors [Cheng Q, 2014]. Optimizing tool posture in multi-axis machining is essential to ensure collision-free operations and maximize the capabilities of multi-axis milling machines [Lauwers B, 2003].

Multi-axis machining offers advantages over traditional machining methods by offering an increased number of degrees of freedom, allowing for a trade-off between performance and economy [Gasick J, 2021]. It is widely applied in various industrial machines such as precision CNC machine tools and laser cutting machines [Wang S, 2021]. The use of multi-axis machine tools in the machining of complex parts such as aircraft turbine discs results in accurate dimensions, high quality, and the required surface roughness [Lee J, 2022].

Efficient toolpath generation is essential for multi-axis machining, especially roughing operations, to ensure optimal material removal and surface finish [Umehara T, 2007]. The integration of post-processors for five-axis machine tools plays a crucial role in increasing the productivity and automation of multi-axis machining operations [She C, 2012]. The linear axis, a fundamental component of multi-axis machine tools, requires accurate measurement and compensation of geometric errors to ensure machining accuracy [Zhang Z, 2011].

1.3.2 Advantages and Challenges of Multi-Axis Machining

Multi-axis machining offers many advantages and applications in modern manufacturing processes. It provides increased flexibility and precision in machining complex geometries, such as sculpted surfaces and turbines [Chen et al., 2022] [Lee R, 1997]. Multi-axis machining allows for higher metal removal rates, shorter cutting times, and improved surface finishes compared to traditional three-axis machining [Hong et al., 2011]. In addition, it allows for the production of precise parts with better surface finishes after a segment of

the part has been molded, demonstrating its versatility in hybrid manufacturing processes [Kelkar and Koc, 2008].

In addition, multi-axis machining is crucial for high-precision machining of multi-axis, high-speed, variable-curvature parts, highlighting its importance in achieving complex designs with outstanding engineering applications [Jiang et al., 2021]. The use of multi-axis machining centers, especially five-axis centers, has been demonstrated to improve the quality and efficiency of machining operations compared to lower-axis machines. This advancement is particularly beneficial for the processing of blades and other components requiring complex geometries and high precision [Dai and Ren, 2016].

However, despite its advantages, multi-axis machining faces challenges such as low rigidity and low vibration resistance compared to three-axis machining [Zhao et al., 2010]. To solve these problems, research focuses on improving the positioning accuracy of multi-axis machine tools [Lee and Yang, 2010]. In addition, the accuracy of multi-axis machining is highly dependent on the tracking error of the control system, which highlights the need for advanced control strategies to improve machining accuracy [Hu and Gai, 2009].

1.4 Machining operation

Many manufactured products require machining at some stage of their production sequence. Machining is the removal of unwanted materials (machining allowance) from the workpiece so as to obtain a finished product of the desired size, shape, and surface quality. Generally, machining ranges from relatively rough cleaning of castings to high-precision micromachining of mechanical components that require narrow tolerances. The removal of the machining allowance through cutting techniques was first adopted using simple handheld tools made from bone, stick, or stone that were replaced by bronze or iron. The water, the steam, and, later, the electricity were used to drive such tools in the power-driven metal-cutting machines (machine tools). The development of new tool materials opened a new era to the machining industry where machine tool development took place. Nontraditional machining techniques offered alternative methods for machining parts of complex shapes in harder, stronger, and tougher materials that were difficult to cut by the traditional methods.[El-Hofy, 2018]

1.4.1 Classification of Machining Processes

Traditional machining requires a tool that is harder than the workpiece that is to be machined. This tool penetrates into the workpiece for a certain depth of cut. A relative motion

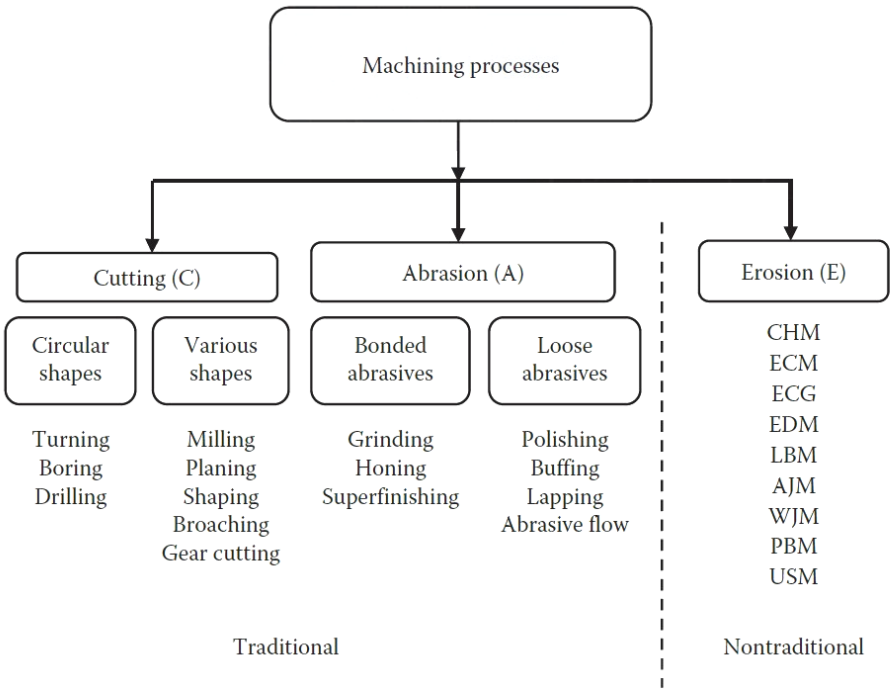


Figure 1.3: classification of machining processes

between the tool and workpiece is responsible for form and generation cutting to produce the required shapes, dimensions, and surface quality. Such a machining arrangement includes all machining by cutting (C) and mechanical abrasion (MA) processes. The absence of tool hardness or contact with the workpiece makes the process nontraditional, such as the erosion processes (E) by electrochemical and thermal machining methods (see Figure 1.3).[El-Hofy, 2018]

1.4.2 Machining by Cutting

Figure 1.3 shows the main components of a typical metal-cutting process. The machining system includes the tool, the workpiece, and the machine tool that controls the workpiece and tool motions required for the machining process. Table 1.1 shows the different tool and workpiece motions for some important metal-cutting operations. During machining by cutting, the tool is penetrated into the workpiece as far as the depth of cut. Cutting tools have a definite number of cutting edges of a known geometry. Moreover, the machining allowance is removed in the form of visible chips. The shape of the produced workpiece depends on the relative motions of the tool and workpiece. In this regard, three different cutting arrangements are possible, as depicted in Figure 1.3.[El-Hofy, 2018]

1.4.3 Machining operations

1.4.3.1 Turning

Turning utilizes a single cutting tool to create a surface of revolution. The cylindrical workpiece is rotated around its axis while a cutting tool is fed parallel to the axis of rotation. As the cutting tool is engaged into the workpiece, a new surface of revolution is generated by removing a layer of material whose thickness is equal to the depth of the tool engagement. A typical machine tool that generates the necessary motions for carrying out this operation is an engine lathe. A typical engine lathe is shown in (Figure 1.4). CNC lathe operates on similar kinematics principles. The machine tool provides a primary motion to the workpiece in revolutions per minute and a secondary motion to the cutting tool in millimeters per revolution. The combined motion that generates the surface is the vector addition of these two motions. For most practical applications, the feed motion is much smaller than the primary motion and the cutting speed is determined by the primary motion alone.[El-Hofy, 2018]

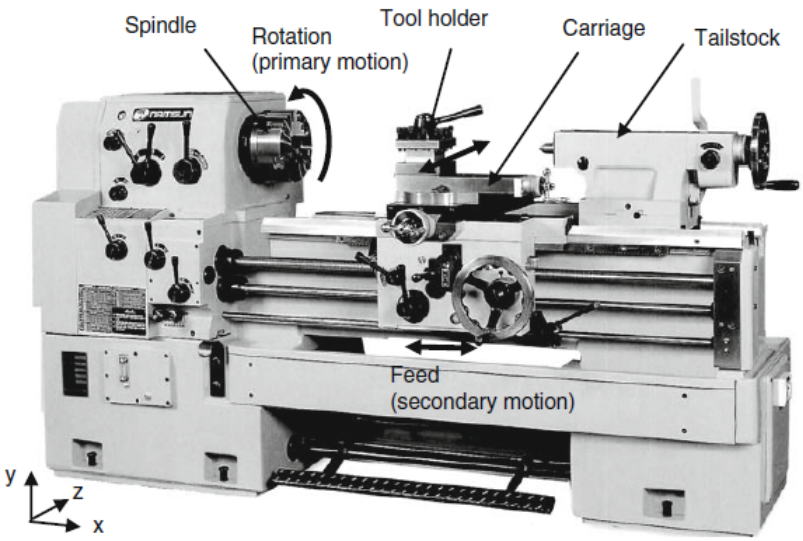


Figure 1.4: Principal components and movements of a typical lathe

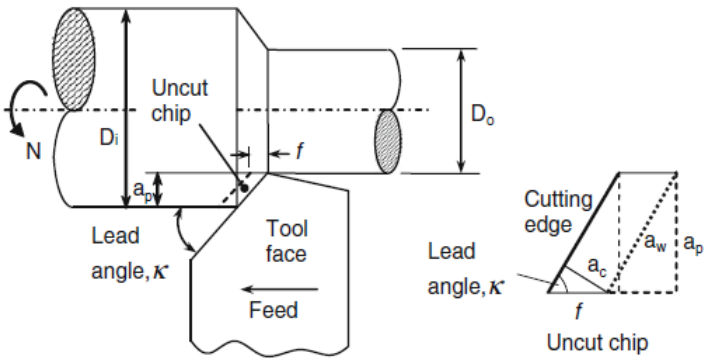


Figure 1.5: Cutting geometry in turning

The following relationships apply to single point tools with small corner radius or when the depth of cut is very large as compared to the corner radius (Figure 1.5) shows the cutting geometry with a single point cutting tool.
The cutting speed is determined by the rotational speed of the spindle, N, given in rev/min, and the workpiece initial and final diameters, Di and Do, respectively [Sheikh-Ahmad, 2009]

$$v = \Pi N \frac{D_i - D_0}{2} \cong \Pi N D_i \tag{1.3}$$

The average feed motion advances the tool per revolution along a specified direction. The feed, F , is given in rev/min and the feed speed, vf, is related to the feed by

$$v_f = fN \tag{1.4}$$

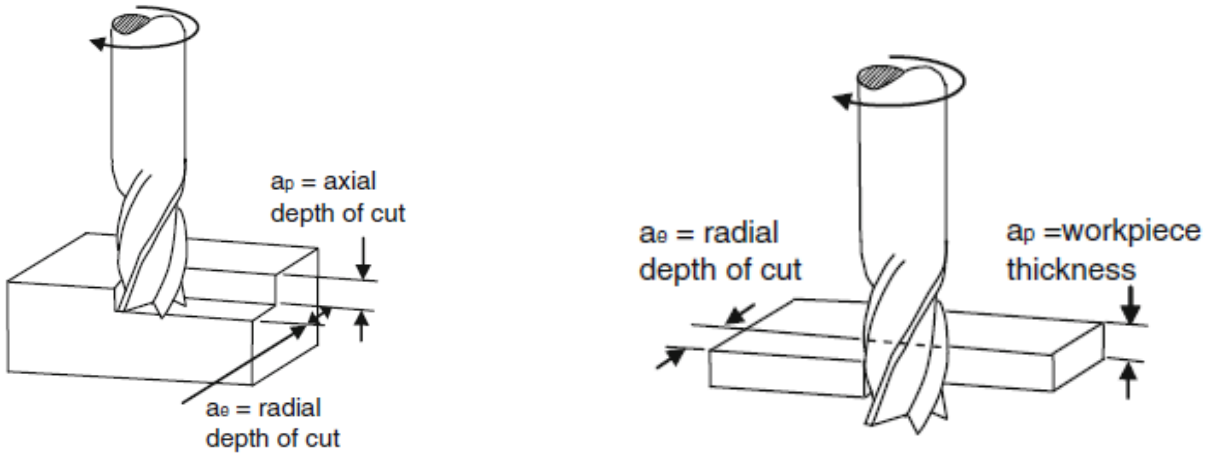
The radial depth of cut describes the thickness of material removed from the workpiece and is given by

$$a_p = \frac{D_i - D_0}{2} \tag{1.5}$$

1.4.3.2 Milling and Trimming

In milling, material is removed from the workpiece by a rotating cutterhead that may have more that one active cutting edge. The types of milling operations that are most common in machining are peripheral milling or profiling and end milling. (Figure 1.6a) illustrates these milling operations. Peripheral milling uses the cutting edges on the periphery of the tool. The machined surface is parallel to the axis of rotation of the cutter and the engagement into the workpiece is in the radial direction of the cutter (Figure 1.6b). Peripheral milling is more appropriately called edge trimming because the tool diameter is usually small and the

axial engagement encompasses the entire thickness of the workpiece. End milling is similar to peripheral milling, except that the axial engagement may be less than the thickness of the part and a slot is obtained.



(a) End Milling (b) Edge Trimming

Figure 1.6: Types of milling operations [Sheikh-Ahmad, 2009]

The machine tool most commonly used is a vertical milling machine. The machine tool provides the primary motion to the spindle (to which the cutter is held) and feed motions to the machine table (to which the workpiece is held). CNC routers capable of providing higher spindle speeds and feed rates, more flexibility, and larger workspace than a typical milling machine are commonly used in high production facilities. (Figure 1.7) shows the principal components and movements of typical industrial three-axis CNC router. Hand-held routers are commonly used for edge trimming of thin workpiece. The router provides the primary rotational motion to the cutter while the operator feeds the tool into the workpiece manually. End milling and trimming operations are further classified into up (or conventional) milling and down (or climb) milling, depending on how the cutting edge approaches the workpiece. These operations are illustrated in (Figure 1.8). In up milling, the direction of cutting speed of the edge in contact with the workpiece is opposite to the direction of feed. In down milling, the direction of the cutting speed is the same as that of the feed. The resulting chip area in both cases has a "comma" shape and the length of the chip is described by a torchoid that results from the superposition of peripheral motion and feed motion. In up milling the cutting edge begins engaging the chip at the thin section of the comma shape. This results in low engagement forces and in lifting up of the workpiece. In down milling, the cutting

edge engages the chip at the thick section of the comma shape. The engagement forces are high

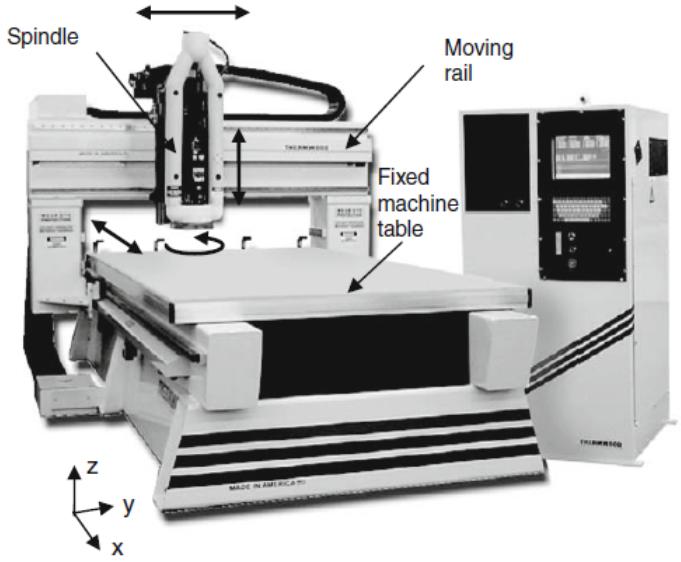


Figure 1.7: Principal components and movements of a three-axis gantry bridge router

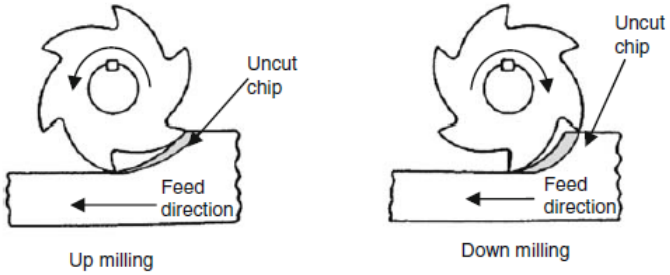


Figure 1.8: Illustration of up and down milling operations [Sheikh-Ahmad, 2009]

and result in pushing the workpiece against the workholding surface. Cutting forces in milling are also not continuous. In up milling, the forces gradually increase from zero at beginning of tool engagement to a maximum when the cutting edge is about to leave the workpiece. Forces drop to zero again when the cutting edge leaves the workpiece. (Figure 1.9) shows a schematic of the cutting geometry for one cutting edge in up milling. The tool path is torchoidal and is generated from the combination of rotational (spindle) and translational (feed)motions. The exact geometry and kinematics of up and down milling have been thoroughly investigated by Martellotti and Foenigsberger and Sabberwal among others. The basic expressions describing this motion are given here:

The cutting speed is given as a function of the spindle speed, N, and tool diameter, D, by the relationship

$$v = \Pi DN \tag{1.6}$$

The feed speed, vf, and the feed per revolution, f , are related by

$$f = \frac{v_f}{N} \tag{1.7}$$

From these equations it is observed that the tool path is longer in up milling than in down milling. The uncut chip areas are the same for up milling and down milling, but the average chip thickness in down milling is greater. For a given depth of cut, ae, the maximum uncut chip thickness for up milling is smaller than that for down milling. This explains the higher requirement of cutting power and the higher cutting forces associated with down milling. For small feed speeds as compared to the spindle speed, the torchoidal path can be approximated by a circular arc and the uncut chip geometry for up and down milling becomes approximately the same. [Sheikh-Ahmad, 2009]

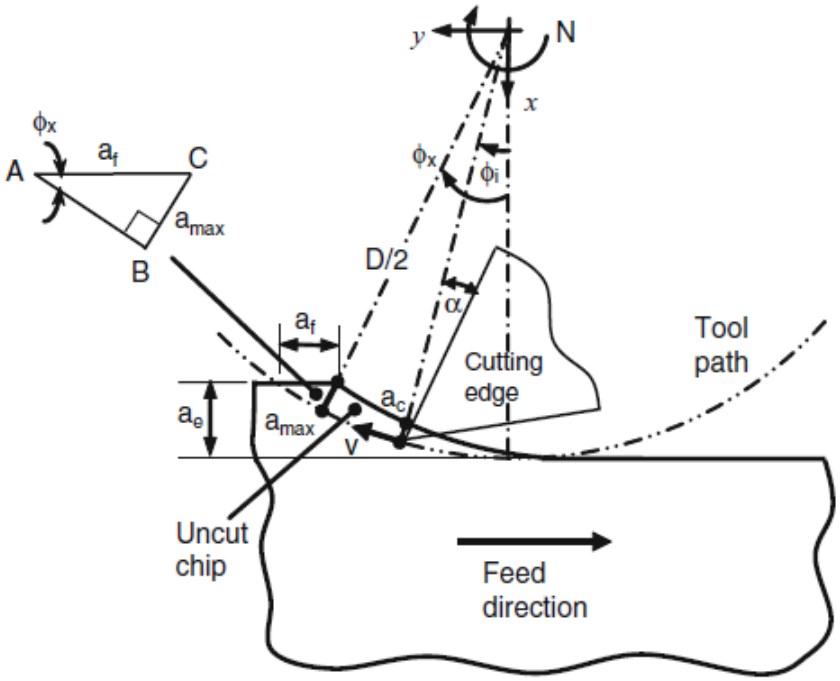


Figure 1.9: Cutting geometry in end milling, peripheral milling and edge trimming

1.5 Conclusion

This chapter delivers a comprehensive overview of surface quality, underscoring the paramount importance of attaining exact dimensions and impeccable surface finishes within the realm of mechanical parts manufacturing. By shedding light on the significance of these aspects, we layed the foundation for understanding the intricate interplay between machining processes and the resulting product quality. Through exploring various dimensions of surface quality, from its fundamental importance to the challenges it poses, this study sets the stage for deeper exploration into methodologies aimed at optimizing machining operations and enhancing the quality of manufactured components.

we conclude that the prediction of surface roughness is essential for producing high quality products at low cost and in short time.

Chapter 2

Surface roughness prediction: State of art

2.1 Introduction

Engineers often face two main problems when making things in factories. First, they need to understand the best settings of the manufacturing process to achieve the desired quality product. Next, they want to make sure that the plant is running as efficiently as possible with the resources they have. Engineers decide how to do this based on what they know and what is typically done in similar situations. But in the manufacture of metal parts, for example, there are a lot of complicated things going on that make it difficult to get a job done right. So, researchers are creating models to try to better understand these processes. They use these models to understand how different factors affect the final product. With the improvement of technology, as with computer-controlled machines, new ...There is no defined classification of approaches mainly for two reasons: First, there are many papers that do not strictly follow a certain methodology in its entirety, they rather select some of its basic principles and combine them into a 'new' approach. Secondly, there are many cases where researchers blend different strategies into a single approach and therefore no single classification would be entirely accurate.[Benardos and Vasniakos,]

Taking in consideration the above we created three major categories to classify the approaches.

1. Machining theory based approaches.

2. Designed experiments based approaches.

3. Artificial intelligence based approaches.

2.2 Machining theory based approaches

This approach studies from various literature conclude that it simulates the cutting process in termof kinematics, cutting tool properties and Computer-aided design (CAD). Vibration parameter included in an attempt to more accurately depict the phenomenon and the obtained results can be good, but this approach not considered the other factor (wear and thermal phenomena) that contribute to the roughness formation mechanism. The integration of this factor increases the accuracy in case of finishing procedures.[Bansal and Ghangas, 2013]

The theoretical background used by the research efforts in this category is considered a prerequisite for anyone who is involved in machining studies and therefore no analytical description is presented.

2.2.1 Studies

- In one of the early studies [Ehmann and Hong, 1994] introduced a new method to represent the surface generation process, which they called 'surface-shaping system'. Their system basically consisted of two parts, one that modeled the machine tool kinematics and another that modeled the cutting tool geometry. For the latter, specific interest was given in the area of the cutting edge that was described as the intersection of the tool's face and flank surfaces along with the respective angles. In general, the surface-shaping system could account for spindle runout and machine vibrations while additional research for the estimation of cutting forces was still underway. In the work the system was applied for the simulation of the 3D topography of a peripherally milled surface.

- The work of [Lee et al., 2001] emphasizes on surface roughness and profile in high-speed end milling. A method for simulating the machined surface was presented using the acceleration signal instead of the cutting forces. The argument provided was that the vibration, which is caused by the high speed of the spindle deteriorates the geometric accuracy of the machined surface. A geometric end milling model was used for modeling the end mill offset and tilt angle. The computer algorithm was developed in terms of cutting conditions, cutter and workpiece geometry, and runout parameters to determine the angular position of the end mill. The coordinates of the flute end

positions were calculated using the geometric model of the end milling process and the peak frequency components of the acceleration signal. From the generated surface data, surface roughness could be calculated, and the profile plotted.

- [Bouhadja, 2023] recent advancements have introduced a methodology for predicting the 3D topography of surfaces machined on a 5-axis machine using material removal simulation. This involves spatially dividing the workpiece into clusters of geometric discretization elements. Validated qualitatively, the method showed that the K-Means clustering method outperformed the traditional Cell Method. Additionally, the FGK method demonstrated superior performance in computation time, cluster uniformity, and reducing overlapping cluster envelopes compared to the BK and IK methods.

2.2.2 Remarks

The conclusion that can be drawn is that these, theoretical for the most part, studies simulate the cutting process in terms of kinematics and cutting tool properties. Additional parameters such as vibrations are included in an attempt to more accurately depict the phenomenon and the obtained results can be characterized as fairly good. The drawback of the approach is that a lot of other factors that contribute to the roughness formation mechanism are not considered, for example wear and deflection of the cutting tool or certain thermal phenomena. The integration of these factors to the already existing models is estimated to increase their accuracy, especially in cases of finishing procedures where their influence is greater.

2.3 Designed experiments based approaches

Although the common goal of the techniques investigated is to organize the experimental procedure and the necessary data processing, each follows a different path. This approach is come in the different category because they constitute a systematic method concerning the planning of experiments, collection and analysis of data with near optimum use of resources.[Bansal and Ghangas, 2013] Here we present two methods

2.3.1 Response surface methodology overview

In response surface methodology, the factors that are considered as most important are used to build a polynomial model in which the independent variable is the experiment's response. In order to find the global minimum of the response, experiments that 'prune' the response

surface are designed and conducted, and the gradient of the response surface is used along with a steepest ascent algorithm as follows [Garcia-Diaz and Phillips, 1995].

Stage 1:

- Select the factors to be investigated.
- Design and run a two-level factorial experiment in a localized region of the response surface.
- Compute the estimates of the effects and thereby calculate the coefficients of the linear model:

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n \tag{2.1}$$

- Select a reference factor to be used as a guide in determining the appropriate steps along the direction of each factor in order to continue moving along the path of steepest ascent.
- Select a few experimental conditions along the path of steepest ascent and run trials to determine if the response continues to increase. If the response ceases to increase, a new path should be generated.
- If a new path is needed, design and run a new two level factorial experiment. All previous steps are repeated until no substantial improvement in the response is obtained. At this point stage 2 is conducted.

Stage 2

- Design and run a three-level factorial experiment in the region where the path of the steepest ascent yields no substantial improvement in the response.
- Compute the coefficients of the model:

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_{11} X_1^2 + b_{22} X_2^2 + \cdots + b_{12} X_1 X_2 + \cdots + b_{n-1} X_{n-1} X_n \tag{2.2}$$

- Using the above model, determine the nature of the stationery point of the response surface. The stationery point is one where the gradient vanishes.

The sequential nature of RSM allows the experimenter to learn about the process or system under study as the investigation proceeds. This ensures that over the course of

the RSM application the experimenter will learn: 1/how much replication is necessary; 2/ the location of the region of the optimum; 3/the type of approximating function required; 4/the proper choice of experimental designs; and 5/ whether or not transformations on the responses or any of the process variables are required.

2.3.2 Taguchi techniques for DoE overview

DoE dictates a series of steps to follow for the experiment to yield an improved understanding of product or process performance [Ross, 1996].

- Planning phase:

1. State the problem.
2. State the objectives of the experiment.
3. Select the quality characteristics and the measurement systems.
4. Select the factors that may influence the quality characteristics.
5. Select levels for the factors.
6. Select the appropriate Taguchi fractional matrices or orthogonal arrays (OAs).
7. Select the interactions that may influence the quality characteristic.
8. Assign factors to OAs and locating interactions.

- execution phase:

9. Conduct the experiment repetitions as described by the OAs.

- Analysis phase:

10. Analyze the experimental results, e.g. using analysis of variance (ANOVA).
11. Conduct a confirmation experiment.

There are three types of OAs, dealing with two-level factors, three-level factors and mixed-level factors. The selection of the appropriate OA is based on the following criteria: the number of factors and interactions of interest, the number of levels for the factors of interest and the desired experimental resolution or cost limitations. The first two determine the

smallest OA that it is possible to use, while the third gives the possibility to conduct a larger experiment with higher resolution. Resolution can vary from 1 (lowest) to 4 (highest) and it indicates the clarity with which each individual effect of factors and interactions may be evaluated in an experiment. In order to assign the various factors to an OA's columns, the following mathematical property should be taken into account. If one factor is assigned to any particular column and a second factor to any other particular column, a specific third column will automatically have the interaction of those factors assigned to it. The pattern of which columns will be interaction columns is known for all the OAs and it is visualized through the interaction tables and linear graphs.

2.3.3 Studies

- The objective of [Davim, 2001] was to establish a correlation between cutting velocity, feed and depth of cut with the surface roughness in turning. For that purpose, a plan of experiments, based on Taguchi techniques, was designed and executed. The results showed that the cutting velocity had the greater influence, followed by the feed and that the error achieved was smaller than that of a geometric theoretical model.

- An effort to predict surface roughness in turning of high-strength steel based on RSM was made in [Choudhury and El-Baradie, 1997]. The adequacy of the developed model results was not very good, but the conclusion was that the effect of feed is much more pronounced than the effects of cutting speed and depth of cut on the surface roughness.

- [Benardos and Vosniakos, 2003] considered various methodologies for surface roughness prediction and showed manufacturing process become more productive and competitive.

- [Kumar et al., 2021] increased surface finishing of material by using VMC milling machine and M series solid carbide tool SS 304 optimization done by Taguchi method.

2.3.4 Remarks

Although the common goal of the techniques investigated is to organize the experimental procedure and the necessary data processing, each follows a different path. The RSM is mainly a model formulation procedure to investigate how important factors affect the response of an experiment and leads to the development of first- and second-order polynomial models that include the parameters under consideration and their statistical significance. These models are used to create contour plots that can be more practically utilized to draw

conclusions compared to using a polynomial function. On the other hand, the Taguchi DoE is more of a factor-screening procedure to determine the significance of each factor, that is, it identifies the most influential parameters and the values that produce the desired output without formulating any kind of model. Nevertheless, it must be pointed out that because of their generality and strong statistical background, certain tasks of these methodologies can be isolated and applied to a wide range of engineering problems where the size of the search space must be reduced. [Benardos and Vasniakos,]

2.4 Artificial intelligence based approaches

The monitoring and control of surface roughness is difficult due to non-linearity. Conventional surface roughness measurement techniques depending on stylus probe instruments, which are post processing and time consuming one. Surface roughness measurement techniques using vision technologies are difficult and costlier. Therefore, researchers have attempted the modeling of machining operations by different artificial intelligence techniques.

Artificial intelligent methods are tools that exhibit the characteristics associated with intelligence in human behavior [jm, 2005]. Since turning and milling plays a vital role in manufacturing prediction of surface roughness using Artificial intelligence in those process is considered.

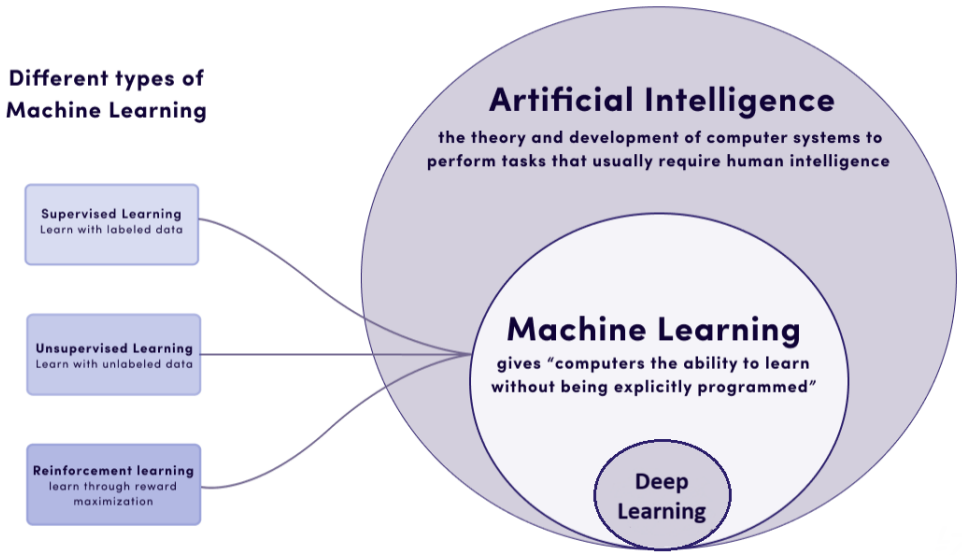


Figure 2.1: Artificial intelligence

2.4.1 Machine learning

In general, machine learning involves adaptive mechanisms that allow computers to learn from experience, to learn by example, and to learn by analogy. Learning capabilities can improve a system's performance over time. Machine learning mechanisms form the basis of adaptive systems. The most popular approaches to machine learning in machining are artificial neural networks and optimization algorithms.

Choosing the "best" ML method for surface quality prediction depends on many factors, including the available data, the application context, and the specific requirements of the problem. However, some methods have shown effectiveness in this field:

- Artificial Neural Network (ANN): ANN is a subset of machine learning and is at the heart of deep learning algorithms. They are inspired by the structure and function of the brain and are used to model complex patterns and prediction problems.
- Adaptive Neuro-Fuzzy Inference System (ANFIS): ANFIS is a kind of artificial neural network that is based on Takagi–Sugeno fuzzy inference system. It has been used for the prediction of surface roughness.
- Support Vector Machine (SVM): SVM is a powerful and flexible class of supervised algorithms for both classification and regression. It has been used for surface quality prediction.

  In terms of predicting surface roughness, SVM has been applied in various studies. For instance, one study applied SVM to develop prediction models for machining processes. Kernel function and loss function were Gaussian radial basis function and E-insensitive loss function, respectively. To improve the prediction accuracy and reduce parameter adjustment time of the SVM model, artificial bee colony algorithm (ABC) was employed to optimize internal parameters of the SVM model.

  These studies show that SVM can be effectively used for predicting surface roughness in different machining operations.[Boga and Koroglu, 2021]
- Gaussian Process Regression (GPR): GPR is a non-parametric method used in machine learning for regression problems. It has been used to predict surface roughness.

  In terms of predicting surface roughness, GPR has been applied in various studies. For instance, one study proposed a surface roughness prediction method based on

GPR, where cutting parameters and cutting forces were used as input variables. For 30CrMnSiNiA steel, high-speed dry milling (HSDM) experiments were carried out by considering spindle speed, feed per tooth, depth of cut, and width of cut as factors. A comparison with support vector machine (SVM) and artificial neural network (ANN) was conducted to verify the effectiveness and superiority of the proposed prediction method.

While some studies used ANN for surface roughness prediction, GPR could be incorporated into the model to provide a probabilistic measure of uncertainty over the predictions. This could potentially improve the robustness and reliability of the model [Adizue et al., 2023]

2.4.2 Artificial Neural Network

Solving data prediction problems has become a crucial task in everything from finance and healthcare to speech recognition and imaging. In this context, artificial neural networks (ANNs) have emerged as an approach capable of modeling the complex relationships between inputs and making accurate predictions of the outputs, the different layers of a neural network, composed of interconnected neurons inspired by human neurons, allow to capture different information and propagate it through the network.[Boga and Koroglu, 2021]

2.4.2.1 Origin of ANN

Throughout history, humanity has been intrigued by the idea of constructing machines capable of human-like thinking and reasoning. Although John McCarthy officially coined the term "artificial intelligence" in 1955, the evolution of AI and related fields began earlier. The Dartmouth Summer Research Project on Artificial Intelligence, organized by McCarthy and colleagues in 1956, witnessed the emergence of machine learning, deep learning, predictive analytics, and prescriptive analytics. Concurrently, data science emerged as a new discipline[45]. In 1960, automation engineer B. Widrow introduced the Adaptive Linear Element, which laid the foundation for the gradient backpropagation algorithm commonly used in multilayer perceptrons . First appearing in 1985, it has become fundamental in modern AI and deep learning.[Touzet, 1992]

2.4.2.2 Architecture of ANN

Artificial neural networks (ANNs) draw inspiration from the functioning of biological neurons. The human brain, with its approximately 10 billion neurons and 60 trillion synapses (connections), outperforms computers by leveraging parallel processing across multiple neurons. ANNs mirror the structure of biological neurons.

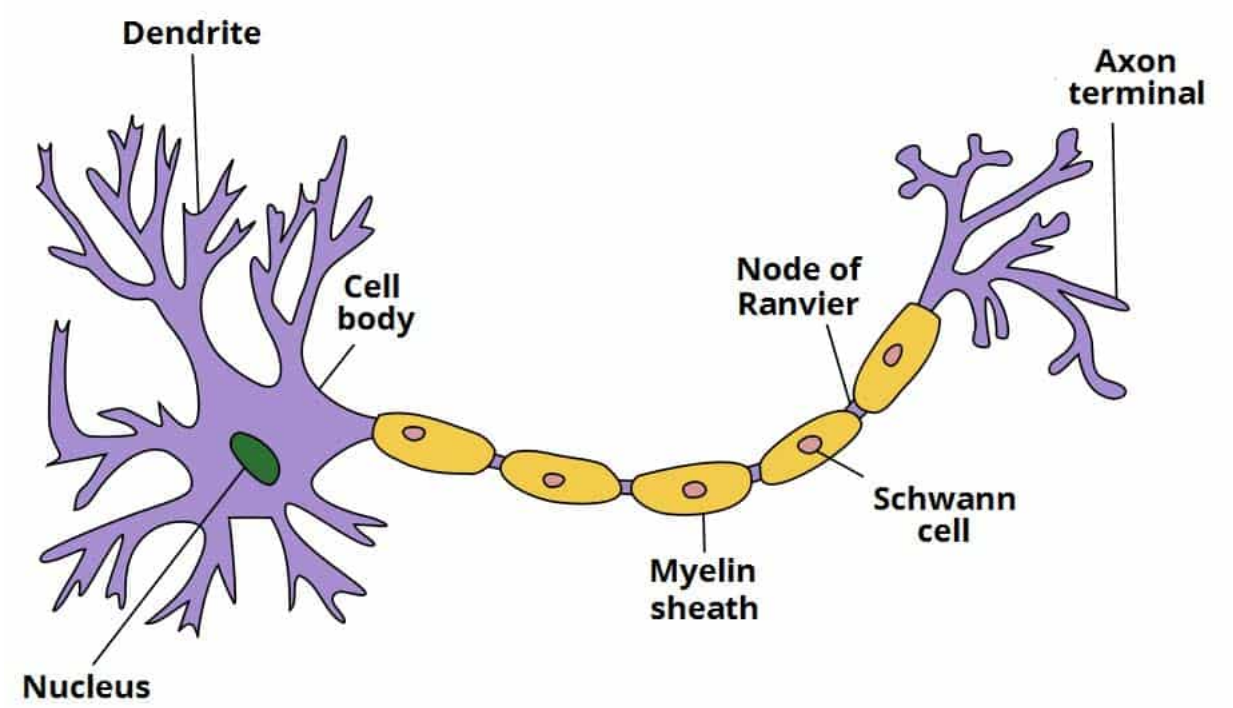


Figure 2.2: Neuron structure [Boga and Koroglu, 2021].

Synapses in a biological neuron use neurotransmitters to transmit electrical signals (Figure 2.2). Similarly, in an ANN neuron, the activation function transforms the signal produced by the processing function into a nonlinear output. Functions Commonly used activation methods include the sigmoid function, the ReLU function, and the hyperbolic tangent function. Synaptic connections in a biological neuron are strengthened or weakened based on experience, and weights in a neuron are adjusted during training to improve the Model performance. Learning in a biological neuron occurs through the adjustment of synaptic connections. In an ANN neuron, learning is done by backpropagating the error and adjusting the weights in order to optimize the model's prediction. Interesting Feature of Neural Networks Artificial is their ability to learn in a non-linear way. Unlike many other prediction techniques, neural networks can model complex relationships and non-linear between input variables, allowing for flexibility and parallel computation Data helps keep the connections

that lead to a good result remove those that lead to a bad result. In addition, artificial neural networks benefit from supervised learning, which is based on a set of data that guides the learning process. This allows the neural network to adjust its weights and minimize the error between predictions and actual values. By combining these biological concepts with mathematical and computational techniques, artificial neural networks have emerged as powerful tools in the field of data science to model and predict complex relationships between variables [Touzet, 1992].

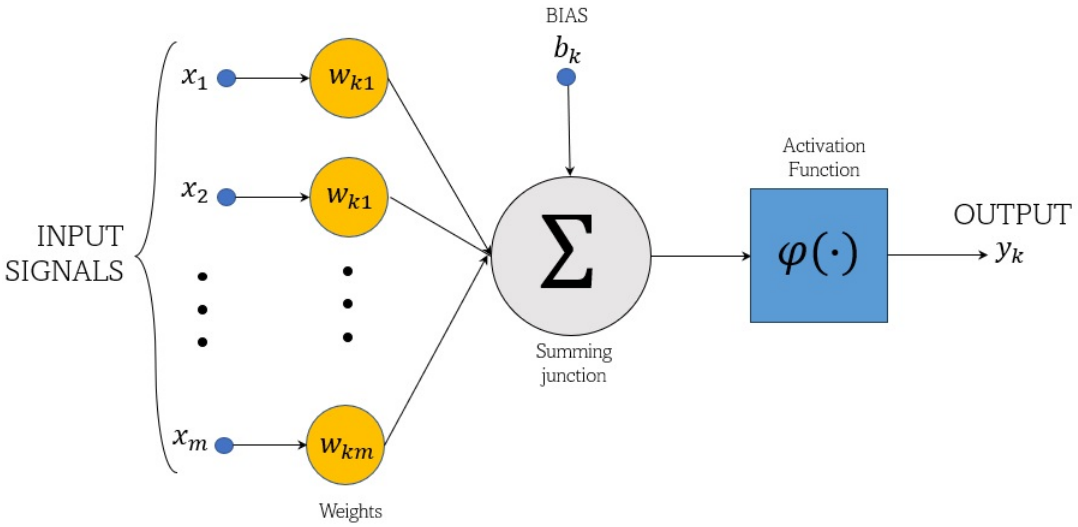


Figure 2.3: Single-layer artificial neuron or model .

A neuron receives several signals (x_i) from its input links (Figure 2.3). These signals are then multiplied by their corresponding weights, which represent the importance or strength of the connections between neurons. Additionally, a bias term is added—a constant value that introduces an offset or translation—before activating the neuron. This bias allows the neuron to better adapt to data and capture more complex nonlinear relationships.The neuron then calculates its output (y) using a specific function and sends this output to another artificial neuron. This process is the propagation of information within the neural network. During training, the network adjusts the weights and biases to make accurate predictions or solve a specific task. The retro-gradient propagation algorithm, commonly used for learning, updates the weights based on the error between network predictions and expected values.Activation functions play a crucial role—they are mathematical functions applied to the outputs of neurons. These functions introduce non-linearity into the model, allowing neural networks to approximate complex and nonlinear relationships between input variables and their corresponding outputs.

2.4.2.3 Multi-layers neural network

These deep learning models consist of multiple layers of interconnected neurons. Each layer has a specific role in processing network information.

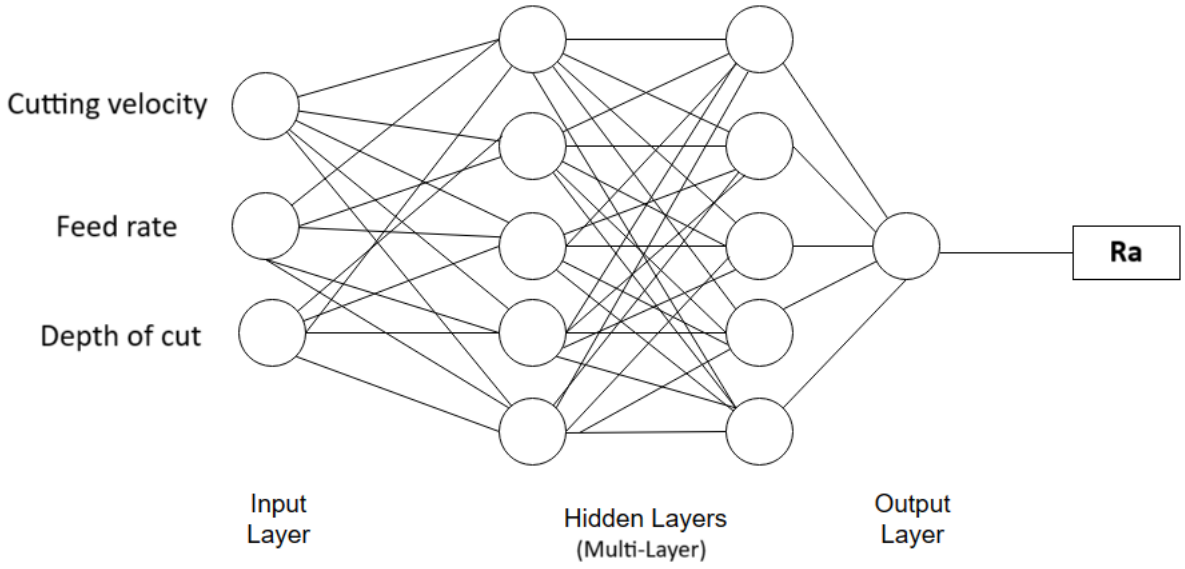


Figure 2.4: Multi-layer ANN structure.

2.4.2.3.1 Input layer:

- The first layer of the network.
- Accepts input signals (e.g., digital data from an experimental design).
- Redistributes this data to neurons in the hidden layers.
- Typically lacks computational neurons and doesn't process input patterns directly.

2.4.2.3.2 Hidden layer:

- Responsible for detecting features in input patterns.
- Each neuron in this layer is associated with weights representing hidden features.
- These features are learned from training data and used for representation and transformation.
- Output signals from the hidden layer feed into the next layer.

2.4.2.3.3 Output layer

- Receives signals from the hidden layer.
- Generates the final output information from the network.
- Used for various tasks:
  1. Classification: Predicting classes (e.g., image categories).
  2. Regression: Estimating continuous values (e.g., house prices).
  3. Other specific tasks based on the problem being addressed.

2.4.2.3.4 Complex Functions:

- Multi-layer neural networks can represent complex functions.
- A single hidden layer allows representation of any continuous function of input signals.
- Additional hidden layers enable representation of even discontinuous functions.
- This flexibility allows modeling complex nonlinear relationships between input variables.

2.4.2.4 Activation functions

Activation functions shape the outputs of artificial neurons and, therefore, are integral parts of neural networks in general and deep learning in particular. Some activation functions, such as logistic and relu, have been used for many decades.[Lederer, 2021]

See the left panel of Figure 2.5 for an illustration. The weights ω_{k1},...,ω_{km} can be interpreted as the neuron's sensitivities to the different inputs and the bias b_k as the neuron's overall sensitivity; the function φ can be interpreted as the neuron's activation pattern and, therefore, is called the activation function.

Hence, artificial neurons resemble biological neurons in the way they translate multiple input signals into a single output signal [Martin et al., 2021].

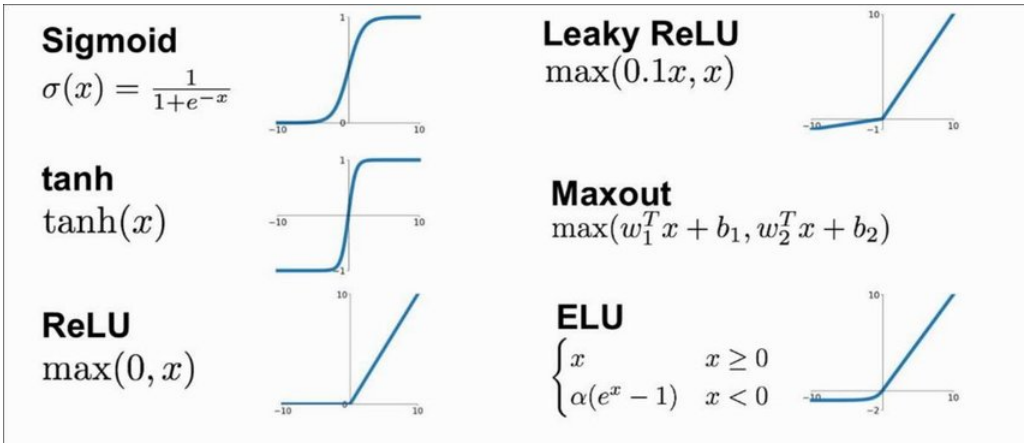


Figure 2.5: Activation functions.

Since a neural network is made of neurons, its characteristics are governed by the neurons' parameters and activation functions. The parameters are usually fitted to training data; in contrast, the activation functions are usually chosen before looking at any data and remain fixed.[Lederer, 2021]

2.4.2.4.1 Sigmoid function is a bounded and differentiable function that is nondecreasing and has exactly one inflection point. Roughly speaking, a sigmoid function is a smooth, "S-shaped" curve. Because sigmoid functions "squash" the real values into a bounded interval, they are sometimes called squashing functions. Sigmoid activation has a long-standing tradition in the theory and practice of neural networks.

$$f_{log} : R \mapsto (0, 1)$$
$$z \mapsto \frac{1}{1 + e^{-z}}$$

2.4.2.4.2 tanh is called hyperbolic tangent function. tanh is infinitely many times differentiable with first derivative. And it's very useful for tasks where negative values are relevant and the Taylor series of tanh and arctan agree up to the fourth order. Thus, we can think of tanh as a shifted and scaled version of logistic or as an approximation of arctan (Figure2.3). In particular, tanh combines two popular features of logistic and arctan: the derivative of tanh is a simple expression of the original function (Sigmoid), and it is centered around zero.

$$f_{tanh} : R \mapsto (-1, 1)$$
$$z \mapsto tanh\,[z] := \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

2.4.2.4.3 ReLU is called the positive-part function or ramp function. The positive-part function is the identity function (linear with slope 1) for positive arguments and the constant function with value zero otherwise. In electrical engineering, a rectifier is a device that converts alternating current to direct current. Similarly, the positive-part function lets positive inputs pass unaltered but cuts negative inputs, that is, it transforms negative and nonnegative inputs into nonnegative outputs. Therefore, a neuron equipped with a positive-part function as the activation function is often called a rectifier linear unit (relu), and the positive-part function itself is often called relu in the context of neural networks. A clear benefit of relu is that both the function itself and its derivatives are easy to implement and computationally inexpensive.

This result motivates using gradient-descent-type approaches in practice with the derivatives at zero (which do not exist) replaced by a fixed value between 0 and 1. relu activation can be subject to the dying-relu phenomenon, which is a version of the vanishing-gradient problem. The dying-relu phenomenon indicates a situation where many relu nodes are inactive during much of the training process.

$$f_{relu} : R \mapsto (0, \infty)$$
$$z \mapsto max\{0, z\}$$

2.4.2.4.4 Leaky ReLU The idea behind leakyrelu is to mimic relu but to avoid the dying-relu phenomenon. leakyrelu equals relu in the case a = 0; for positive parameters a, however, the functions differ for negative inputs, most notably in their derivatives.

A practical challenge inflicted by leakyrelu is choosing the parameter a. As we have just seen, a is the slope of leakyrelu for negative inputs. It is usually chosen between 0 (where leakyrelu equals relu) and 1 (where leakyrelu equals linear).

$$f_{relu,a} : R \mapsto (0, \infty)$$
$$z \mapsto max\{0, z\} + min\{0, az\}$$

2.4.2.4.5 ELU is called—in analogy with relu-exponential linear unit (elu). In fact, relu is a special case of elu. The mathematically most convenient parameter is a = 1, because this is the only choice that makes elu one time differentiable on the entire real line (but not twice differentiable). But except for this observation, there is little insight into how to choose a in practice. Observe also that the first derivatives of elu can be computed easily from the

original functions, but the original functions involve an exponential function and, therefore, are more costly to compute than relu. Hence, in view of the unclear practical benefits and the computational disadvantages of elu, relu is currently preferred.

$$f_{elu,a} : R \mapsto (-a, \infty)$$

$$z \mapsto \begin{cases} \text{z ..... if } z \geq 0 \\ \text{a(e-1) if } z < 0 \end{cases}$$

2.4.2.5 Artificial neural network learning

Training a neural network : An overview

Training a neural network involves adjusting its weights (or parameters) so that it can learn from data and make accurate predictions or solve specific tasks. The process of learning relies on two key techniques: forward propagation and backpropagation.(Figure 2.6)

- Forward Propagation: Input data is transmitted through the neural network layer by layer. Than, neurons perform a weighted sum of the received inputs using previously initialized weights. An activation function is then applied to generate output, which computes network predictions.
- Backpropagation: Gradient backpropagation is an algorithm used to adjust neural network weights. The goal is to minimize the error between network predictions and expected values. Gradients of the error with respect to the weights are calculated using the chain rule. These gradients are propagated in reverse through the network to update the weights and they are performed iteratively to gradually minimize network error.
- Training Process: Training typically involves multiple iterations (epochs) of forward and backward propagation. During each epoch, weights are adjusted to improve network performance on input data. The ultimate

aim is to obtain weights that allow the network to generalize well to new data and make accurate predictions.

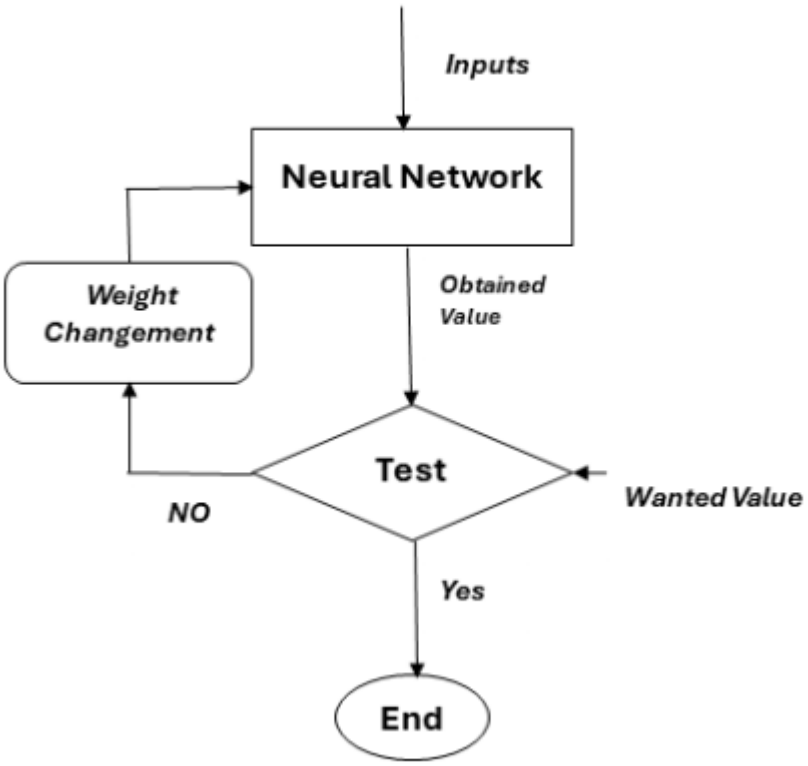


Figure 2.6: Learning algorithm for ANN

Learning Rate in Neural Networks.

During the training process of neural networks, a crucial parameter comes into play: the learning rate. This rate determines how quickly the weights of the network are updated during gradient backpropagation. Essentially, it controls the magnitude of weight adjustments in each training epoch. Mathematically, the weight update can be expressed as:

$$newweights = oldweights \ominus learningrate \times gradiant$$

Gradient: Calculated using gradient backpropagation, it Indicates both

the direction and magnitude of the steepest increase in error. Essentially guides weight adjustments.

Learning Rate determines the step size for weight updates during training. High learning rates result in larger weight adjustments, potentially leading to faster convergence but also instability. Conversely, low learning rates yield slower but more stable convergence.

Choosing the optimal learning rate depends on the specific problem and often involves empirical tuning and experimentation.

Types of Learning.

There are two fundamental types of learning in machine learning (Figure 2.1):

Supervised learning:

- Involves training a model using labeled training data.
- Each training example has an associated expected output value.
- Common tasks include classification (predicting classes) and regression (estimating continuous values).

Unsupervised learnig:

- Works with unlabeled or untargeted data.
- The model aims to discover patterns or structures within the data without prior information.
- Clustering and dimensionality reduction are typical unsupervised learning tasks.

Both supervised and unsupervised learning play essential roles in building effective machine learning models.

2.4.3 Optimization algorithms

In the realm of optimization, where finding the best solution to a problem is paramount, traditional methods often fall short when faced with complex, non-linear, or high-dimensional search spaces. Metaheuristic algorithms emerge as a powerful arsenal in tackling such optimization challenges, offering versatile and efficient approaches to finding near-optimal solutions within reasonable time frames.

Metaheuristic algorithms are a class of optimization techniques designed to efficiently explore large solution spaces to find high-quality solutions.

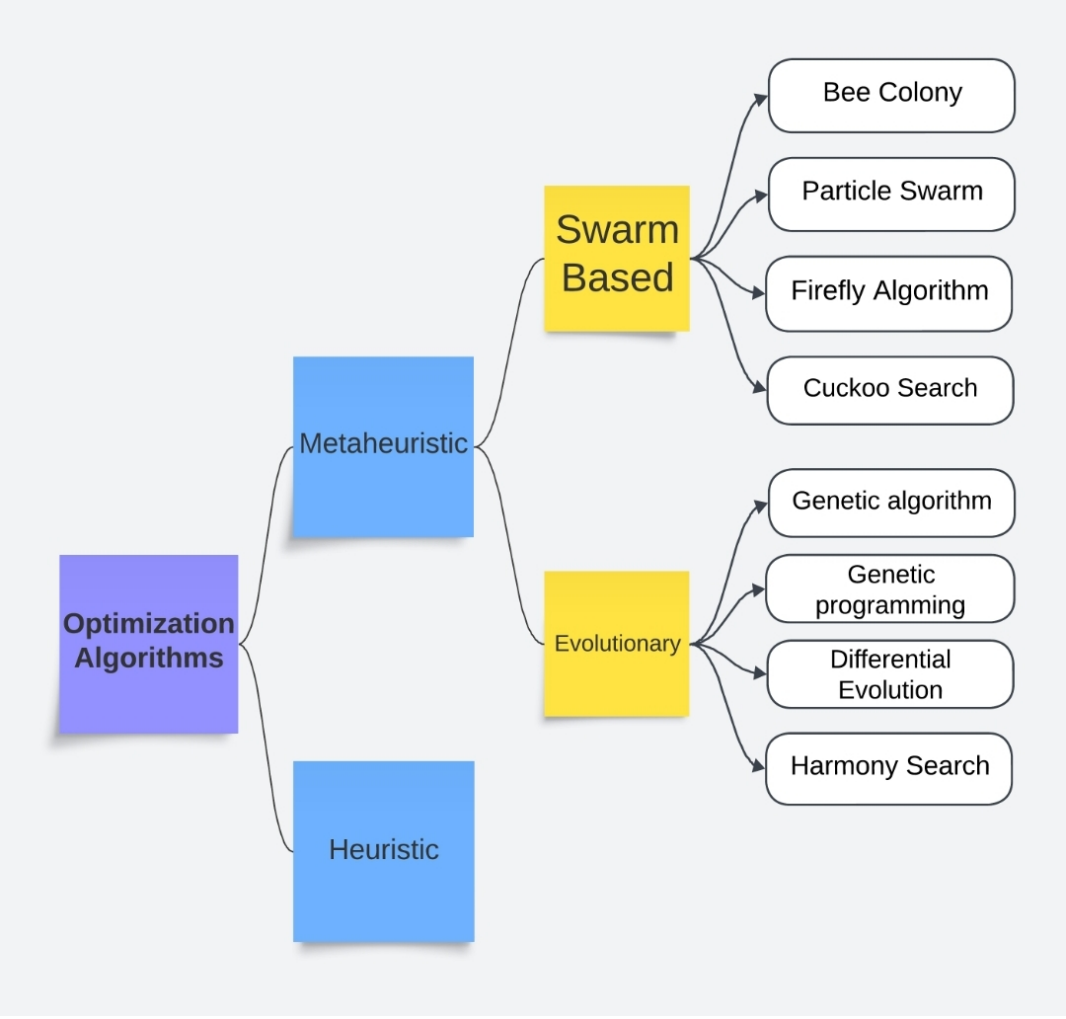


Figure 2.7: Classification of optimization Algorithms

Unlike traditional algorithms, which rely on explicit problem-specific knowledge or assumptions, metaheuristics are general-purpose, heuristic-driven methods capable of adapting to a wide range of optimization problems without requiring problem-specific modifications.

2.4.3.1 Artificial bee colony optimization (ABC)

ABC is a metaheuristic algorithm proposed by Karaboga in 2005. It is one of the most cited new generation metaheuristics in literature. ABC, being a population-based algorithm, has been applied to various optimization problems. The natural aspiration of ABC comes from the fact that candidate solutions are represented as bees exploring/exploiting food resources [Dokeroglu et al., 2019]. A solution indicates a food resource and the nectar amount of each resource represents the quality/fitness of each solution. There are three types of bees in the hive: "employed", "onlooker", and "scout" bees. In nature, employed bees look for a food source, come back to hive and share their information by dancing. When an employed bee finishes the collection of the nectar, it turns into a scout and looks for new food resources. Onlooker bees watch how the employed bees dance and choose food sources, while scout bees explore for food sources. First, a random initial population is generated. The fitness of a state of a bee colony is indicated by the acquired resources. Fig. 3 presents the basic behavior of artificial bees. A forager bee starts as an unemployed bee having no information about the food sources around the hive. An ordinary bee can be a scout bee and explore the solution space (see S in Fig. 3) or it can watch the dance of other bees and search for new food sources, R. The bee gathers the food, comes back to the hive, drops off the nectar. The bee can become a recruit nestmates (EF1), an uncommitted follower (UF), or go searching the food without recruiting after bees (EF2). The pseudocode of an ABC algorithm is given in Algorithm 1.

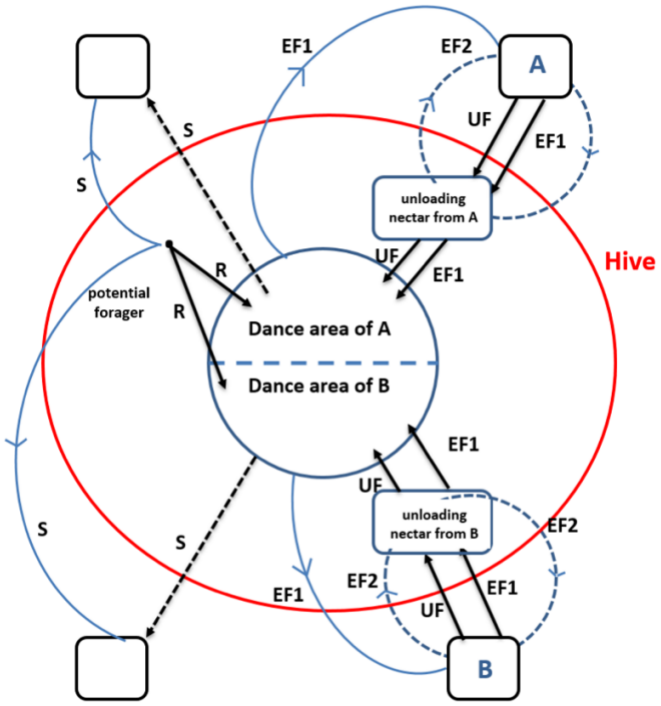


Figure 2.8: The classical behavior of honeybees looking for nectar.[Dokeroglu et al., 2019]

In order to generate an initial set of food sources, the algorithm starts randomly generating solutions in the search space. The Equation given below defines the randomized rule to produce a new solution within the range of the boundaries of the parameters. i = 1... SN j D , 1 = ... SN is the size of the food sources and D is the number of dimensions in the problem space. X_{ij} is the j-th dimension of food source i.

$$X_{ij} = X_j^{min} \times rand(0,1)(X_j^{max} - X_j^{min}) \tag{2.3}$$

Each employed bee produces a new food source (solution) depending on its local information and finds a neighboring food source. Finding a neighboring food source is defined in the Equation given below

$$V_{ij} = X_{ij} + \phi_{ij}(X_{ij} - X_{kj}) \tag{2.4}$$

A new food source V_i is produced by changing one parameter of X_i. In

the Equation above, j is a random integer between [1, D] where k is a random index different from i and Φ_{ij} is a real number between $[-1, 1]$. As the difference between X_{ij} and X_{kj} diminishes, the perturbation on solution X_{ij} becomes smaller. If a new generated parameter exceeds its boundaries, the parameter is set to acceptable values. Fitness value of the new generated solution is calculated as ($1/(1$) + f_i) when f i is positive. The fitness value is ($1/(1$ + abs f_i) if f i is positive negative.

Algorithm 1

$Int..i = 1$
while $i++ < iterations$ **do**
    Scout bees search for food ()
    Scout bees return to the hive and dance()
    Onlooker bees evaluate the food sources()
    Check previously visited food resources ()
    Decied the best food resources()
    Employed bees travel to the food sources()
    Return to hive()
    Collect the solution in the hive ()
end while

2.4.3.2 Bat algorithm (BA)

BA metaheuristic is first proposed by (Yang 2010c). Bats use echolocation (i.e., a type of sonar) to avoid obstacles, detect prey, and locate their nests in the dark. A bat emits a sound and follows the echo that reflects from the objects in the environment. Bats can also detect the difference between food/prey and barriers by using echolocation. The motivation of BA is that this echolocation talent of bats can be formalized as a means to find an optimal solution in an objective function. BA runs in an iterative fashion. Bats fly with velocity v_i with position x_i having a frequency f_{min}, varying wavelength and loudness A_0 while searching for their prey randomly. They can set the

frequency of the pulse and adjust its rate r [0, 1] (with respect to the proximity of the prey). The loudness is changed from a maximum A_0 value to a minimum value A_{min}. The frequency f in the range of [f_{min} , f_{max}] correlates with a range of wavelengths [λ_{min},λ_{max}]. While any wavelength can be used for a run of BA, selection of a suitable wavelength has significant impact on the convergence of the algorithm. In most cases, wavelength is variable during a run, and a range of wavelengths are set up and adjusted accordingly during a run. The detectable range should be decided that it is comparable to the size of the domain of interest. The frequency can be changed while fixing the wavelength and is related to . Parameter f is assumed to be between [0,f_{max}]. Higher frequencies have short wavelengths and can travel a shorter distance while lower frequencies have large wavelengths and can travel further. That is, wavelength and frequency relates to both computational cost and exploration capacity of the setting. The rate of pulse is in the range of [0, 1] where 0 means no pulse and 1 means the maximum rate of pulse emission. Algorithm 2 summarizes the execution of a sample BA run.

The positions of virtual bats x_i (i = 1, 2,. . . , n) and velocities v_i in a d-dimensional space are updated in iterations. New solutions x_i^t and velocities v_i^t at time t are generated with the equations given below.

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{2.5}$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i \tag{2.6}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{2.7}$$

where $\beta \in [0, 1]$ is a uniform distribution random vector. x is the global best location among n bats. If $\lambda_i f_i$ is the velocity increment, we can use either f_i (or λ_i) to set the new velocity while fixing the factor f_i (or λ_i). $f_{min} = 0$ and $f_{max} = 100$ can be used with respect to the domain of the problem. Each bat is assigned a random frequency. It is drawn uniformly from [f_{min},f_{max}].

Algorithm 2

Objective functions $x = (x_1, \cdots, x_d)^T$
Initialize the population of bats $x_i (i = 1, \cdots, n)$ and v_i
Define pulse frequency f_i at x_i
initialize pulse rate r_i and the loudness A_i
while t < iterations **do**
    Generate new solution be setting frequency
    Update velocity and location/solution
    if $rand > r_i$ **then**
        Select a solution
        Generate a local solution
    end if
    Generate a new solution by flying randomly
    if $rand < A_i$ **then**
        Accept the solution
        Increase r_i and reduce A_i
    end if
    Find the best x_*
end while
Report the global best result

In the local search, after selecting a solution among the best solutions, a random solution for each bat is generated locally using a random walk process using the formula:

$$x_{new} = x_{old} + A_t \tag{2.8}$$

where $[1, 1]$ is a random value, $A_t =< A_i^t >$ is the average loudness of all the bats at this period of time. The update of positions and velocities of bats are performed in a similar fashion after each iteration as in a PSO algorithm. Parameter f_t controls the pace and the range of the movement. At each iteration, the loudness A_i and the rate r_t of pulse emission need to be updated. The loudness decreases as bat finds its prey. The rate of pulse emission increases, the loudness can be of any value. The values for A_0 and A_{min} can

be assigned as 100 and 1 respectively. For simplicity, $A_i = 0$ and $A_{min} = 0$ can be applied. $A_{min} = 0$ means that a bat has found the prey and finished emitting a sound.[Dokeroglu et al., 2019]

$$A_i^{t+1} = \alpha A_i^t, ...r^{t+1} = r_i^0[1 - exp(\gamma t)] \tag{2.9}$$

where α and γ are constants.

2.4.3.3 Cuckoo search algorithm (CSA)

Yang and Deb (2009) propose CSA. The CSA simulates the brood parasitic behavior of cuckoo species with the Lévy flight action of birds and fruit flies. Cuckoo birds have an aggressive breeding attitude. They lay eggs in the nest of other birds and remove the other eggs to increase the hatching chance of their own eggs. In CSA metaheuristic, three simple rules are used. (1) One egg can be laid at a time, and cuckoo leaves its egg in a random nest; (2) Nests having high-quality eggs can survive; (3) The number of host nests is constant, and the egg can be detected by the host bird with a probability $p_a \in [0, 1]$ a . The egg can be thrown away from nests or the bird can leave the nest, and construct a new nest. The pseudocode of the CSA is presented in Algorithm 3.

When producing new solutions $x_{(t=1)}$ (i = 1, 2, . . . , n) for cuckoo bird i, a Lévy flight is realized as in (Eq 2.10)

$$x_i^{t+1} = x_i^t + \alpha \oplus Levy(\lambda) \tag{2.10}$$

where $(\alpha < 0)$ is the size of a step related to the problem. In most cases, is selected as 1. The equation above is stochastic to provide a random walk that is a process of Markov Chain with a next location that relies on the current location and the transition probability. The product $\oplus$ means entry wise multiplications. This entry wise product via Lévy flight ving structural

Algorithm 3

Optimization f(x) functions $x = (x_1, \cdots, x_d)^T$
Construct an initial population with n nests $x_i (i = 1, \cdots, n)$
while t++ < stopping criterion **do**
    Get a random cuckoo by Lévy flights
    Calculate it's fitness value F_i
    Select a nest among n (say, j) randomly
    if $F_i > F_j$ **then**
        replace j with the new solution
    end if
    A fraction (p_a) of worse nests are left
    New nests are built
    Keep the best solution/nests
    Find the current best
end while
Process result

design optimization problems. This is one of the first applications of the CSA for the shape design optimization problems.[Dokeroglu et al., 2019]

2.4.3.4 Firefly algorithm(FA)

The FA is proposed by Yang (2010a). FA is inspired by the behavior of the short and rhythmic flashing characteristics of fireflies. Two main functions of such flashes attract mating partners or warning against predators. The rhythmic flash, the rate of flashing brings sexes together. The flashing can be formulated as a function to be optimized for combinatorial algorithms. These flashing characteristics are idealized by the following rules. (1) All fireflies can attract other fireflies without any concern about their gender. (2) Attractiveness is the brightness of the firefly. Therefore, the less brighter firefly moves towards brighter ones. They decrease the attractiveness as the distance between fireflies increases. It moves randomly when there is no brighter one.

(3) The search space of the objective function affects the brightness of a firefly. Algorithm 4 depicts a typical run of FA. Two crucial issues of FA are the light intensity and formulation of the attractiveness. The brightness of the firefly decides its attractiveness where it is the encoded objective function. The brightness of a firefly (I) at a location x can be chosen as $I_x \alpha f_x$ and the attractiveness β is relative, it will be decided by other fireflies in the population. The brightness varies with the respect to the distance r_i between two fireflies. The randomization term can be implemented with other distribution methods such as Lévy flighs.[Dokeroglu et al., 2019]

Algorithm 4 Firefly Algorithm

$f(x)$ Objective functions $x = (x_1, \cdots, x_d)^T$
// n is the number of fireflies
Generate a population of fireflies x_i $(i = 1, 2, \ldots, n)$
Define coefficient γ of light absorption
while $t <$ Max generation **do**
  for $i < n$ fireflies **do**
    for $j < n$ fireflies **do**
      Light intensity I_i at x_i is determined by $f(x)$
      if $I_j > I_i$ **then**
        Move firefly i towards j in all d dimensions
      end if
      Attractiveness varies with distance γ via $\exp(-\gamma r)$
      Calculate new solutions and update intensity of light
    end for
  end for
  Report the current best solution
end while

2.4.3.5 Harmony search (HS)

HS is a metaheuristic algorithm based on musical compositions and the process of writing a composition (Geem et al., 2001). HS is proposed by Yang (2009) in 2001 and has been applied to numerous optimization problems since then. HS makes use of methods applied by musicians to create harmonic musical compositions in order to model optimization problems. In HS, a musician has three possible choices when improvising a song: (1) playing any well-known piece of music (pitches in harmony) naturally from his or her memory; (2) playing music similar to an existing piece (by adjusting the pitch); or (3) composing random harmonic notes. Geem et al. (2001) use these possible choices during the optimization $r_{accept} \in [0, 1]$ that is called harmony memory accepting rate. When the rate is too small, a few best harmonies are selected and this causes slower convergence of the HS algorithm. When the rate is too high (a value close to 1), it may not be possible to explore all the harmonies well. This can lead to wrong solutions. The parameter r_{accept} is selected between [0.7, 0.95] to prevent his problem. The pitch adjustment is the second parameter determined by the bandwidth (b_{range}) and the adjusting rate of a pitch r_{pa}. Pitch adjustment changes the frequencies and generates diversity in the HS. Linear or nonlinear adjustment is used to set the pitch value.

$$x_{new} = x_{old} + b_{range} * \epsilon \tag{2.11}$$

x_{old} is the current pitch, and x_{new} is the new solution after the adjustment of pitch. This process generates a neighboring solution to the existing solution by changing the pitch slightly. Pitch adjustment mimics like the mutation operator in evolutionary algorithms. A parameter (pitch-adjusting rate r_{pa}) can be used to co level. A small adjustment rate can slow the convergence time of HS, whereas a high adjustment rate can act as a random search process. A value between [0.1, 0.5] is observed to be a good balance for r_{pa}.

The third parameter (randomization) is used to provide diversified solutions. The randomization enables the system to explore different solutions. The randomization can direct the search to explore various different solutions to obtain the global optimal solutions. the probability of randomization is given below:

$$P_{random} = 1 - r_{accept} \tag{2.12}$$

where the probability of adjusting pitches are:

$$P_{pitch} = r_{accept} * r_{pa} \tag{2.13}$$

Algorithm 5 summarizes a typical HS.[Dokeroglu et al., 2019]

Algorithm 5

Generate initial harmonics
Introduce pitch adjusting rate (r_{pa}), pitch limits and bandwidth
Introduce harmony memory accepting rate (r_{accept})
while t < iterations **do**
    Generate harmonics by accepting the best harmonics
    Tune the pitch to get new harmonics/solutions
    if $rand > r_{accept}$ **then**
        choose a random harmonic from population
    else
        if $rand > r_{pa}$ **then**
            tune the pitch within limits randomly
        end if
        generate new harmonics randomly
    end if
end while
Find the current best harmonics

2.4.4 Particle swarm optimization algorithm (PSO)

Particle swarm optimization (PSO) algorithm is a stochastic optimization technique based on swarm, which was proposed by Eberhart and Kennedy (1995). PSO algorithm simulates animal's social behavior, including insects, herds, birds and fishes. These swarms conform a cooperative way to find food, and each member in the swarms keeps changing the search pattern according to the learning experiences of its own and other members. Main design idea of the PSO algorithm is closely related to two researches: One is evolutionary algorithm, just like evolutionary algorithm; PSO also uses a swarm mode which makes it to simultaneously search large region in the solution space of the optimized objective function. The other is artificial life, namely it studies the artificial systems with life characteristics. In studying the behavior of social animals with the artificial life theory, for how to construct the swarm artificial life systems with cooperative behavior by computer, Millonas proposed five basic principles (van den Bergh 2001):

1. Proximity: the swarm should be able to carry out simple space and time computations.

2. Quality: the swarm should be able to sense the quality change in the environment and response it.

3. Diverse response: the swarm should not limit its way to get the resources in a narrow scope.

4. Stability: the swarm should not change its behavior mode with every environmental change.

5. Adaptability: the swarm should change its behavior mode when this change is worthy.

Note that the fourth principle and the fifth one are the opposite sides of the same coin. These five principles include the main characteristics of the artificial life systems, and they have become guiding principles to establish the swarm artificial life system. In PSO, particles can update their positions and velocities according to the environment change, namely it meets the requirements of proximity and quality. In addition, the swarm in PSO does not limit its movement but continuously search the optimal solution in the possible solution space. Particles in PSO can keep their stable movement in the search space, while change their movement mode to adapt the change in the environment. So particle swarm systems meet the above five principles.[Wang et al., 2018]

2.4.4.1 Concept of PSO

The simplest model can be depicted as follows. Each individual of the birds is represented by a point in the Cartesian coordinate system, randomly assigned with initial velocity and position. Then run the program in accordance with "the nearest proximity velocity match rule," so that one individual has the same speed as its nearest neighbor. With the iteration going on in the same way, all the points will have the same velocity quickly. As this model is too simple and far away from the real cases, a random variable is added to the speed item. That is to say, at each iteration, aside from meeting "the nearest proximity velocity match," each speed will be added with a random variable, which makes the total simulation to approach the real case.

First, we assume that position coordinate of the cornfield is (x_0, y_0), and position coordinate and velocity coordinate of individual bird are (x,y) and (v_x, v_y), respectively. Distance between the current position and cornfield is used to measure the performance of the current position and speed. The closer the distance to the "cornfield", the better the performance, on the contrary,

the performance is worse. Assume that each bird has the memory ability and can memorize the best position it ever reached, denoted as local best (p_{best}). a is velocity adjusting constant, rand denotes a random number in [0,1], change in the velocity item can be set according to the following rules:

$$if x > p_{best}x, v_x = v_x - rand \times a, otherwise, v_x = v_x + rand \times a$$

$$if y > p_{best}y, v_y = v_y - rand \times a, otherwise, v_y = v_y + rand \times a$$

Then assume that the swarm can communicate in some way, and each individual is able to know and memorize the best location global best (marked as g_{best}) of the total swarm so far. And b is the velocity adjusting constant; then, after the velocity item was adjusted according to above rules, it must also update according to the following rules:

$$if x > g_{best}x, v_x = v_x - rand \times b, otherwise, v_x = v_x + rand \times b$$

$$if y > g_{best}y, v_y = v_y - rand \times a, otherwise, v_y = v_y + rand \times b$$

Computer simulation results show that when a/b is relatively large, all individuals will gather to the "cornfield" quickly; on the contrary, if a/b is small, the particles will gather around the "cornfield" unsteadily and slowly. Through this simple simulation, it can be found that the swarm can find the optimal point quickly. Inspired by this model, Kennedy and Eberhart devised an evolutionary optimization algorithm, after a sea of trials and errors, they finally fixed the basic algorithm as follows:

$$v_x = v_x + 2 * rand * (p_{best}x - x) + 2 * rand * (g_{best}x - x) \tag{2.14}$$

They abstracted each individual to be a particle without mass and volume, with only velocity and position, so they called this algorithm "particle swarm optimization algorithm."

On this basis, PSO algorithm can be summarized as follows: PSO algorithm is a kind of searching process based on swarm, in which each individual is called a particle defined as a potential solution of the optimized problem in D-dimensional search space, and it can memorize the optimal position of the swarm and that of its own, as well as the velocity. In each generation, the particles information is combined together to adjust the velocity of each dimension, which is used to compute the new position of the particle. Particles change their states constantly in the multi-dimensional search space, until they reach balance or optimal state, or beyond the calculating limits. Unique connection among different dimensions of the problem space is introduced via the objective functions. Many empirical evidences have showed that this algorithm is an effective optimization tool. Flowchart of the PSO algorithm is shown in (Fig 2.9).

The following gives a relatively complete presentation of the PSO algorithm. In the continuous space coordinate system, mathematically, the PSO can be described as follows. Assume that swarm size is N, each particle's position vector in D-dimensional space is $X_i = (x_{i1}, x_{i2}, \cdots, x_{id}, x_{iD})$, velocity vector is $V_i = (v_{i1}, v_{i2}, \cdots, v_{id}, v_{iD})$, individual's optimal position (i.e., the optimal position that the particle has experienced) is $P_i = (p_{i1}, p_{i2}, \cdots, p_{id}, p_{iD})$, swarm's optimal position (i.e., the optimal position that any individual in this swarm has experienced) is represented as $P_g = (p_{g1}, p_{g2}, \cdots, p_{gd}, p_{gD}))$. Without loss of generality, taking the minimizing problem as the example, in the initial version of the PSO algorithm, update formula of the individual's optimal position is :

$$P_{i,t+1}^{d} = \begin{cases} X_{i,t+1}^{d} & \text{if } ..X_{i,t+1} < f(P_{i,t}) \\ P_{i,t}^{d} \quad Otherwise & \end{cases}$$

The swarm's optimal position is that of all the individual's optimal positions.

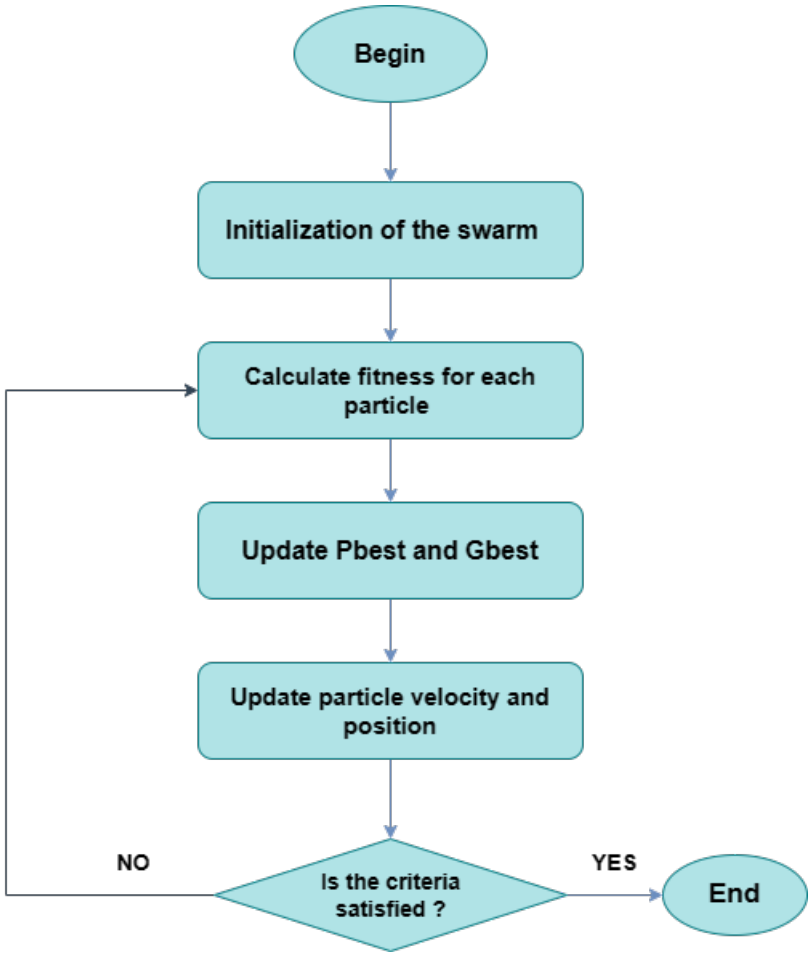


Figure 2.9: Flowchart of the particle swarm optimization algorithm

Update formula of velocity and position is denoted as follows, respectively:

$$v_{i,t+1}^{d} = v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{g,t}^{d} - x_{i,t}^{d}) \tag{2.15}$$

$$x_{i,t+1}^{d} = x_{i,t}^{d} + v_{i,t+1}^{d} \tag{2.16}$$

Since the initial version of PSO was not very effective in optimization problem, a modified PSO algorithm appeared soon after the initial algorithm was proposed. Inertia weight was introduced to the velocity update formula, and the new velocity update formula became:

$$v_{i,t+1}^{d} = \omega * v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{g,t}^{d} - x_{i,t}^{d}) \tag{2.17}$$

Although this modified algorithm has almost the same complexity as the initial version, it has greatly improved the algorithm performance; therefore,

it has achieved extensive applications. Generally, the modified algorithm is called canonical PSO algorithm, and the initial version is called original PSO algorithm.

PSO algorithm has two versions, called global version and local version, respectively. In the global version, two extremes that the particles track are the optimal position p_{best} of its own and the optimal position g_{best} of the swarm. Accordingly, in local version, aside from tracking its own optimal position p_{best}, the particle does not track the swarm optimal position g_{best}, instead it tracks all particles' optimal position n_{best} in its topology neighborhood. For the local version, the velocity update equation (5) became:

$$v_{i,t+1}^{d} = \omega * v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{l,t}^{d} - x_{i,t}^{d}) \qquad (2.18)$$

where p_l was the optimal position in the local neighborhood.and rand is [0 1] In each generation, iteration procedure of any particle is illustrated in (Fig 2.10). Analyzing the velocity update formula from a sociological perspective, we can see that in this update formula, the first part is the influence of the particle's previous velocity. It means that the particle has confidence on its

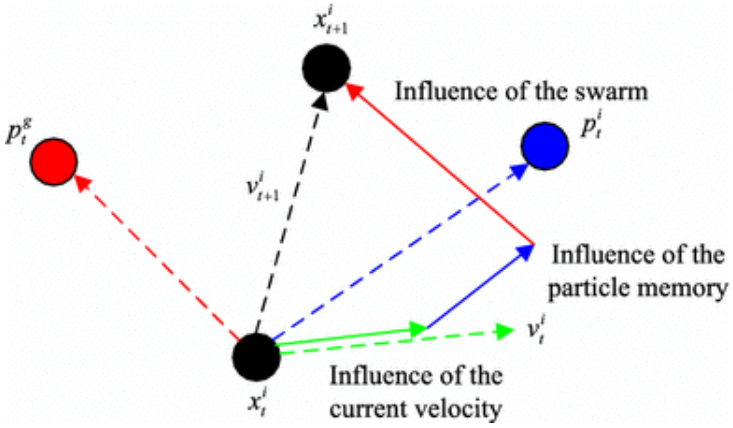


Figure 2.10: Iteration scheme of the particles

current moving state and conducts inertial moving according to its own velocity, so parameter ω is called inertia weight. The second part depends

on the distance between the particle's current position and its own optimal position, called the "cognitive" item. It means particle's own thinking, i.e., particle's move resulting from its own experience. Therefore, parameter c1 is called cognitive learning factor (also called cognitive acceleration factor). The third part relies on the distance between the particle's current position and the global (or local) optimal position in the swarm, called "social" factor. It means the information share and cooperation among the particles, namely particle's moving coming from other particles' experience in the swarm. It simulates the move of good particle through the cognition, so the parameter c2 is called social learning factor (also called social acceleration factor) [Wang et al., 2018].

2.4.4.2 Advantages and challenges of PSO

PSO is a stochastic and parallel optimization algorithm. Its advantages can be summarized as follows: It does not require the optimized functions differential, derivative and continuous; its convergence rate is fast; and the algorithm is simple and easy to execute through programming. Unfortunately, it also has some disadvantages (Wang 2012): (1) For the functions with multiple local extremes, it probably falls into the local extreme and cannot get correct result. Two reasons result in this phenomenon: One is the characteristics of the optimized functions and the other is the particles' diversity disappearing quickly, causing premature convergence. These two factors are usually inextricably intertwined. (2) Due to lack of cooperation of good search methods, PSO algorithm cannot get satisfactory results. The reason is that the PSO algorithm does not sufficiently use the information obtained in the calculation procedure. During each iteration, instead it only uses the information of the swarm optima and individual optima. (3) Though PSO algorithm provides the possibility of global search, it cannot guarantee convergence to the global

optima. (4) PSO algorithm is a meta-heuristic bionic optimization algorithm, and there is no rigorous theory foundation so far. It is designed only through simplifying and simulating the search phenomenon of some swarms, but it neither explains why this algorithm is effective from the principle, nor specifies its applicable range. Therefore, PSO algorithm is generally suitable for a class of optimization problems which are high dimensional and need not to get very accurate solutions.[Wang et al., 2018]

2.4.5 Studies

- [Benardos and Vosniakos, 2003]conducted a survey of surface roughness prediction models developed and factors influencing surface roughness. They found that the most promising seem to be the theoretical and the AI approaches.
- [Lee et al., 2004] developed an Accurate modeling and prediction of surface roughness by computer vision in turning operations using an adaptive neuro-fuzzy inference system.
- [Oktem et al., 2006] worked on optimization of cutting parameters in the machining of mold surfaces of an ortez part used in biomedical instruments. They optimized the parameters with the help of genetic algorithm coupled with neural network. A feed forward neural network was developed and trained as well as tested in MATLAB. They also validate their predicted results with experimental result by additional measurement and found the very good agreement between them. They introduced machining tolerance as new cutting parameters in the experiment. They found that ANN along with GA gave more satisfactory results than ANN. Percent of error obtained was less than 1.33%.
- [Khorasani et al., 2012] carried out various investigation for offline and

online parameters estimation. Discussed technique was AI, ANN and knowledge based expert system. Parameters were divided into six numbers as properties of tool, work piece, tool of machine, thermal and dynamic parameter and cutting.

- [D. R. M. Rajesh, 2014] worked on prediction of surface roughness of free form surfaces by use of artificial neural network. Freeform surfaces were formed using CNC machine. Such type of surfaces had large application in aerospace field. So, to maintain surface quality was very important task for aerospace industry. They used CNC ball end milling machine for development of freeform surfaces using speed, depth of cut, feed rate and step over as input parameters for machining. They used back propagation neural network to minimize error in prediction of surface roughness. 96.37% accuracy was achieved using this ANN based predictive model.

- [Moghri et al., 2014] asserted an integrated ANN-genetic algorithm (GA) approach for modeling and optimizing surface roughness of polyamide-6/nanoclay (PA-6/NC) nanocomposites. A surface roughness predictive model was developed by considering milling parameters (spindle speed and feed rate) and nano clay (NC) content. It was inferred that the minimum surface roughness could be achieved at the lowest level of feed rate and intermediate level of spindle speed.

- [Eser et al., 2021] Conducted a study improves surface roughness prediction models for milling AA6061 alloy with carbide cutting tools coated with CVD-TiCN in dry conditions. It refines an experimental model using ANN and RSM, with cutting parameters as inputs. Backpropagation is chosen for ANN training, with 3-8-1 network structure found optimal. RSM analysis highlights depth of cut as most influential. Comparison of ANN and RSM models shows RSM's higher stability and robustness,

indicated by its superior R^2 value.

- [Benallou and Benchikh, 2022] explored the impact of various machining parameters on surface roughness. The study introduces artificial neural network (ANN) and hybrid ANN with genetic algorithm (GA) for comparing surface roughness estimation. Experimental data from 84 tests on Aluminum milling were used to construct the ANN database. Machining parameter influences were analyzed using Pearson's correlation, showing feed rate (Vy) and radial depth of cut (feed Vx) as most significant. The ANN model demonstrated good prediction accuracy with mean square error (MSE) of 0.1985, indicating close agreement with measured values. The hybrid ANN-GA model showed slightly higher MSE at 0.3930.

- In his end-of-study project, [Oulmane, 2023], to examine milling optimization to improve surface quality using the Taguchi-RSM method and artificial neural networks (ANNs) to predict surface roughness based on cutting parameters. ANOVA assessed the prediction error and the influence of the factors. He concluded that the combination of Taguchi-RSM and ANN can improve the quality of machined surfaces.

2.5 Conclusion

From the above literature surveys, it seems the artificial intelligent are the most appropriate solution for the quick and precise predictive model. Most of the researchers have worked on fuzzy logic system, ANN and ANFIS for prediction and genetic algorithm for optimization of machining parameter in multi-axis machining. Some of them also suggested to couple two AI technique like fuzzy and ANN or ANN and GA to get most précised and optimized predicted result. Models obtained from coupling gave better result than in-

dividual model like ANFIS. Result obtained ANN based predictive model always varies. They depend upon number of layers and number of nodes on intermediate layer, so optimum number of layer and intermediate node is very essential for good prediction, and this is obtained by trial and error method.

we conclude that the most common parameter that are in use are speed, feed rate and depth of cut, but the most significant parameter is feed rate followed by cutting speed and depth of cut. Other process parameters are less in use as compared to these.

Chapter 3

Implementation and result

3.1 Introduction

In this study, we developed an ANN-PSO module to optimize neural network weights and biases. The module integrates PSO as an optimization algorithm during the training phase of the ANN. Specifically, PSO iteratively adjusts particle positions to find optimal weight configurations, resulting in an improved ANN model. Our approach demonstrates the effectiveness of combining ANN and PSO for enhanced performance.

3.2 Training neural networks with PSO

Artificial Neural Networks (ANNs) have emerged as powerful tools for solving complex problems across various domains, including classification, regression, pattern recognition, and optimization. However, training ANNs often involves finding optimal sets of weights and biases, which can be a challenging task due to the high-dimensional and non-convex nature of the optimization problem.

One popular metaheuristic optimization algorithm used to train ANNs is Particle Swarm Optimization (PSO). PSO is inspired by the social behavior of bird flocking or fish schooling, where individuals (particles) in the swarm

collectively search for the optimal solution by iteratively updating their positions based on their own experience and the experiences of their neighbors.

In this application, we will walk through the process of building an ANN-PSO module using Python and libraries such as NumPy for numerical computations, Matplotlib for visualization, and PySwarms for implementing the PSO algorithm.

The goal of this project is to develop a module that utilizes the synergy between ANN and PSO to train a model for predicting roughness based on machining parameters such as speed, feed rate, and depth of cut. We will follow the steps outlined below to implement our ANN-PSO module:

1. Load Data: We will start by loading the dataset containing machining parameters and corresponding roughness values from a CSV file.

2. Feature Extraction: Next, we will extract the relevant features (speed, feed rate, radial depth, axial depth) and the target variable (roughness) from the dataset.

3. Forward Propagation Function: We will define a function for performing forward propagation in the neural network. This function will take the particle's position as input, unroll the parameters (weights and biases), perform forward propagation to compute the predicted roughness, and calculate the mean squared error loss between the predicted and actual roughness.

4. PSO Optimization: We will define a higher-level function for PSO optimization. This function will iterate over each particle's position, compute the forward propagation loss, and return an array of losses for all particles. We will initialize the PSO parameters such as cognitive and social parameters, inertia weight, dimensions based on the neural network

architecture, and the number of particles for optimization. Then, we will perform PSO optimization to obtain the optimized cost and particle positions.

5. Prediction Function: We will define a function for making predictions using the trained model. This function will unroll the optimized particle positions to obtain the weights and biases, perform forward propagation on the input features to predict roughness, and return the predicted roughness values.

6. Model Evaluation: We will create a DataFrame to compare the actual and predicted roughness values. This will allow us to evaluate the performance of our trained model.

7. Visualization: Finally, we will plot the actual and predicted roughness values to visually compare their distributions. This will provide insights into how well our model is performing.

By following these steps, we aim to build an efficient and accurate ANN-PSO module for predicting roughness in machining processes.

3.3 Building the ANN-PSO model

In this project, we utilized Python 3.12 as the programming language and Visual Studio Code (VS Code) as the integrated development environment (IDE) to build our ANN-PSO model for predicting roughness in machining processes. Python is a versatile and widely-used programming language known for its simplicity and readability, making it an excellent choice for developing machine learning and data analysis applications. With the release of Python 3.12, we leveraged the latest features and enhancements to ensure optimal performance and compatibility with modern libraries and frameworks.

Visual Studio Code (VS Code) is a lightweight yet powerful code editor developed by Microsoft. It provides a rich set of features such as syntax highlighting, code completion, debugging, and version control integration, making it ideal for software development across various platforms. With its intuitive user interface and extensive ecosystem of extensions, VS Code offers a seamless and productive environment for writing, testing, and debugging Python code.

A laptop was used for the developing of this model with the following specs:

- Processor : 13th Gen Intel(R) Core(TM) i5-1335U 1.30 GHz
- RAM : 8.00 Go
- Type of system : System 64 bits, processor x64
- Windows: windows 11

3.3.1 Loading Data

First, we acquired the data from (Ugochukwu et all 2015) research paper, uesed an online tool to convert from a pdf file to csv file and than we used Google Sheets (online version) to modify it to be compatible with our model (Fig 3.1).

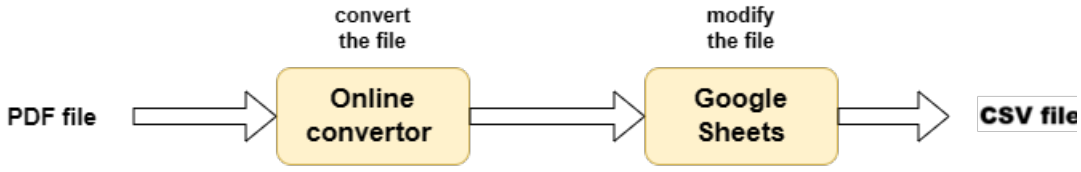


Figure 3.1: the chart of acquiring the data

secondly, we'll load the dataset using (pandas) package. It has 29 samples making it a very balanced dataset. Each sample is characterized by five

features (or dimensions): Spindle speed, Feed rate, Axial depth of cut, Radial depth of cut, Surface roughness.

```
import pandas as pd
# Load data from CSV
data = pd.read_csv('File_path.csv')
```

3.3.2 Extracting features

Here we will extract the five features and store them as x and y.

```
# Extract features and target
X = data[['speed', 'feed', 'radial-depth', 'axial-depth']].values
y = data['surface-roughness'].values
```

3.3.3 Neural network architecture

For now, we'll build a simple neural network with the following characteristics: * Input layer size: 4 * Hidden layer size: 25 (activation: tanh(x)) * Output layer size: 1 (activation: softmax(x)) . The intial number of hidden neuron was choosed from the previous studies in chapter 2 than it was optimized through trial and error methode (discussed later).

Things we'll do: 1. Create a forward_prop method that will do forward propagation for one particle. 2. Create an overhead objective function f() that will compute forward_prop() for the whole swarm.

What we'll be doing then is to create a swarm with a number of dimensions equal to the weights and biases. We will unroll these parameters into an n-dimensional array, and have each particle take on different values. Thus, each particle represents a candidate neural network with its own weights and bias.

When feeding back to the network, we will reconstruct the learned weights and biases.

When rolling-back the parameters into weights and biases, it is useful to recall the shape and bias matrices: Shape of input-to-hidden weight matrix: (4, 25). Shape of input-to-hidden bias array: (25). Shape of hidden-to-output weight matrix: (25, 1). Shape of hidden-to-output bias array: (1)

By unrolling them together, we have $(4 * 25) + (25 * 1) + 25 + 1 = 151$ parameters, or 151 dimensions for each particle in the swarm.

The negative log-likelihood will be used to compute for the error between the ground-truth values and the predictions. Also, because PSO doesn't rely on the gradients, we'll not be performing backpropagation

```
# Forward propagation function
def forward_prop(params):
"""Forward propagation function"""
# Neural network architecture
n_inputs = 4 # Number of input features
n_hidden = 25 # Modified to 25 n_classes = 1 # Regression task, so
only one output


# Unroll parameters
W1 = params[0:100].reshape((n_inputs, n_hidden)) # Adjusted
dimensions
b1 = params[100:125].reshape((n_hidden,)) # Adjusted dimensions
W2 = params[125:150].reshape((n_hidden, n_classes)) # Adjusted
dimensions
b2 = params[150:].reshape((n_classes,)) # Adjusted dimensions
```

```
# Perform forward propagation
z1 = X.dot(W1) + b1
a1 = np.tanh(z1)
z2 = a1.dot(W2) + b2
y_pred = z2.flatten() # Flatten to get a 1D array

# Compute mean squared error loss
loss = np.mean((y_pred - y) ** 2)
return loss
```

Now that we have a method to do forward propagation for one particle (or for one set of dimensions), we can then create a higher-level method to compute forward_prop() to the whole swarm:

```
# Higher-level method for PSO optimization def f(x):
"""Higher-level method to do forward_prop in the whole swarm"""
n_particles = x.shape[0]
j = [forward_prop(x[i]) for i in range(n_particles)]
return np.array(j)
```

3.3.4 PSO optimization

Now that everything has been set-up,using the (pyswarm) package we just call our global-best PSO and run the optimizer. Initially, we just set the PSO parameters arbitrarily, than we optimized them through trials (discussed later)

```
import pyswarms as ps
# Initialize PSO
options = 'c1': 1.5, 'c2': 1.5, 'w': 0.49 # Adjusted parameters
dimensions = (4 * 25) + (25 * 1) + 25 + 1 # Number of dimensions adjusted accordingly
optimizer = ps.single.GlobalBestPSO(n_particles=45, dimensions=dimensions, options=options) # Adjusted number of particles

# Perform optimization
cost, pos = optimizer.optimize(f, iters=45, verbose=3) # Adjusted number of iterations
```

We can then check the accuracy by performing forward propagation once again to create a set of predictions. Then it's only a simple matter of matching which one's correct or not. For the logits, we take the argmax. Recall that the softmax function returns probabilities where the whole vector sums to 1. We just take the one with the highest probability then treat it as the network's prediction.

Moreover, we let the best position vector found by the swarm be the weight and bias parameters of the network.

3.3.5 Predictoin function

Upon completing the optimization process with PSO, we obtain the optimized particle positions that encapsulate the learned parameters (weights and biases) of the artificial neural network (ANN). Leveraging these optimized positions, our prediction function can effectively estimate the roughness of machining processes based on input features.

- The first step in the prediction function involves unrolling the optimized

particle positions obtained from the PSO optimization process. These positions encode the optimized values of weights and biases across the neural network layers.

- With the unrolled particle positions representing the learned parameters, we proceed to perform forward propagation on the input features. This entails feeding the input features through the neural network architecture, computing the activations at each layer, and ultimately generating predictions for roughness.
- After executing forward propagation, the prediction function produces the predicted roughness values as its output. These predicted values represent the model's estimations of roughness for the given input features.

By encapsulating the process of unrolling optimized particle positions, conducting forward propagation, and returning predicted roughness values, our prediction function streamlines the process of utilizing the trained ANN-PSO model for making predictions. This function serves as a vital tool for applying our predictive model to real-world machining scenarios, enabling informed decision-making and optimization of machining processes.

```
# Predict function
def predict(X, pos):
"""Use the trained weights to perform predictions"""
n_inputs = 4
n_hidden = 25
n_classes = 1

W1 = pos[0:100].reshape((n_inputs, n_hidden))
b1 = pos[100:125].reshape((n_hidden,))
```

```
W2 = pos[125:150].reshape((n_hidden, n_classes))
b2 = pos[150:].reshape((n_classes,))

z1 = X.dot(W1) + b1
a1 = np.tanh(z1)
z2 = a1.dot(W2) + b2
y_pred = z2.flatten()

return y_pred

# Make predictions
predicted_roughness = predict(X, pos)
```

3.3.6 Evaluating the model

By calling the (pandas) package we will create a DataFrame to compare the actual and predicted roughness values. This will allow us to evaluate the performance of our trained model.

```
# Create a DataFrame to compare actual and predicted roughness
comparison_df = pd.DataFrame('Actual Roughness': y, 'Predicted
Roughness': predicted_roughness)
```

3.3.7 Displaying the result

Ultimately, we'll generate visualizations that compares the actual and predicted roughness values, allowing for a direct comparison of their distribu-

tions. Through this visual analysis, we aim to gain a comprehensive understanding of our model's performance and its ability to accurately predict roughness in machining processes. A table and a graph were generated.

```
# Display the DataFrame as a table
print("Comparison of Actual and Predicted Roughness for all Samples:")
print(comparison_df)

# Plot actual and predicted roughness
plt.figure(figsize=(10, 6))
plt.plot(y, color='red', label='Actual Roughness')
plt.plot(predicted_roughness, color='blue', label='Predicted Roughness')
plt.xlabel('Sample Number')
plt.ylabel('Roughness')
plt.title('Actual vs. Predicted Roughness')
plt.legend()
plt.grid(True)
plt.show()
```

3.4 Optimization of ANN-PSO parameters

The second series of experiments involved optimizing the parameters of the ANN-PSO model to enhance prediction accuracy. Our aim was to identify optimal values for these parameters that would yield the highest level of accuracy in predicting roughness. The parameters considered for optimization included:

- Number of hidden neurons
- Number of particles
- Number of iterations
- c1/c2 values
- Inertia weight (ω) value

3.4.1 Number of hidden neurons

The PSO-ANN model was examined across varying numbers of hidden neurons in the architecture's hidden layer, including 20, 25, 30, 35, 40 and 50. While the number of hidden neurons is not a PSO parameter per se, it significantly influences the performance of the ANN-PSO model. Using (sklearn) package we could train the model with different sets of hidden neurons.

```
# Calculate accuracy metrics
mae = mean_absolute_error(y, predicted_roughness)
mse = mean_squared_error(y, predicted_roughness)
rmse = np.sqrt(mse)
```

The findings of this investigation are summarized in figure 3.2. With only 20 hidden neurons, the network lacked the necessary flexibility to effectively learn and adapt to the data, resulting in diminished prediction accuracy. Conversely, employing (30, 35, 50) hidden neurons led to prolonged training times and the potential for overfitting the data. Consequently, the optimal number of hidden neurons was determined to be between 25 and 50. These outcomes are illustrated in Figure 3.2.

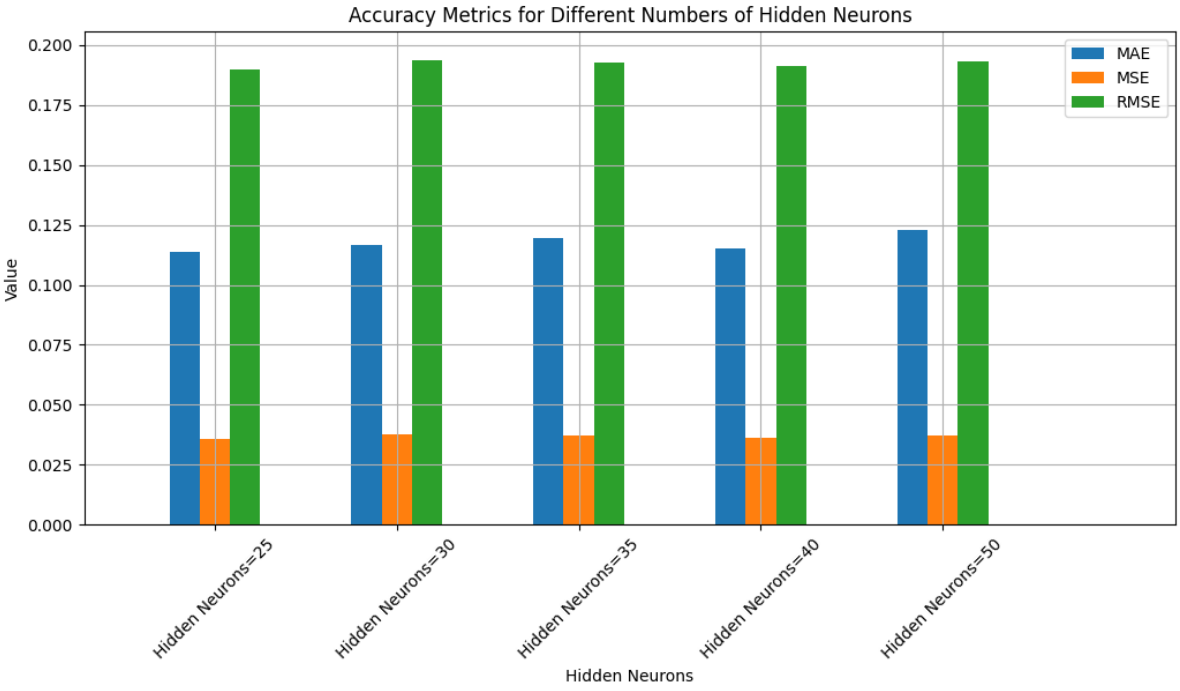


Figure 3.2: Accuracy metrics for different neurons

3.4.2 Number of particles

The number of particles utilized in ANN-PSO reflects the extent of the problem space coverage during each iteration of the loop. Our investigation involved testing ANN-PSO with particle counts ranging from 25 to 100.

```
# Define different numbers of particles to compare
particles_to_compare = [25, 35, 45, 55, 60, 70, 80, 90, 100]


# Train ANN-PSO models with different numbers of particles and
calculate accuracy metrics
results =
for particles in particles_to_compare:
```

```
predicted_roughness, mae, mse, rmse = train_ann_pso(X, y, particles)
results[f"Particles particles"] = 'Predicted Roughness': predicted_roughness,
'MAE': mae, 'MSE': mse, 'RMSE': rmse
```

It was observed that the model struggled with low particle counts of 25 and 35 due to insufficient coverage of the problem space. As the particle count increased to 45 and 60, there was a noticeable enhancement in prediction accuracy. However, beyond 50 particles, further increases did not yield additional improvements. In fact, employing 100 particles resulted in a decline in prediction accuracy, likely due to overfitting of the data. Moreover, optimization time increased as the particle count rose, as more particles necessitated broader coverage of the solution space, thereby prolonging the optimization process. Figure 3.3 illustrates the prediction accuracy corresponding to varying particle numbers.

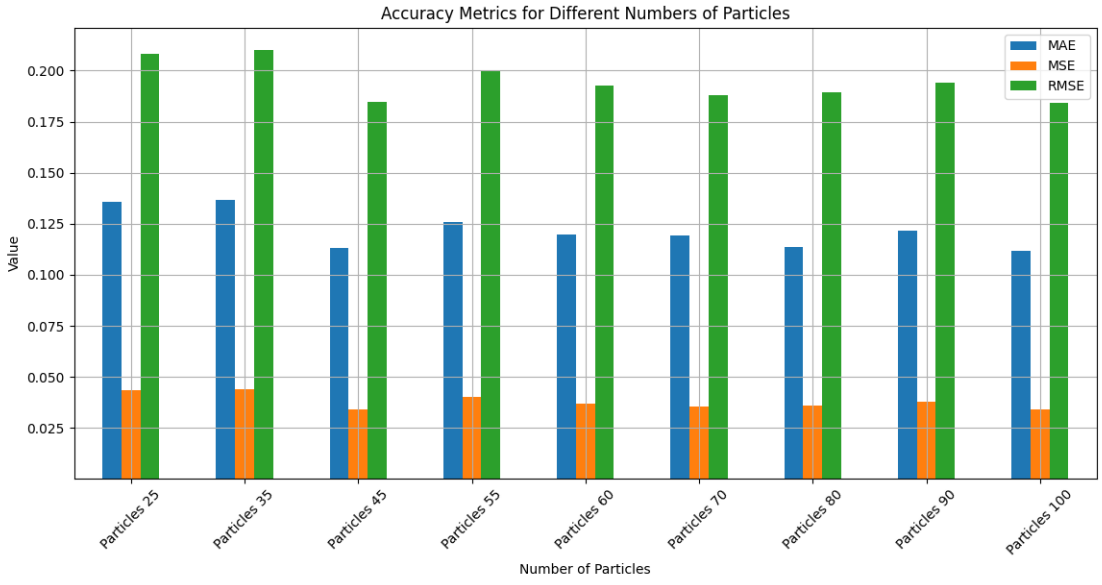


Figure 3.3: Accuracy metrics for different numbers of particles

3.4.3 Number of iterations

ANN-PSO was examined across different total iteration counts, specifically 5, 25, 50, 100, and 200.

```
import matplotlib.pyplot as plt
# Define different numbers of iterations to compare itera-
tions_to_compare = [20, 25, 30, 35, 40, 45, 50]

# Train ANN-PSO models with different numbers of iterations and
calculate accuracy metrics
results =
for iters in iterations_to_compare:
predicted_roughness, mae, mse, rmse = train_ann_pso(X, y, iters)
results[f"Iterations iters"] = 'Predicted Roughness': pre-
dicted_roughness,
'MAE': mae, 'MSE': mse, 'RMSE': rmse
# Plot accuracy metrics
plt.figure(figsize=(10, 6))
metrics_df.plot(kind='bar', y=['MAE', 'MSE', 'RMSE'], rot=45)
plt.title('Accuracy Metrics for Different Numbers of Iterations')
plt.xlabel('Number of Iterations')
plt.ylabel('Value')
plt.grid(True)
plt.legend(loc='upper right')
plt.tight_layout()
plt.show()
```

As the number of iterations increased, a larger portion of the problem space was explored, resulting in enhanced accuracy. However, at 20 and 35 iterations, ANN-PSO exhibited signs of being undertrained, failing to adequately approximate the underlying metrics-to-fault function. These observations are illustrated in Figure 3.4.

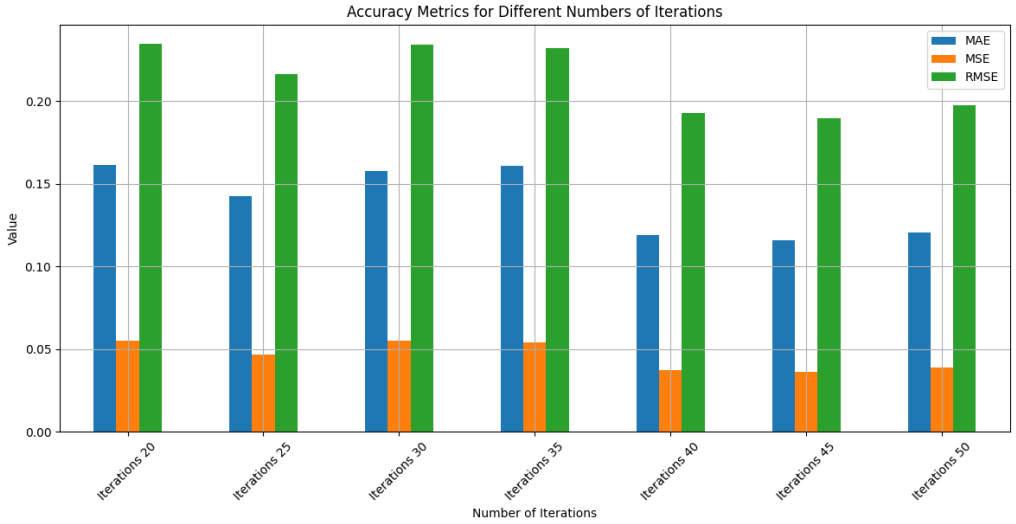


Figure 3.4: Accuracy metrics for different numbers of iterations

3.4.4 c1/c2 values

The constants c1 and c2 serve as acceleration constants that guide the particle towards either its global or personal best. Specifically, c1 represents the impact of the global best on the particle, while c2 signifies the influence of the personal best.

```
# Define different values for c1 and c2 to compare
c_values = [1, 1.2, 1.5, 1.7, 2]
# Train ANN-PSO models with different c1 and c2 values and calculate
accuracy metrics
```

```
results =
for c_value in c_values:
predicted_roughness, mae, mse, rmse = train_ann_pso(X, y, c_value)
results[f"c1=c2=c_value"] = 'Predicted Roughness': predicted_roughness,
'MAE': mae, 'MSE': mse, 'RMSE': rmse
```

Optimal outcomes were achieved when c1 and c2 fell within the range of 1.2 to 1.7.When set below 1.0, the acceleration proved insufficient to adequately explore the problem space, resulting in diminished prediction accuracy. Conversely, there was a marginal decline in prediction accuracy as the acceleration constants approached 2. Figure 3.5 illustrates this relationship of the prediction values corresponding to different c1 and c2 value graphically.

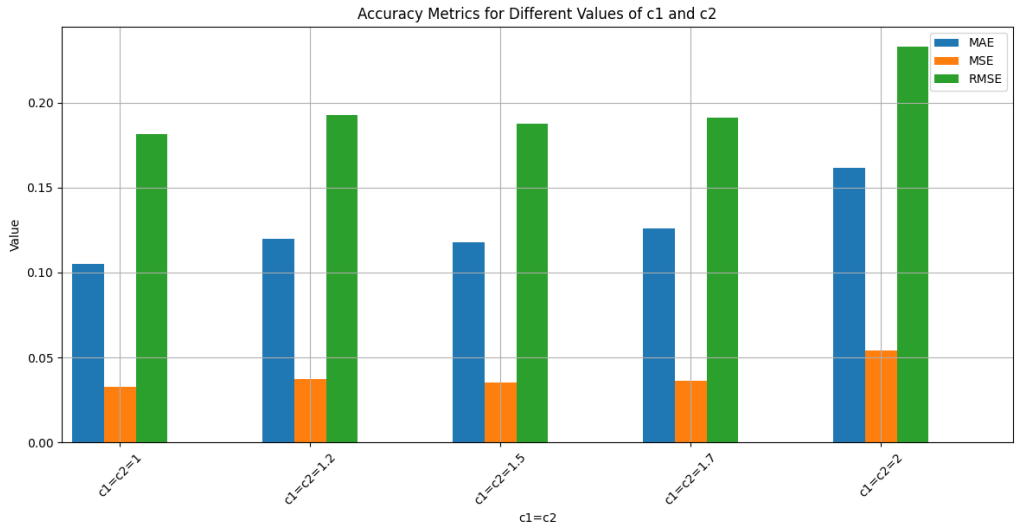


Figure 3.5: Accuracy metrics for different c values

3.4.5 Inertia weight (ω) value

The PSO-ANN was investigated with 0.3, 0.5, 0.7 and 0.9 inertia weights.

```
# Define different values for inertia weight (w) to compare
w_values = [0.3, 0.5, 0.7, 0.9]

# Train ANN-PSO models with different w values and calculate accuracy
metrics
results =
for w_value in w_values:
predicted_roughness, mae, mse, rmse = train_ann_pso(X, y, w_value)
results[f"w=w_value"] = 'Predicted Roughness': predicted_roughness,
'MAE': mae, 'MSE': mse, 'RMSE': rmse
```

At a low value of 0.3, the network does not give sufficient flexibility to learn and adapt to the data. This results in low prediction accuracy. Similarly for values over 0.5, the ANN taken a long time to get trained and may also be over fitting the data. So the optimum inertia weight was found to be 0.5. The results are captured in Figure 3.6.

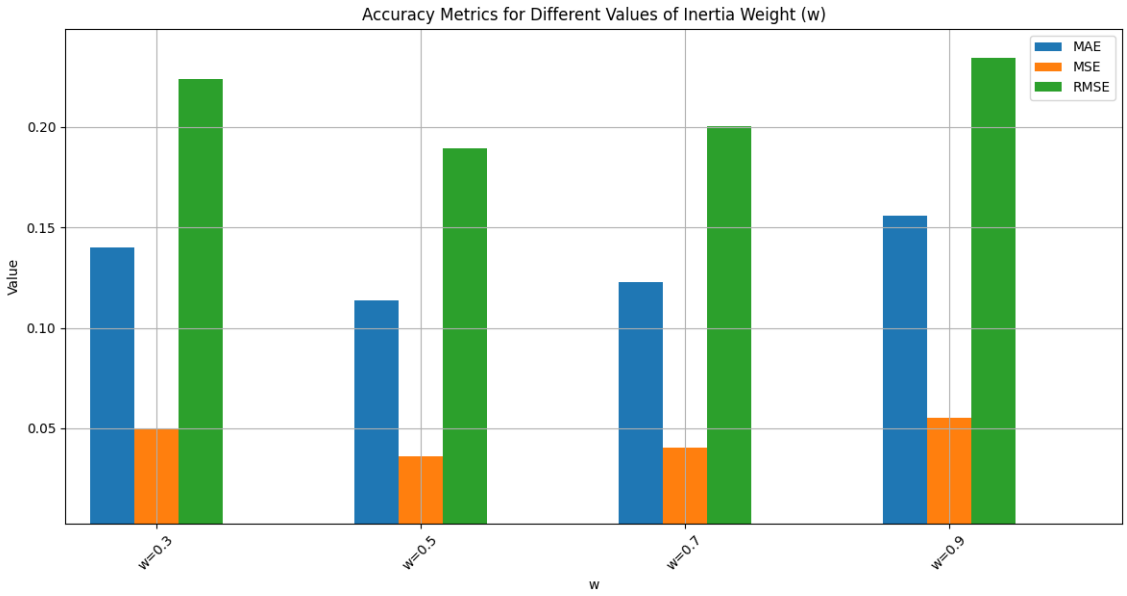


Figure 3.6: Accuracy metrics for different values of inertia weight

3.4.6 Optimal configuration for ANN-PSO

The optimal configuration for the ANN-PSO model was determined through a series of experiments aimed at fine-tuning its parameters. After extensive exploration, the following configuration emerged as the most effective:

1. Number of Hidden Neurons: Between 25 and 30 hidden neurons yielded the best results. Fewer than 25 neurons led to inadequate model flexibility, resulting in reduced prediction accuracy. Conversely, exceeding 30 neurons risked overfitting the data and increased training time.

2. Number of Particles: A particle count of 20 to 50 particles provided optimal coverage of the problem space without compromising computational efficiency. Fewer than 20 particles limited exploration, while exceeding 50 particles led to diminishing returns and longer optimization times.

3. Number of Total Iterations: Iterating between 50 and 100 times allowed

sufficient exploration of the problem space while balancing computational resources. Fewer iterations resulted in undertrained models, whereas additional iterations beyond 100 did not significantly improve prediction accuracy but prolonged optimization time.

4. Acceleration Constants (c1 and c2): Acceleration constants ranging from 1.2 to 1.7 struck a balance between global and personal best influences, facilitating effective exploration of the search space. Lower values limited exploration, while higher values led to sub-optimal convergence and decreased prediction accuracy.

By configuring the ANN-PSO model according to these guidelines, we achieved optimal performance in terms of prediction accuracy, computational efficiency, and convergence speed. the parameters are shown in Table 3.1

Table 3.1: Optimal parameters of ANN-PSO

Parameters	Values
Hidden neurons	25
Number of particles	45
Number of iterations	45
c1/c2	1.5
Inertia weight (ω)	0.5

3.5 Results and discussion

After optimizing the parameters of the PSO algorithm, the performance of the ANN-PSO model significantly improved. Leading to enhanced prediction accuracy, reduced training time, increased robustness, and stable convergence. These results underscore the importance of parameter tuning in achieving optimal performance in machine learning models.

3.5.1 Results of initial parameters

After building the model we tested it using either arbitrary parameters or from the previous studies in chapter two (Table 3.2). The results were not very accurate but the model showed potential for improving (section 3.1.3).

Table 3.2: Initial parameters of ANN-PSO

Parameters	Values
Hidden neurons	5
Number of particles	50
Number of iterations	50
c1/c2	0.3
Inertia weight (ω)	0.9

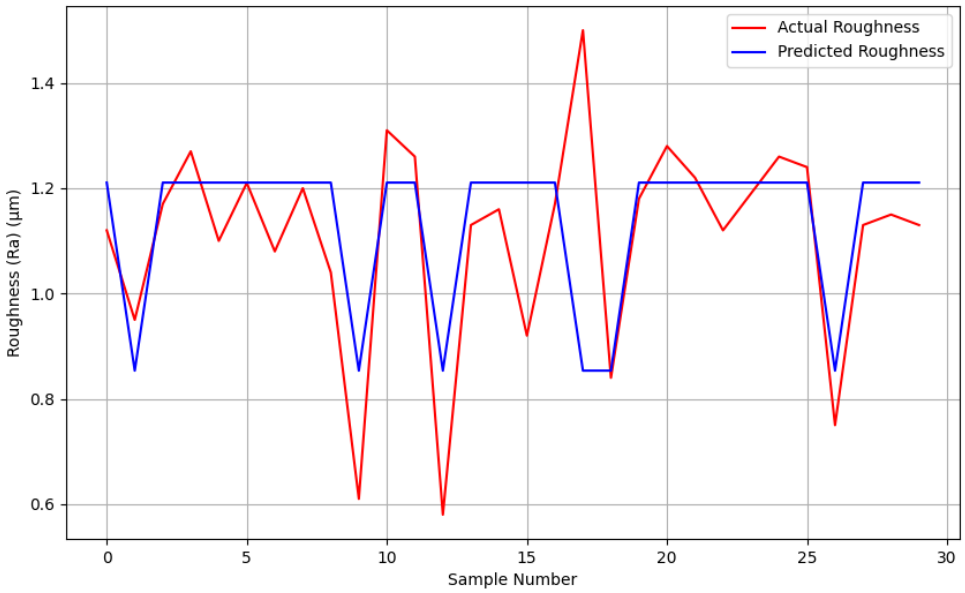


Figure 3.7: Comparison of actual and predicted surface roughness with intial parameters

	Actual Roughness	Predicted Roughness
0	1.12	1.210671
1	0.95	0.853530
2	1.17	1.210671
3	1.27	1.210671
4	1.10	1.210671
5	1.21	1.210671
6	1.80	1.210671
7	1.20	1.210671
8	1.04	1.210671
9	0.61	0.853530
10	1.31	1.210671
11	1.26	1.210671
12	0.58	0.853531
13	1.13	1.210671
14	1.16	1.210671
15	0.92	1.210671
16	1.17	1.210671
17	1.50	0.853530
18	0.84	0.853530
19	1.18	1.210671
20	1.28	1.210671
21	1.22	1.210671
22	1.29	1.210671
23	1.12	1.210658
24	1.19	1.210671
25	1.26	1.210671
26	1.24	1.210671
27	0.75	0.853530
28	1.13	1.210671
29	1.15	1.210671

Figure 3.8: comparison of actual and predicted surface roughness with initial parameters

a firt look at the graphe in Figure 3.7 shows the discrepancy between the acual and predicted results. this imply that the initial parameters used to predict the surface roughness won't be able to capture or understand the complexity of the relationship between cutting parameters and surface finish.

although it managed to accurately predict the SR samples(3, 16, 19, 25, 26) it failed on the rest by error margin that ranges between 0.1 1nd 0.6. this results are can be shown in table 3.8 and are illustrated in Figure 3.7

3.5.2 Results of optimal parameters

At first glance we notice that the model was only able to accurately predict in five sample. but the margin of error is significantly lower that ranges from 0.01 to 0.21 , or an accuracy of %80 to %99 . This is due to the optimization

of the ANN-PSO parameters. this result are shown and illustrated in more detail in Figure 3.9 and 3.10

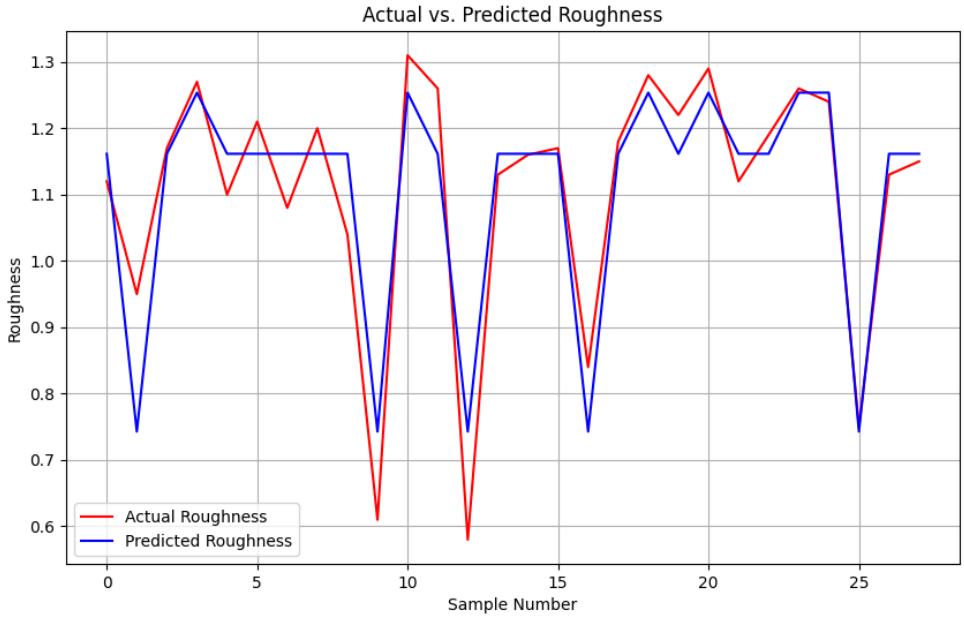


Figure 3.9: actual vs predicted surface roughness after the optimization

Comparison of Actual and Predicted Roughness for all Samples:

	Actual Roughness	Predicted Roughness
0	1.12	1.161487
1	0.95	0.742907
2	1.17	1.161514
3	1.27	1.253644
4	1.10	1.161512
5	1.21	1.161514
6	1.08	1.161512
7	1.20	1.161514
8	1.04	1.161472
9	0.61	0.742907
10	1.31	1.253644
11	1.26	1.161514
12	0.58	0.742907
13	1.13	1.161514
14	1.16	1.161513
15	1.17	1.161514
16	0.84	0.742907
17	1.18	1.161514
18	1.28	1.253644
19	1.22	1.161514
20	1.29	1.253644
21	1.12	1.161514
22	1.19	1.161514
23	1.26	1.253644
24	1.24	1.253644
25	0.75	0.742907
26	1.13	1.161514
27	1.15	1.161514

Figure 3.10: comparison of actual and predicted surface roughness after optimization

Table 3.3: Comparison between actual, initial prediction and optimal prediction of surface roughness

Spindle speed	Feed rate	Radial depth of cut	Axial depth of cut	Actual SR	Initial predicted SR	Optimal predicted SR
1500	150	2	25	1.12	1.078719	1.161487
2500	150	2	25	0.95	0.722366	0.742907
2500	300	1	25	1.17	1.209034	1.161514
1500	300	2	15	1.27	1.209034	1.253644
1500	150	2	15	1.1	1.209023	1.161512
2000	200	2.5	20	1.21	1.209033	1.161514
1500	150	1	15	45505	1.209024	1.161512
2000	200	1.5	20	1.2	1.209033	1.161514
1500	150	1	25	45383	1.090346	1.161472
3000	200	1.5	20	0.61	0.722366	0.742907
2000	500	1.5	20	1.31	1.209034	1.253644
2500	300	2	25	1.26	1.209034	1.161514
2000	100	1.5	20	0.58	0.573236	0.742907
2500	300	1	15	1.13	1.209034	1.161514
2000	200	1.5	30	1.16	1.209083	1.161513
2000	200	1.5	20	1.17	1.209033	1.161514
2500	150	1	15	0.84	0.722366	0.742907
2000	200	1.5	20	1.18	1.209033	1.161514
1000	200	1.5	20	1.28	1.209034	1.253644
2500	300	2	15	1.22	1.209034	1.161514
1500	300	2	25	1.29	1.209034	1.253644
2000	200	1.5	10	1.12	1.209034	1.161514
2000	200	1.5	20	1.19	1.209034	1.161514
1500	300	1	15	1.26	1.209034	1.253644
1500	300	1	25	1.24	1.209034	1.253644
2500	150	1	25	0.75	0.722366	0.742907
2000	200	1.5	20	1.13	1.209033	1.161514
2000	200	1.5	20	1.15	1.209033	1.161514

3.5.3 Comparison

After optimization, several key observations can be made from the provided dataset:

1. Impact of Machining Parameters: The optimized predictions show a closer alignment with the actual surface roughness compared to the initial predictions. This indicates that the optimization process has effectively improved the accuracy of the surface roughness predictions across various machining parameters.

2. Discrepancies between Actual and Predicted Surface Roughness: A comparison between the actual surface roughness and the initially predicted surface roughness exposes disparities in certain cases. This suggests that the initial prediction models couldn't accurately capture the complexities of the machining process.

3. Variation in Surface Roughness: Despite optimization, there are instances where the predicted surface roughness still deviates from the actual values. This could be attributed to the complex interplay between different machining parameters and their effects on surface quality, which may not have been fully captured or accounted for in the optimization process.

4. Influence of Optimization Techniques: The disparity between initial and optimal predicted surface roughness values highlights the importance of employing advanced optimization techniques, such as ANN-PSO, to refine predictive models. By iteratively adjusting model parameters, optimization algorithms can enhance the model's ability to accurately predict surface roughness under varying machining conditions.

5. Further Refinements: Despite the improvements achieved through optimization, there may still be room for further refinements in the prediction model. Fine-tuning the optimization parameters or exploring alternative modeling approaches could potentially lead to even better predictive performance and alignment with actual surface roughness values.

6. Practical Implications: The optimized predictions offer valuable insights for manufacturing processes, enabling operators to anticipate surface roughness outcomes based on specific machining parameters. This knowledge can inform decision-making and parameter adjustments to achieve desired surface quality standards more consistently and efficiently.

In summary, the optimization process has demonstrated its efficacy in improving the accuracy of surface roughness predictions, laying the groundwork for enhanced quality control and optimization in machining operations.

3.6 Validation of the Results

While comparing our model with other models using the same data is an essential part of model evaluation, it complements the validation process by providing a relative performance context. This combined approach helps ensure that our model is accurate, generalizable and competitive compared to other models in the field.

For this research we used two papers for comparison:

Paper 1: [Ugochukwu et al., 2015]

Paper 2: [Oulmane, 2023]

the actual values gotten from the experiment and the predicted values from the two papers are depicted in figures 3.13, 3.11 and 3.12. It can be seen that they have good agreement.

Quantitatively, In order to judge the accuracy of the models, percentage deviation ϕi and average percentage deviation Φi were used. The percentage deviation ϕi is stated thus:

$$\phi_i = \frac{\left|Ra_{(e)} - Ra_{(p)}\right|}{Ra_{(e)}} * 100 \tag{3.1}$$

Where:

Ra(e): measured.

Ra(p): predicted.

Similarly, the average percentage deviation Φi is stated thus:

$$\Phi = \frac{\sum \phi_i}{n} \tag{3.2}$$

Where:

$\sum \phi_i$: average percentage deviation of all sample data.

n: the size of sample data.

The result of average percentage deviation (Φ) showed that the training data set (n=28) was 5.73% for our model, 9.54% for the model used in paper1 and 0.37% for the model in paper2 . This means that the models could predict the surface roughness (Ra) with about 95%, 91% and 99% (respectively) accuracy of the training data set. For a full test on the model created on the training data, table 3.4 and 3.5 shows the predicted value for surface roughness and percentage deviation from the measured or actual Ra values.

Table 3.4: Comparison between Predicted Data and Predicted Data from paper1

Surface Roughness (Ra) (µm)	Predicted values (Ra) (µm)	Percentage of deviation ϕi	Predicted values (Paper1)(Ra) (µm)	Percentage of deviation (Paper1) ϕi
1,12	1.161487	3.704196428571413	1,09	2,82
0,95	0.742907	21.799263157894735	0,91	4,61
1,17	1.161514	0.7252991452991447	1,07	8,47
1,27	1.253644	1.2878740157480344	1,45	-13,91
1,1"	1.161512	5.5920000000000005	1,11	-0,93
1,21	1.161514	4.007107438016531	1,14	5,74
1,08	1.161512	7.54740740740741	1,01	6,77
1,2	1.161512	3.2073333333333216	1,06	11,56
1,04	1.161514	11.684038461538451	0,99	5,09
0,61	0.742907	21.788032786885246	0,92	-50,44
1,31	1.253644	4.301984732824433	1,51	-14,96
1,26	1.161514	7.816349206349211	1,18	6,28
0,58	0.742907	28.087413793103455	0,81	-40,43
1,13	1.161514	2.788849557522128	1,09	3,32
1,16	1.161514	0.13051724137931167	1,04	9,94
1,17	1.161514	0.7252991452991447	1,06	9,29
0,84	0.742907	11.558690476190476	0,84	0,19
1,18	1.161514	1.5666101694915255	1,06	10,06
1,28	1.253644	2.0590625000000036	1,36	-6,31
1,22	1.161514	4.793934426229511	1,2	1,27
1,29	1.253644	2.788849557522128	1,42	-9,94
1,12	1.161514	3.7066071428571274	1,09	2,65
1,19	1.161514	2.393781512605043	1,06	10,81
1,26	1.253644	0.5044444444444467	1,31	-4,13
1,24	1.253644	1.1003225806451606	1,29	-3,73
0,75	0.742907	0.9457333333333354	0,82	-9,58
1,13	1.161514	2.788849557522128	1,06	6,08
1,15	1.161514	1.0012173913043498	1,06	7,71
	$\Phi = 5.73$		$\Phi_1 = 9.54$	

Table 3.5: Comparison between Predicted Data and Predicted Data from paper2

Predicted values (Ra) (µm)	Percentage of deviationϕi	Predicted values (Paper2)(Ra) (µm)	Percentage deviation (Paper2) ϕi
1.161487	3.704196428571413	1.1200001239776611	-1.1069434020199204e-05
0.742907	21.799263157894735	0.950000047683758	-5.019338512075939e-06
1.161514	0.7252991452991447	1.1699997186660767	2.4045634467417905e-05
1.253644	1.2878740157480344	1.27000093460083	-7.359061654018593e-05
1.161512	5.5920000000000005	1.100000023841858	-2.1674416201216733e-06
1.161514	4.007107438016531	1.2099995613098145	3.62553872323428e-05
1.161512	7.54740740740741	1.0799999237060547	7.064254202180951e-06
1.161512	3.2073333333333216	1.1700000762939453	2.4999936421712206
1.161514	11.684038461538451	1.0399999618530273	3.6679781434401093e-06
0.742907	21.788032786885246	0.60999995470047	7.426152461635115e-06
1.253644	4.301984732824433	1.309999942779541	4.367973972340893e-06
1.161514	7.816349206349211	1.2600009441375732	-7.493155343121473e-05
0.742907	28.087413793103455	0.580000102519989	-1.7675860181662054e-05
1.161514	2.788849557522128	1.130000114440918	-1.0127514873923134e-05
1.161514	0.13051724137931167	1.1599998474121094	1.3154128495264133e-05
1.161514	0.7252991452991447	1.1700000762939453	-6.520850032782417e-06
0.742907	11.558690476190476	0.8399999737739563	3.122148055962575e-06
1.161514	1.5666101694915255	1.1700000762939453	0.8474511615300531
1.253644	2.0590625000000036	1.2800002098083496	-1.6391277311150754e-05
1.161514	4.793934426229511	1.21999990940094	7.426152461635115e-06
1.253644	2.788849557522128	1.2899987697601318	9.536743164337904e-05
1.161514	3.7066071428571274	1.1200001239776611	-1.1069434020199204e-05
1.161514	2.393781512605043	1.1700000762939453	1.68066585765165
1.253644	0.5044444444444467	1.2599984407424927	0.00012375059582008733
1.253644	1.1003225806451606	1.2400007247924805	-5.845100649013159e-05
0.742907	0.9457333333333354	0.75	0.0
1.161514	2.788849557522128	1.1700000762939453	-3.5398297605261435
1.161514	1.0012173913043498	1.1700000762939453	-1.7391370690387307
$\Phi = 5.73$		$\Phi_2 = 0.37$	

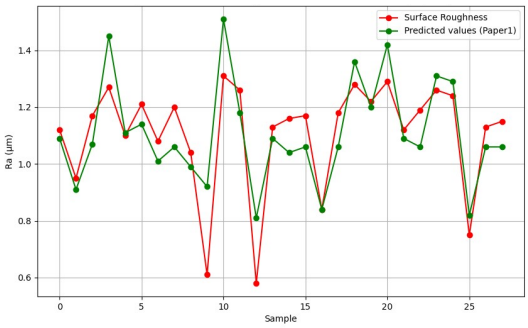


Figure 3.11: Comparison of Surface Roughness and Predicted Values (Paper1)

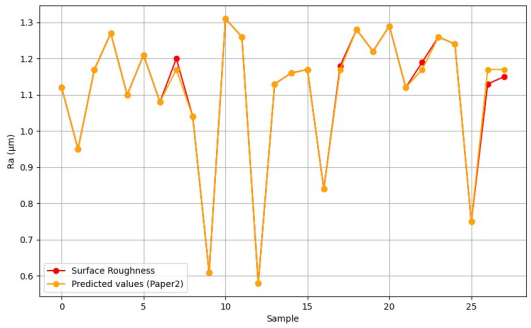


Figure 3.12: Comparison of Surface Roughness and Predicted Values (Paper2)

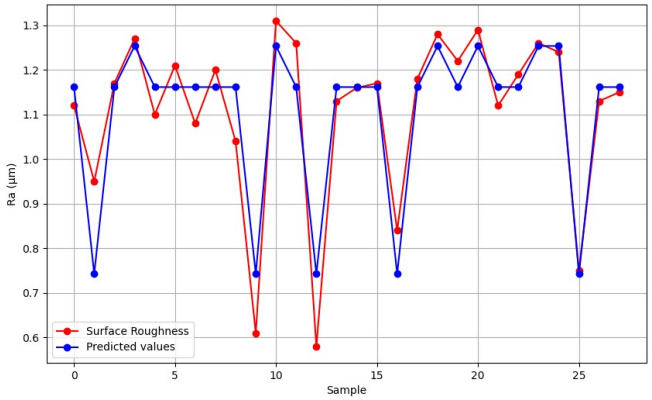


Figure 3.13: Comparison of Surface Roughness and Predicted Values

We compared the predictive performance of three models: our model, the model from Paper1, and the model from Paper2. Our model demonstrates a high accuracy of 95%, closely following the actual surface roughness values with minimal deviations. This validates the effectiveness of using Particle Swarm Optimization (PSO) to fine-tune the neural network parameters.

In comparison, Paper1's model, with a 91% accuracy, shows larger deviations, indicating less optimal performance. Paper2's model, with an impressive 99% accuracy, aligns almost perfectly with the actual values, suggesting a more sophisticated approach. While our model is robust and highly accurate, the near-perfect performance of Paper2's model highlights potential areas for improvement.

By exploring the advanced methodologies used in Paper2, such as additional feature engineering and more sophisticated model tuning, we can aim to enhance our model's accuracy further. This validation highlights the strengths of our model while providing a clear direction for future enhancements.

3.7 Conclusion

In conclusion, the implementation of the ANN-PSO model for surface roughness prediction has yielded valuable insights into the optimization of machining processes. Through the detailed training process and exploration of model architecture, we have gained a deeper understanding of how machine learning techniques can be leveraged to improve predictive accuracy.

The optimization of model parameters using PSO has shown promising results, with optimized predictions demonstrating closer alignment with actual surface roughness values across various machining conditions. This underscores the importance of employing advanced optimization techniques to refine predictive models and enhance their practical applicability in real-world manufacturing settings.

These findings have significant implications for quality control and optimization in mechanical machining processes. By enabling operators to anticipate surface roughness outcomes based on specific machining parameters, the ANN-PSO model empowers informed decision-making and parameter adjustments to achieve desired surface quality standards more consistently and efficiently.

General conclusion

In conclusion, this project has been a comprehensive exploration into the prediction and optimization of surface quality in mechanical machining processes. Through a systematic investigation, we have delved into various techniques and methodologies aimed at enhancing our understanding and control of surface roughness.

The journey began with an examination of the critical importance of surface quality in mechanical part manufacturing, emphasizing the need for precise dimensions and surface finishes to ensure functional integrity and aesthetic appeal.

Building upon this foundation, we explored state-of-the-art techniques for predicting surface roughness, ranging from traditional machining theory-based approaches to advanced artificial intelligence-based methodologies. This comprehensive review provided valuable insights into the strengths and limitations of existing techniques, laying the groundwork for the development of our predictive model.

The implementation of the ANN-PSO model represented a significant step forward in our quest for improved surface quality prediction. Through meticulous training, model architecture design, and parameter optimization using PSO, we were able to refine our predictive capabilities and achieve more accurate surface roughness predictions across a range of machining conditions.

The results obtained from our experiments provided compelling evidence of the effectiveness of the ANN-PSO model in predicting surface roughness. By comparing actual surface roughness values with both initial and optimized predicted values, we were able to demonstrate significant improvements in predictive accuracy, highlighting the practical relevance of our approach.

In closing, this project has not only deepened our understanding of surface quality prediction and optimization but has also opened up new avenues for future research and innovation in the field. By continuing to refine our predictive models, explore alternative optimization algorithms, and integrate additional input parameters, we can further enhance our ability to predict and control surface quality in mechanical machining processes, ultimately driving improvements in manufacturing efficiency and product quality.

Looking ahead, there are several potential avenues for future research in the field of surface quality prediction and optimization. Further refinement of predictive models, exploration of alternative optimization algorithms, and integration of additional input parameters could enhance the predictive accuracy and robustness of surface roughness prediction models. Additionally, the application of machine learning techniques to other aspects of mechanical machining processes, such as tool wear prediction and optimization of cutting parameters, presents exciting opportunities for future research and innovation in the field.

Bibliography

[c14,]

[Abernethy et al., 1985] Abernethy, R. B., Benedict, R. P., and Dowdell, R. B. (1985). Asme measurement uncertainty. *Journal of Fluids Engineering-transactions of The Asme*, 107:161–164.

[Adizue et al., 2023] Adizue, U., Tura, A., Isaya Elly, O., Farkas, B. Z., and Takacs, M. (2023). Surface quality prediction by machine learning methods and process parameter optimization in ultra-precision machining of aisi d2 using cbn tool. *The International Journal of Advanced Manufacturing Technology*, 129.

[Bansal and Ghangas, 2013] Bansal, N. and Ghangas, G. (2013). Optimization techniques for surface roughness of cnc milling: A review.

[Ben Mhenni, 2007] Ben Mhenni, A. A. (2007). *Influence de l'état de surface et du serrage sur les outils assemblés par frettage*. PhD thesis, École Polytechnique de Montréal.

[Benallou and Benchikh, 2022] Benallou, M. Z. and Benchikh, M. (2022). Étude des techniques d'amélioration de la qualité des surfaces usinées sur machine 05 axes. Master's thesis, Université M'Hamed Bougara-Boumerdes. Mémoire de fin d'études.

[Benardos and Vasniakos,] Benardos, P. G. and Vasniakos, G. C. Predicting surface roughness in machining: a review. *International journal of machine tools and manufacture*, 43:833–844.

[Benardos and Vosniakos, 2003] Benardos, P. G. and Vosniakos, G. C. (2003). Predicting surface roughness in machining: A review. *International Journal of Machine Tools and Manufacture*, 43(8):833–844.

[Bhushan, 2012] Bhushan, B. (2012). *Tribology and mechanics of magnetic storage devices*. Springer, Science Business Media.

[Blateyron and États de, 2006] Blateyron, F. and États de (2006). *surface : la Norme ISO 25178 Qui Va Tout Changer, Solutions, Mesures Mécaniques, n°787, pages 44 à 47*. Septembre.

[Boga and Koroglu, 2021] Boga, C. and Koroglu, T. (2021). Proper estimation of surface roughness using hybrid intelligence based on artificial neural network and genetic algorithm. *Journal of Manufacturing Processes*, 70:560–569.

[Bouhadja, 2023] Bouhadja, K. (2023). *Contribution à l'étude de la topographie 3D des surfaces gauches usinées sur des machines 5 axes*. PhD thesis, Ecole militaire polytechnique.

[Chen et al., 2022] Chen, T., Lee, J., Chen, H., Shie, M., and Lin, C. (2022). Optimization of milling parameters based on five-axis machining for centrifugal impeller with titanium alloy. *Journal of Physics Conference Series*, 2345(1):012019.

[Cheng Q, 2014] Cheng Q, Zhao H, Z. G. G. P. C. L. (2014). An analytical approach for crucial geometric errors identification of multi-axis machine tool

based on global sensitivity analysis. *The International Journal of Advanced Manufacturing Technology*, 75(1-4):107–121.

[Choudhury and El-Baradie, 1997] Choudhury, I. A. and El-Baradie, M. A. (1997). Surface roughness in the turning of high-strength steel by factorial design of experiments. *Journal of Materials Processing Technology*, 67:55–61.

[D. R. M. Rajesh, 2014] D. R. M. Rajesh, M. (2014). Prediction of surface roughness of freeform surfaces using artificial neural network. *no. Aimtdr, pp.*, pages 12–17.

[Dai and Ren, 2016] Dai, Q. and Ren, S. (2016). The principle and application of the multi-axis linkage machining center.

[Davim, 2010] Davim, J. (2010). Surface integrity in machining. 1848828742.

[Davim, 2001] Davim, J. P. (2001). A note on the determination of optimal cutting conditions for surface finish obtained in turning using design of experiments. *Journal of Materials Processing Technology*, 116:305–308.

[Dokeroglu et al., 2019] Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., and Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers Industrial Engineering*, 137:106040.

[Duc E and P, 1999] Duc E, Lartigue C, T. C. and P, B. (1999). A new concept for the design and the manufacturing of free-form surfaces: the machining surface. *Annals of the CIRP*, 48/1:103–106.

[E., 1998] E., D. (1998). *Usinages des formes gauches, contribution à l'amélioration de la qualité des trajectoires d'usinage.* PhD thesis, ENS Cachan.

[Ehmann and Hong, 1994] Ehmann, K. F. and Hong, M. S. (1994). A generalized model of the surface generation process in metal cutting. *CIRP Annals*, 43:483–486.

[El-Hofy, 2018] El-Hofy, H. (2018). *Fundamentals of Machining Processes, 3rd Edition*. CRC Press.

[Eser et al., 2021] Eser, A., Ayyıldız, E. A., Ayyıldız, M., and Kara, F. (2021). Artificial intelligence-based surface roughness estimation modelling for milling of aa6061 alloy. *Advances in Materials Science and Engineering*, 2021:10. Article ID 5576600.

[Farouki, 1986] Farouki, R. T. (1986). The approximation of non-degenerate offset surfaces. *Computer Aided Geometric Design*, 3:15–43.

[Fuer, 1982] Fuer, D. I. (1982). deviations, concepts, classification system. Din4760, Form.

[Garcia-Diaz and Phillips, 1995] Garcia-Diaz, A. and Phillips, D. T. (1995). *Principles of Experimental Design and Analysis*. Chapman and Hall, London.

[Gasick J, 2021] Gasick J, Q. X. (2021). Simultaneous topology and machine orientation optimization for multiaxis machining. *International Journal for Numerical Methods in Engineering*, 122(24):7504–7535.

[Hong et al., 2011] Hong, C., Ibaraki, S., and Matsubara, A. (2011). Influence of position-dependent geometric errors of rotary axes on a machining test of cone frustum by five-axis machine tools. *Precision Engineering*, 35(1):1–11.

[Hu and Gai, 2009] Hu, L. and Gai, R. (2009). Adaptive fuzzy sliding controller with dynamic compensation for multi-axis machining. *Journal of Software Engineering and Applications*, 02(4):288–294.

[Institute, 1986] Institute, A. N. S. (1986). Surface integrity. *Society of Manufacturing Engineers. Surface Integrity.*, pages 1–17. Ansi, B. 211. 1.

[Iso-25178-2, 2012] Iso-25178-2 (2012). Afnor. *Geometrical product specifications (GPS).*

[Iso-4287, 1998] Iso-4287 (1998). Afnor. *Geometrical product specification (GPS).*

[Jesuthanam and Kumanan, 2003] Jesuthanam, C. P. and Kumanan, S. (2003). In-process quality control in machining using artficial neural network. In *Proceedings of the National conference on Advanced Machine Vision system for Societal applications*, pages 215–221.

[Jiang et al., 2021] Jiang, J., Li, B., Lin, F., Zhang, H., and Ye, P. (2021). Prediction and compensation strategy of contour error in multi-axis motion system.

[jm, 2005] jm, Jafferson Jesuthanam, C. . K. S. (2005). Prediction of surface roughness using ai techniques - a review. pages 83–90.

[K and H., 1994] K, S. and H., Y. D. C. (1994). Constant scallop-height machining of free-form surfaces. *Journal of Engineering for Industry*, 116.

[Kalyanmoy, 1995] Kalyanmoy, D. (1995). *Optimization for Engineering Design—Algorithms and Examples.* Prentice-Hall, India.

[Karlo, 2009] Karlo, A. (2009). *Secrets of 5-axis machining.* Industrial Press.

[Kelkar and Koc, 2008] Kelkar, A. and Koc, B. (2008). Geometric planning and analysis for hybrid re-configurable molding and machining process. *Rapid Prototyping Journal*, 14(1):23–34.

[Khawaja, 2014a] Khawaja, Z. (2014a). *Analyse des états de surface en science des matériaux : caractérisation multi échelles par ondelette et détermination de l'anisotropie des surfaces.* Mécanique [physics. med-ph]. Université de Technologie de Compiègne Français.

[Khawaja, 2014b] Khawaja, Z. (2014b). *Analyse des états de surface en science des matériaux : caractérisation multi-échelles par ondelette et détermination de l'anisotropie des surfaces.* PhD thesis, Université de Technologie de Compiègne.

[Khorasani et al., 2012] Khorasani, A. M., Yazdi, M. R. S., and S., S. M. (2012). Analysis of machining parameters effects on surface roughness: a review. *International Journal of Computational Materials Science and Surface Engineering*, 5(1):68–84.

[Kim, 1995] Kim, K. I., K. K. (1995). A new machine strategy for sculptured surfaces using offset surface. *International Journal of Production Research*, 33(6).

[Klopfstein M. J, 2011] Klopfstein M. J, L. D. A. (2011). Recent assessment of surface integrity resulting from fine finishing processes. *Procedia Engineering*, 19:209–221.

[Kumar et al., 2021] Kumar, G., Kumar, M., and Tomer (2021). Optimization of end milling machining parameters of ss 304 by taguchi technique. In *In Recent Advances in Mechanical Engineering*, pages 683–689. Springer, Singapore.

[Lauwers B, 2003] Lauwers B, Dejonghe P, K. J. (2003). Optimal and collision free tool posture in five-axis machining through the tight integration of tool path generation and machine simulation. *Computer-Aided Design*, 35(5):421–432.

[Lebon, 2017] Lebon, N. (2017). Impact de l'usinage par cfao sur l'intégrité de surface des prothèses dentaires coronaires. *Mécanique des matériaux [physics.class-ph].* Université Sorbonne Paris Cité.

[Lederer, 2021] Lederer, J. (2021). Activation functions in artificial neural networks: A systematic overview.

[Lee and Yang, 2010] Lee, D. and Yang, S. (2010). Mathematical approach and general formulation for error synthesis modeling of multi-axis system. *International Journal of Modern Physics B*, 24(15n16):2737–2742.

[Lee et al., 2004] Lee, K. C., Ho, S. J., and Ho, S. Y. (2004). Accurate estimation of surface roughness from texture features of the surface image using an adaptive neuro-fuzzy inference system. *International Journal of Precision Engineering*, 29:95–100.

[Lee et al., 2001] Lee, K. Y., Kang, M. C., Jeong, Y. H., Lee, D. W., and Kim, J. S. (2001). Simulation of the surface roughness and profile in high speed end milling. *Journal of Materials Processing Technology*, 113:410–415.

[Lee J, 2022] Lee J, Yeh H, S. M. C. T. (2022). Improvement in the efficiency of the five-axis machining of aerospace blisks. *Science Progress*, 105(4).

[Lee R, 1997] Lee R, S. C. (1997). Developing a postprocessor for three types of five-axis machine tools. *The International Journal of Advanced Manufacturing Technology*, 13(9):658–665.

[Maekawa et al., 1997] Maekawa, T., Cho, W., and Patrikalakis, N. M. (1997). Computation of self-intersections of offsets of bezier surface patches. *Journal of Mechanical Design*, 119.

[Martin et al., 2021] Martin, A., Brown, D., Diamond, M., Cattaneo, A., de Miguel, F., and Nicholls, J. (2021). *From Neuron to Brain.* Oxford University Press.

[Moghri et al., 2014] Moghri, M., Madic, M., Omidi, M., and Farahnakian, M. (2014). Surface roughness optimization of polyamide 6/nanoclay nanocomposites using artificial neural network: genetic algorithm approach [hindawi publishing corporation. *Article*, ID 485205] Sci World J:1–7.

[Métrologie, 1999] Métrologie, P. B. (1999). Métrologie des surfaces. *Techniques de l'Ingénieur, Traité Mesure et Contrôle*, RAB.

[Oktem et al., 2006] Oktem, H., Erzurumlu, T., and Erzincanli, F. (2006). Prediction of minimum surface roughness in end milling mold parts using neural network and genetic algorithm. *Mater. Des.*, 27(9):735–744.

[Oulmane, 2023] Oulmane, S. (2023). Optimisation des conditions de coupe et l'étude de leurs effets sur l'état de surface en fraisage. Master's thesis, École National Polytechnique Alger. Mémoire de fin d'études.

[Pateloup S, 2011] Pateloup S, Chanal H, D. E. (2011). Étude des performances de machines-outils 5 axes à structure parallèle et sérielle pour l'usinage d'une pièce aéronautique. *Mécanique Amp; Industries*, 12(6):479–486.

[Petropoulos G. P, 2010] Petropoulos G. P, Pandazaras C. N, D. J. P. (2010). Surface texture characterization and evaluation related to machining. *In Surface integrity in machining*, pages 37– 66. Springer London.

[Rauch M, 2010] Rauch M, X. X. (2010). Five-axis machining: technologies and challenges. *Int. J. Manufacturing Research*, 5(3).

[Ross, 1996] Ross, P. J. (1996). *Taguchi Techniques for Quality Engineering: Loss Function, Orthogonal Experiments, Parameter and Tolerance Design.* McGraw-Hill, New York.

[Serope, 1928] Serope, K., S. R. S. (1928). *Manufacturing Engineering and Technology.* Eighth edition edition.

[Shapiro, 2007] Shapiro, A. (2007). An overview of insurance uses of fuzzy logic. *Computational Intelligence in Economics and Finance:*, 2:25–61.

[She C, 2012] She C, Lin C, Y. T. (2012). Development of the integration of postprocessor for five-axis machine tools.

[Sheikh-Ahmad, 2009] Sheikh-Ahmad, J. (2009). *Machining of Polymer Composites.*

[Subbiah, 2014] Subbiah, S. (2014). Science of machining. In *Handbook of Manufacturing Engineering and Technology*, pages 1–21. Springer, London.

[Tournier and Duc, 2001] Tournier, C. and Duc, E. (2001). Usinage de formes gauches : génération de trajectoires outils à hauteur de crête constante.

[Touzet, 1992] Touzet, C. (1992). *LES RESEAUX DE NEURONES ARTIFICIELS, INTRODUCTION AU CONNEXIONNISME.* Collection de l'EERIE. EC2.

[Ugochukwu et al., 2015] Ugochukwu, K., OkokpujieImhade, and Chinenye, O. (2015). Cutting parameters effects on surface roughness during end milling of aluminium 6061 alloy under dry machining operation. *International Journal of Science and research (IJSR).* Department of Mecha Niger Delta University Wilberforce Island Bayelsa, Nigeria,.

[Umehara T, 2007] Umehara T, Ishida T, T. K. T. Y. (2007). Effective tool pass generation method for roughing in multi-axis control machining. *Transactions of the Japan Society of Mechanical Engineers Series C*, 73(732):2387–2393.

[Wang et al., 2018] Wang, D., Tan, D., and Liu, L. (2018). Particle swarm optimization algorithm: an overview. *Soft Computing*, 22.

[Wang S, 2021] Wang S, Chen Y, Z. G. (2021). Adaptive fuzzy pid cross coupled control for multi-axis motion system based on sliding mode disturbance observation. *Science Progress*, 104(2).

[Whitehouse, 1997] Whitehouse, D. J. (1997). Surface metrology. *Meas. Sci. Technol*, 8.

[Yang et al., 2004] Yang, C.-X., Tham, L., Feng, X.-T., Wang, Y. J., and Lee, P. K. K. (2004). Two-stepped evolutionary algorithm and its application to stability analysis of slopes. *Journal of Computing in Civil Engineering - J COMPUT*, CIVIL ENG. 18.

[Yin L, 2003] Yin L, Jahanmir S, I. L. K. (2003). Abrasive machining of porcelain and zirconia with a dental handpiece. *Wear*, 255(7):975–989.

[Yin L, 2006a] Yin L, Song X. F, Q. S. F. H. Y. G. W. H. (2006a). Surface integrity and removal mechanism in simulated dental finishing of a feldspathic porcelain. *Journal of Biomedical Materials Research Part B: Applied Biomaterials*, 79(2):365–378.

[Yin L, 2006b] Yin L, Son X. F, S. Y. L. H. T. L. J. (2006b). An overview of in vitro abrasive finishing cad/cam of bioceramics in restorative dentistry. *International Journal of Machine Tools and Manufacture*, 46(9):1013–1026.

[Zhang Z, 2011] Zhang Z, H. H. (2011). Measuring geometrical errors of linear axis of machine tools based on the laser tracker.

[Zhao et al., 2010] Zhao, H., Tang, X., Chen, X., and Wang, L. (2010). Simulation and experimental research on modal analysis for a new 5-axis superalloy blade machine tool.