Institut für
Kontinuumsmechanik

Meshfree Modelling of Metal Cutting using Phenomenological and Data-driven Material Models

Dengpeng Huang

B20/5

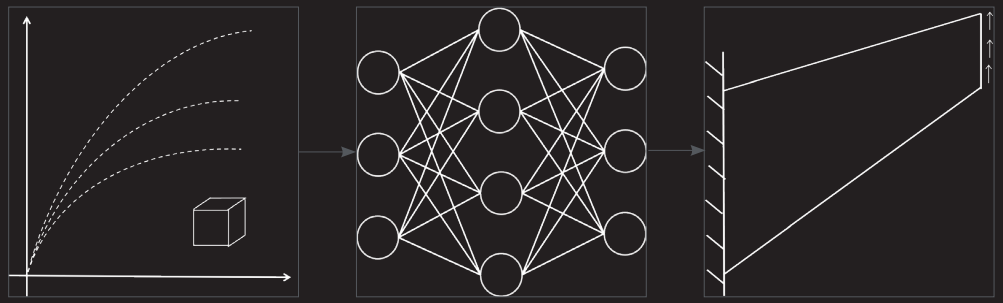
Leibniz
Universität
Hannover

ISBN 978-3-941302-39-6

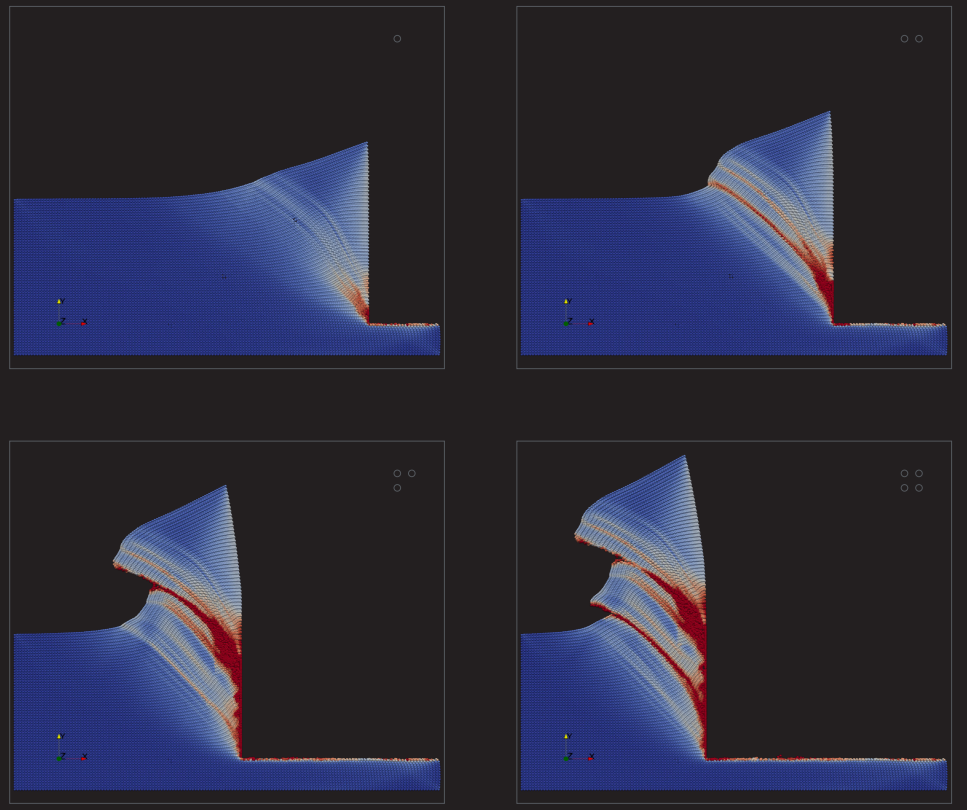
Numerical modelling of the metal cutting is important for understanding, improvement and digitalisation of the high speed machining processes.

In this work, the phenomenological model is firstly applied to simulate the serrated chip formation within a Galerkin type meshfree scheme. By investigating the stress state and thermal softening in cutting zone, a realistic simulation of material separation and adiabatic shear band is obtained.

Then, the data-driven material model is developed to replace the classical material model, where the artificial neural networks combined with the proper orthogonal decomposition is first trained offline to fit an observed elastoplastic behaviour and later used in online applications. The effectiveness and generalisation of the presented models are investigated in finite element analysis.



Dengpeng Huang | Meshfree Modelling of Metal Cutting using Phenomenological and Data-driven Material Models



Meshfree Modelling of Metal Cutting using Phenomenological and Data-driven Material Models

Von der Fakultät für Maschinenbau
der Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des akademischen Grades
Doktor-Ingenieur

genehmigte Dissertation
von

M. Sc. Dengpeng Huang

geboren am 03.07.1989 in Gansu, China

2020

Herausgeber:
Prof. Dr.-Ing. habil. Dr. h.c. mult. Dr.-Ing. E. h Peter Wriggers

Verwaltung:
Institut für Kontinuumsmechanik
Gottfried Wilhelm Leibniz Universität Hannover
An der Universität 1, Gebäude 8142
30823 Garbsen

Tel: +49 511 762 3220
Fax: +49 511 762 5496
Web: www.ikm.uni-hannover.de

ISBN 978-3-941302-39-6

1. Referent: Prof. Dr.-Ing. habil. Dr. h.c. mult. Dr.-Ing. E. h Peter Wriggers
2. Referent: Prof. Dr.-Ing. Udo Nackenhorst

Tag der Promotion: 11.12.2020

Zusammenfassung

Die numerische Modellierung der Spanbildung ist für ein besseres Verständnis und damit für eine Verbesserung von Hochgeschwindigkeits-Metallbearbeitungsprozessen wichtig. Ziel dieser Arbeit ist die Entwicklung eines netzfrei numerischen Simulationsmethode für den Spanbildungsprozess, in dem sowohl ein phänomenologisches Materialmodell als auch datengesteuerte Materialmodelle angewendet werden können.
Zuerst wird dabei das phänomenologische Modell zur Erfassung der gezackten Spanbildung verwendet, wobei ein kürzlich entwickeltes netzfrei Methode des Galerkin Typs, die stabilisierte Optimal Transportation Meshfree (OTM)-Methode als numerische Lösungsmethode angewendet wird. Dies ermöglicht die Modellierung der Materialtrennung und der gezahnten Morphologieerzeugung des Schneidprozesses auf realistischere und zweckmäßigere Weise zu gestalten. Die Scherbandformation wird durch den thermischen Erweichungsbegriff des Johnson-Cook Fließspannungsmodells beschrieben. Mit diesem Modell kann nachgewiesen werden, dass die thermische Erweichung die Hauptursache für die Scherbandformation ist. Darüber hinaus ist zu erkennen, dass das Johnson-Cook Frakturmodell Einschränkungen bei der Erfassung des Bruchs auf der Chip-Oberfläche aufweist. Damit wird eine zusätzliche Bedingung für die Spannungsdreiachsenzahl gestellt. Diese Bedingung ermöglicht eine genauere Bemessung der Chipgröße, wie z.B. Chipabstand, Spitze und Tal. Diese Verbesserungen werden durch den Vergleich der berechneten Spanmorphologie, Schnittkraft und des Spanbildungsprozesses mit experimentellen Ergebnissen demonstriert.
Anschließend wird das datengesteuerte Materialmodell entwickelt, um das klassische Materialmodell zu ersetzen, bei dem das auf Machine Learning (ML) basierende Modell zunächst offline auf ein beobachtetes Materialverhalten trainiert und später in Online-Anwendungen eingesetzt wird. Das Erlernen und Vorhersagen historisch bedingter Materialmodelle, wie z.B. Plastizität, ist jedoch nach wie vor eine Herausforderung. In dieser Arbeit wird ein ML-basierter Method der Materialmodellierung sowohl für die Elastizität als auch für die Plastizität vorgeschlagen, bei dem das ML-basierte Hyperelastizitätsmodell mit dem Feed forward Neural Network (FNN) direkt entwickelt wird, während das ML-basierte Plastizitätsmodell unter Verwendung von zwei Ansätzen entwickelt wird, eines Recurrent Neural Network (RNN) und einer neuen Methode namens Proper Orthogonal Decomposition Feed Forward Neural Network (PODFNN). Im letzteren Fall wird die akkumulierte absolute Dehnung verwendet, um die Belastungshistorie zu modellieren. Zusätzlich werden Dehnungs-Spannungs-Sequenzdaten, basierend auf dem Konzept der Sequenz für Plastizität, für das Plastizitätsmodell aus verschiedenen Belastungspfaden gesammelt. Mit Hilfe des POD wird die mehrdimensionale Spannungssequenz als unabhängige eindimensionale Koeffizientensequenz entkoppelt. Um das ML-basierte Materialmodell in der Finite-Elemente-Analyse anzuwenden, wird die Tangentenmatrix durch das automatische symbolische Differenzierungswerkzeug AceGen aufgestellt. Die Effektivität und Generalisierungsfähigkeit der vorgestellten Modelle wird anhand einer Reihe von numerischen Beispielen untersucht, wobei sowohl 2D- als auch 3D-Finite Elemente analysiert werden. Abschließend wird das ML-basierte Materialmodell in Metallschneidesimulationen angewendet.
Schlagworte: Spanbildung, Duktiler Bruch, Stabilisiertes OTM, Machine Learning, Künstliches Neuronales Netzwerk, Plastizität, Korrekte Orthogonale Zerlegung

Abstract

Numerical modelling of chip formation is important for a better understanding thus for improvement of the high speed metal cutting process. The objective of this work is to develop a meshfree numerical simulation framework for the chip formation process, where both the phenomenological material model and the data-driven material model can be applied.
Firstly, the phenomenological model is applied to capture the serrated chip formation, where a recently developed Galerkin type meshfree scheme, the stabilized Optimal Transportation Meshfree (OTM) method, is applied as a numerical solution method. This enables the modelling of material separation and serrated morphology generation of the cutting process in a more realistic and convenient way. The shear band formation is described by the thermal softening term in the Johnson-Cook plastic flow stress model. Using this model, it can be demonstrated that thermal softening is the main cause of the shear band formation. Additionally, it can be seen that the Johnson-Cook fracture model shows limitations in capturing the fracture on the chip upper surface. Thus, a supplementary condition for the stress triaxiality is applied. This condition allows more accurate measurements of the chip size, i.e. chip spacing, peak and valley. These improvements are demonstrated by comparing the calculated chip morphology, cutting force and chip formation process with experimental results.
Subsequently, the data-driven material model is developed to replace the classical material model, where the Machine Learning (ML) based model is first trained offline to fit an observed material behaviour and later be used in online applications. However, learning and predicting history dependent material models such as plasticity is still challenging. In this work, a ML based material modelling framework is proposed for both elasticity and plasticity, in which the ML based hyperelasticity model is directly developed with the Feed forward Neural Network (FNN), whereas the ML based plasticity model is developed by using two approaches including Recurrent Neural Network (RNN) and a novel method called Proper Orthogonal Decomposition Feed forward Neural Network (PODFNN). In the latter case, the accumulated absolute strain is proposed to distinguish the loading history. Additionally, the strain-stress sequence data for the plasticity model is collected from different loading paths based on the concept of sequence for plasticity. By means of the POD, the multi-dimensional stress sequence is decoupled as independent one dimensional coefficient sequences. To apply the ML based material model in finite element analysis, the tangent matrix is derived by the automatic symbolic differentiation tool AceGen. The effectiveness and generalisation of the presented models are investigated by a series of numerical examples using both 2D and 3D finite element analysis. Finally, the ML based material model is applied to the metal cutting simulation.

Keywords: Chip Formation, Ductile Fracture, Stabilized OTM, Machine Learning, Artificial Neural Network, Plasticity, Proper Orthogonal Decomposition

Acknowledgements

This dissertation is the result of my research work during the past four years at the Insitute of Continuum Mechanics (IKM) at the Leibniz Universität Hannover (LUH) under the guidance of Prof. Dr.-Ing. habil. Dr. h.c. mult. Dr.-Ing. E.h. Peter Wriggers. The project was supported by the China Scholarship Council (CSC scholarship) and the Gradient Academy of LUH (Abschluss-Stipendium).

Firstly, I would like to express my sincere gratitude to my advisor and principal referee, Professor Peter Wriggers, for giving me the opportunity to study at his institute, for his professional suggestions on my project and for his strong support to participate in scientific conferences. His detailed corrections and remarks on my research work helped me understand scientific problems better.

My deep thanks go to Dr.-Ing. Christian Weißenfels, who guided me all through my doctoral program. His door has been always open to me and his feedback on my work has been extremely efficient all the time. His professional suggestions on my topic helped me to accomplish what I want to do, and his encouragement inspired me to explore something out of my horizon. I am extremely thankful for his patient and the fruitful discussions.

Special thanks also go to Jan Niklas Fuhg. I benefited a lot from the discussions with him on machine learning and learned a lot from the cooperation we had. I want to thank my office mates, Dr.-Ing. Philipp Hartmann and Thanh Chuong Nguyen, for the discussions, the strong support and the pleasant working atmosphere in our office. Furthermore, I am thankful to Vera Halfar and Dorit Schulte for handling the documents, to Jens Bsdok and Burkhard Salge for providing IT-support, to Volker Meine for printing book chapters and to all of the current and former colleagues at IKM for the enjoyable team atmosphere.

Most important, I am grateful to my family for their constant support all the time. I would like to express my deepest gratitude to my wife, Dr. Yusen Luo, for her love, understanding and encouragement during the last seven years.

Hannover, June 2020 *Dengpeng Huang*

Contents

Nomenclature

Scalars

a	Area in current configuration
A	Area in reference configuration
λ_i	Eigenvalues
γ	Plastic multiplier
μ	Shear modulus
ν	Poisson's ratio
ρ	Current density
ρ_0	Initial density
σ_Y	Yield stress
σ_v	von Mises stress
σ_{eq}	Equivalent stress
ξ	Isotropic plastic hardening variable
θ	Temperature
θ_r	Room temperature
θ_m	Melting temperature
Φ	Dissipation potential
Ψ	Helmholtz free energy per unit volume in reference configuration
ψ	Helmholtz free energy per unit volume in current configuration
C_p	Specific heat capacity
D_{int}	Dissipation

P_{int}	Stress power
E	Young's modulus
K	Bulk modulus
G	Shear modulus
m	Mass
$J = \det(\mathbf{F})$	Jacobian
p	Hydrostatic pressure
r	Internal volumetric heat source
t	Time
η	Specific entropy per unit volume
e	Specific internal energy per unit volume
v	Volume in current configuration
V	Volume in reference configuration
n_{np}	Number of nodal points
n_{mp}	Number of material points
g^N	Normal gap
c^N	Penalty parameter in normal direction
c^T	Penalty parameter in tangential direction
β	Taylor-Quinney coefficient
η_s	Stress triaxiality
α_r	Rake angle
α_f	Flank angle
v_c	Cutting velocity

Vectors

$\boldsymbol{x}$	Geometrical point in current configuration
$\boldsymbol{X}$	Geometrical point in reference configuration
$\boldsymbol{\varphi}$	Deformation
$\boldsymbol{u}$	Displacement
$\boldsymbol{v}$	Velocity
$\boldsymbol{a}$	Acceleration
$\boldsymbol{b}$	Body force per unit mass
$\boldsymbol{t}$	Surface force with respect to current configuration
$\boldsymbol{T}$	Surface force with respect to reference configuration
$\boldsymbol{n}$	Normal vector in current configuration
$\boldsymbol{N}$	Normal vector in reference configuration
$\boldsymbol{q}$	Cauchy heat flux
$\boldsymbol{t}^N$	Normal contact force
$\boldsymbol{t}^T$	Tangential contact force

Second order tensors

$\boldsymbol{\varepsilon}$	Small strain tensor
$\boldsymbol{\sigma}$	Cauchy stress tensor
$\boldsymbol{\tau}$	Kirchhoff stress tensor
$\boldsymbol{b}$	Left Cauchy-Green tensor
$\boldsymbol{C}$	Right Cauchy-Green tensor
$\boldsymbol{d}$	Symmetric part of spatial velocity gradient
$\boldsymbol{e}$	Euler-Almansi strain tensor
$\boldsymbol{E}$	Green-Lagrange strain tensor
$\boldsymbol{F}$	Deformation gradient
$\boldsymbol{I}$	Second order identity tensor

$\boldsymbol{l}$	Spatial velocity gradient
$\boldsymbol{P}$	First Piola-Kirchhoff stress tensor
$\boldsymbol{S}$	Second Piola-Kirchhoff stress tensor
$\boldsymbol{U}$	Right stretch tensor
$\boldsymbol{V}$	Left stretch tensor

Mathematical operators

div	Divergence in current configuration
Div	Divergence in reference configuration
grad	Gradient in current configuration
Grad	Gradient in reference configuration
sym	Symmetric part
$\det(\cdot)$	Determinant
$\mathcal{L}(\cdot)$	Lie-derivative
$\mathbf{F}^T(\cdot)\mathbf{F}$	Pull back of spatial tensor into reference configuration
$\mathbf{F}^{-T}(\cdot)\mathbf{F}^{-1}$	Push forward of material tensor into current configuration
$\mathrm{tr}(\cdot)$	Trace
$\dot{(\cdot)}$	First order time derivative
$\ddot{(\cdot)}$	Second order time derivative
$(\cdot) \cdot (\cdot)$	Scalar product
$(\cdot) : (\cdot)$	Double scalar product (or Double contraction)
$(\cdot) \otimes (\cdot)$	Tensor product or dyadic product

Notations

Ω_0	Body in reference configuration
Ω_t	Body in current configuration
Ω_s	Slave body in contact
Ω_m	Master body in contact

Abbreviations

OTM	Optimal Transportation Meshfree
LME	Local Maximum Entropy
FEM	Finite Element Method
ML	Machine Learning
FNN	Feed forword Neural Network
RNN	Reccurrent Neural Network
PODFNN	Proper Orthogonal Decomposition Feed forword Neural Network

Chapter 1

Introduction

1.1 Motivation

Metal cutting is a fundamental manufacturing technology in industrial production, where the material is removed continuously from the workpiece by a cutting tool as shown in Fig. 1.1. The cutting conditions have important influences on the machining precision, such as form accuracy and surface roughness, which determine the quality of the product. To optimize the cutting conditions and improve the machining precision, the analytical modelling, numerical modelling and experimental validation have been investigated extensively. However, it is still a challenge to predict the physical behaviours and the machining performance because of the assumptions in material modelling and the limitations in experimental measurement. With the advent of high-performance computing, robust numerical algorithms and machine learning technology, the traditional manufacturing is being improved to the Intelligent Manufacturing (IM) through the data-integrated numerical simulation approach. In this case, the fast and accurate online prediction of the machining process is required to control and optimize the machining performance.

Figure 1.1. Metal cutting process.

Computational modelling serves as an economic tool to quantitatively understand and pre-

dict complex physical behaviours. Based on the knowledge of computational mechanics, the deformation, separation and temperature rise of the chip can be computed by solving the partial differential equations. This quantitatively predicts the chip size, morphology and temperature which are usually hard to measure in cutting experiments. However, modelling of the metal cutting process has been proved to be particularly complex due to the coupled physical phenomena, such as the extremely large plastic deformation, the adiabatic shear band formation, the ductile fracture and the frictional contact. It is still a challenge to realistically predict the chip morphology, the cutting force as well as the cutting temperature by both experimental and numerical approaches, see VAZ ET AL. (2007). For accurate modelling of the chip formation process, the first key point is to apply appropriate computational methods to cope with the topology change of the body due to large deformation and fracture. At the same time, it is also important to find appropriate material models that can successfully capture related physical behaviours.

In the conventional computational material modelling, the phenomenological material model is proposed based on experimental observations and then calibrated by the specially designed experimental tests. In the past years, tremendous efforts have been made in developing material models and a lot of them have been proposed for metal forming processes, see the review of models by CAO (2017) for instance. However, the proposed models show limitations in generalisation and accuracy in some cases when the model is applied to engineering applications, see e.g. CORONA & REEDLUNN (2013) and CORONA & ORIENT (2014). Thus the phenomenological model has to be modified or the parameters of the model have to be further calibrated by experiments. As a data-driven approach, the machine learning based computational modelling provides an alternative tool to narrow the gap between the experimental data and the numerical modelling for the engineering problems including the metal cutting process. In data-driven simulations, the phenomenological material model proposed based on the experimental observations will be replaced by the machine learning based model. To this end, the machine learning technology is usually applied to learn the material behaviour from the provided experimental data. Machine learning based material models have many advantages over classical numerical approaches, such as the direct utilisation of experimental data and the possibility to improve performances when additional data are available. Another advantage of machine learning based material model is that it can be iteratively improved if more experimental data are available, which yields more flexible and sustainable material descriptions.

The objective of this work is to develop a meshfree numerical simulation framework for the metal cutting process by using both the phenomenological material model and the data-driven material model. The meshfree method will be applied to cope with the large topology change in the chip formation process. The phenomenological material model will be first applied to capture the serrated chip formation process, where the physical quantities, such as the chip size, the cutting force and the cutting temperature will be predicted. Then, the machine learning based material model will be developed as a data-driven approach to replace the phenomenological material model. The data collection strategy and the artificial neural networks based material models will be presented. The effectiveness and the generalisation of the developed model will be verified by the finite element applications.

1.2 State of art

The spatial discretisation in the numerical method plays a major role in capturing the large deformations in metal cutting. The Finite Element Method (FEM) has been applied to simulate the chip formation by MARUSICH & ORTIZ (1995) and LABERGERE ET AL. (2014), where the adaptive remeshing is used to eliminate the deformation induced element distortion. However, the mapping of the state variables from one configuration to the next configuration is required, which leads to an inefficient computation as well as to accumulated numerical errors. As a more flexible and convenient approach for large deformation problems, meshfree methods, such as the Discrete Element Method (DEM) and the Particle Finite Element Method (PFEM), are recently applied to metal cutting simulations, see EBERHARD & GAUGELE (2013), PRIETO ET AL. (2018), RODRIGUEZ PRIETO ET AL. (2016) and SABEL ET AL. (2014) for instance. However, appropriate force laws between particles need to be selected in DEM and the geometric boundary of the body needs to be redefined in PFEM. For a recent review on modelling of metal machining processes, see ARRAZOLA ET AL. (2013) and references therein.
Recently, the Optimal Transportation Meshfree (OTM) method is developed by LI ET AL. (2010) based on the optimal transportation theory, see VILLANI (2003), and the material point sampling concept. Alternatively, a stabilized OTM algorithm is derived by WEISSENFELS & WRIGGERS (2018) from the principle of virtual work. Because of its advantage in algorithmic robustness and its convenience in fracture modelling, the stabilized OTM method is applied as a meshfree numerical method in this work to model the chip formation.
In addition to the numerical method, the material model is of vital importance to describe the complex physical phenomena in metal cutting, such as the adiabatic shear band, the material separation as well as the serrated morphology. The adiabatic shear band is the main cause for the serrated chip formation in high speed machining. The Johnson-Cook flow stress model, see JOHNSON & COOK (1983), has been widely employed as the constitutive law to model the shear band formation in metal cutting processes, such as YE ET AL. (2013) and CALAMAZ ET AL. (2008), where the temperature related term describes a softening effect and drives the shear band formation. To account for the strain softening effect in metal cutting, the modified Johnson-Cook model is developed in SIMA & ÖZEL (2010), in which the strain softening term is added to the Johnson-Cook model in a multiplicative form, see DUCOBU ET AL. (2014).
Another difficulty in modelling of the serrated chip formation is to capture the material separation at the chip root as well as the serrated morphology on the chip upper surface. In the literature, both of the behaviours are treated either by a large plastic deformation approach, see PRIETO ET AL. (2018), RODRIGUEZ PRIETO ET AL. (2016) and SABEL ET AL. (2014), or a separation layer approach, see BÄKER ET AL. (2002), MAHNKEN ET AL. (2013) and MIGUÉLEZ ET AL. (2013). In the first approach, the chip formation is considered as a solid undergoing large deformations without fracture. At the same time, the adaptive remeshing is used to deal with the mesh distortion. In the second approach, a separation layer of finite elements is artificially set on the workpiece where elements will be removed when the critical value of equivalent plastic strain is reached. In this case, no fracture model is implemented on the chip material such that the serrated morphology on the chip upper surface is generated

solely by adiabatic shearing due to thermal softening. Hence, both of the above approaches can not generate a realistically serrated morphology on the chip upper surface. As a well-validated ductile fracture model, such as in CORONA & ORIENT (2014) and ROTH & MOHR (2014), the Johnson-Cook fracture model proposed in JOHNSON & COOK (1985) is applied to predict the fracture in metal cutting as well. However, it shows limitations in capturing the shear band and the fracture on the chip upper surface. Since the phenomenological models have limitations in generalisation or accuracy in some cases, the data-driven material models have been proposed as an alternative approach.

To replace the classical constitutive model in computational mechanics by data-driven modelling, multiple approaches have been proposed in the literature. The model-free data-driven computing paradigm proposed by KIRCHDOERFER & ORTIZ (2016), KIRCHDOERFER & ORTIZ (2018), EGGERSMANN ET AL. (2019) and STAINIER ET AL. (2019), conducts the computing directly from experimental material data under the constraints of conservation laws, which bypasses the empirical material modelling step. This approach works without a constitutive model and seeks to find the closest possible state from a prespecified material data set. A manifold learning approach is proposed by IBAÑEZ ET AL. (2017), IBAÑEZ ET AL. (2018) and IBAÑEZ ET AL. (2019), where the so-called constitutive manifold is constructed from the collected data. A self-consistent clustering approach has been developed to predict the behaviour of heterogeneous materials under inelastic deformation, see LIU ET AL. (2016) and SHAKOOR ET AL. (2019). TANG ET AL. (2019) proposed a mapping approach, in which the one-dimensional data is mapped into three-dimensions for nonlinear elastic material modelling without the construction of an analytic mathematical function for the material law. Since the performance of the data-driven computing is highly determined by the quality and completeness of the available data, the data completion method and data uncertainty problem have been investigated, see AYENSA-JIMÉNEZ ET AL. (2018) and AYENSA-JIMÉNEZ ET AL. (2019). Additionally, the data-driven constitutive model has been developed within a thermodynamic framework based on the so-called GENERIC structure, which guarantees the energy conservation and positive entropy production, see GONZÁLEZ ET AL. (2019a) and GONZÁLEZ ET AL. (2019b).

Apart from the data-driven approaches mentioned above, the artificial neural network as a machine learning approach has been applied to approximate the constitutive model based on data as well, see GHABOUSSI & SIDARTA (1998), HASHASH ET AL. (2004), and LEFIK & SCHREFLER (2003). By the use of the machine learning technology, such as the artificial neural network, see e.g. HASSOUN ET AL. (1995), or the Gaussian Processes, see e.g. RASMUSSEN (2003), constitutive laws behind experimental data can be approximated without postulation of a specific constitutive model. For a review of the machine learning in computational mechanics, see OISHI & YAGAWA (2017) and the references therein.

In order to fit a constitutive material law, the neural network is trained offline using experimental data collected from different loading paths. Afterwards, the network based model is applied online for testing and applications. A nested adaptive neural network has been applied in GHABOUSSI ET AL. (1998), GHABOUSSI & SIDARTA (1998) for modelling the constitutive behaviour of geomaterials. In HASHASH ET AL. (2004), a feed forward neural network based constitutive model is implemented in finite element analysis to capture the nonlinear material behaviour, where the consistent material tangent matrix is derived and

evaluated. The artificial neural network is also applied as an incremental non-linear constitutive model in LEFIK & SCHREFLER (2003) for finite element applications. Furthermore, this approach has been applied to predict the stress-strain curves and texture evolution of polycrystalline metals by ALI ET AL. (2019). Instead of the offline training, the neural network based constitutive model can be trained online by auto-progressive algorithms as well, see PABISEK (2008) and GHABOUSSI ET AL. (1998). Lastly, the artificial neural network have been applied to the heterogeneous material modelling, such as LE ET AL. (2015), LU ET AL. (2019), LI ET AL. (2019), LIU ET AL. (2019) and YANG ET AL. (2019).
The data-driven model free approach conducts calculations directly from the data, which bypasses the model on one hand but highly relies on the quality and completeness of the data on the other hand. The machine learning approaches mentioned above applies the previous strain and stress as history variables, which will introduce extra error for the elastic stage of the inelastic deformation, and hence affect the capabilities to capture the load history in real applications. Additionally, the derivation of the tangent matrix for the neural network based model is inconvenient when changing the network architecture. As a result, there are many issues present in machine learning based material modelling approaches, such as the data collection strategy, the selection of history variables and the applications in finite element analysis.

1.3 Outline of the thesis

This thesis is structured as follows: In Chapter 2, the physical mechanisms of the chip formation process are first analyzed. Then, the fundamentals of continuum mechanics and the governing equations of the metal cutting process are presented in Chapter 3. In Chapter 4, the formulation of the stabilized optimal transportation meshfree method, the contact force imposing and the material point erosion approach for fracture are introduced. Afterwards, the constitutive model within the finite plasticity framework and the ductile fracture model used to capture the serrated chip formation are presented in Chapter 5. In Chapter 6, the serrated chip formation process and the effects of constitutive parameters on the chip morphology are investigated by the numerical simulation results. As a data-driven modelling approach, the machine learning based material modelling framework is developed for both hyperelasticity and plasticity in Chapter 7. Furthermore, the data collection strategy, the history variable selection, and the effectiveness of the developed model are investigated by finite element applications. Finally, the machine learning based plasticity model is applied to the metal cutting simulation in Chapter 8, which is followed by the conclusion and outlook in Chapter 9.

Chapter 2

Metal Cutting Physics

To model the metal cutting process numerically, the physical mechanisms of the chip formation process have to be investigated. As the fundamental cutting technique, the principles of orthogonal cutting is first introduced in this chapter. Then, the analytical model of cutting force is presented, in which the cutting force is computed based on the shear plane model proposed by MERCHANT & ERNST (1941). Afterwards, the cutting temperature and the chip morphology of high speed machining are introduced. Finally, the physical mechanisms of the serrated chip formation process are analyzed based on the experimental observations. The phenomenological material model for capturing the physical mechanisms will be investigated in Chapter 5. The simulation results of the chip formation process in Chapter 6 will in turn explain the physical mechanisms in this chapter, and thus, enable a better understanding for the physical process.

2.1 Orthogonal cutting

Metal cutting is the fundamental machining technique in subtractive manufacturing, such as the turning, milling and drilling. In this process, the material is removed from the workpiece by the cutting tool continuously. According to the relative movement between the cutting tool and the workpiece, the cutting process is distinguished by the orthogonal cutting and the oblique cutting, as shown in Fig. 2.1. In orthogonal cutting, the edge of the cutting tool is perpendicular to the direction of relative motion between workpiece and cutting tool. Due to its simplicity and generalisation, the orthogonal cutting is applied widely in the theoretical, experimental and numerical investigations. This work will focus on the numerical modelling of the orthogonal cutting processes.

In orthogonal cutting, the cutting conditions are defined based on the relative position between the cutting tool and workpiece. As shown in Fig. 2.2, the cutting depth h is defined as the distance between the unmachined and the machined surface. The rake angle γ is the angle between the rake face and a line perpendicular to the machined surface. The angle α between the flank face of the cutting tool and the workpiece is named as clearance angle. The tip of the cutting tool has a round shape with a certain radius. For ductile material such as metals, a positive rake angle is usually applied to perform the cutting operation. The

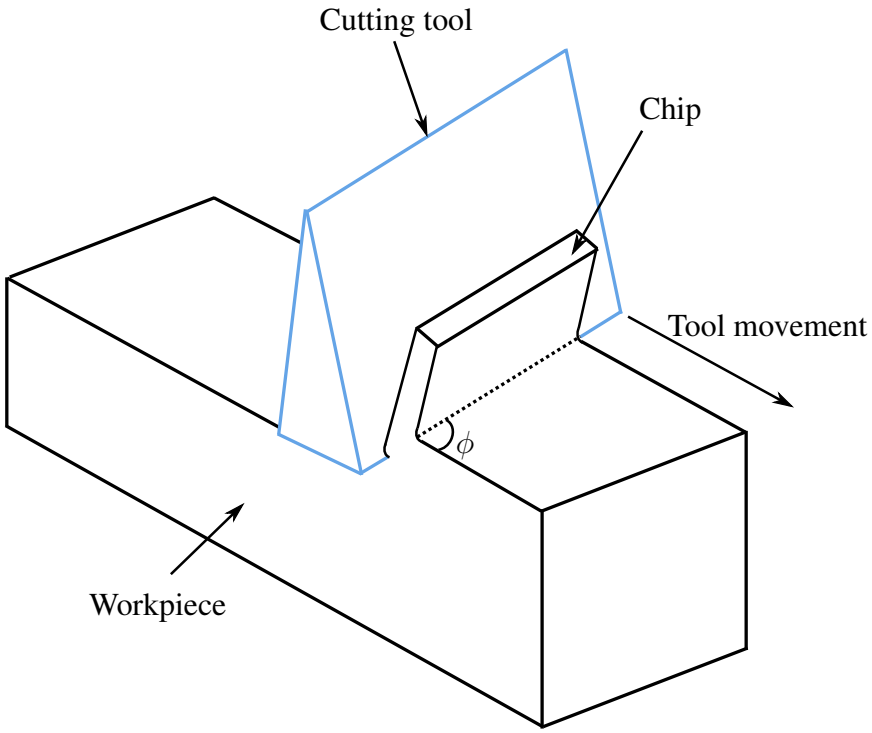


Figure 2.1. Orthogonal cutting ($\phi = 90°$) and oblique cutting ($\phi \neq 90°$).

cutting conditions have large influences on machining performance.
Since the deformations within the cutting zones and the chip are highly non-uniform, the cutting zone can be divided into deformation zones, namely the primary and the secondary deformation zones, see Fig. 2.2. The workpiece material in the primary deformation zone (OAB) undergoes mainly large shear deformation at high strain rates during the cutting process. The chip separation from the workpiece, the shear band formation as well as the chip morphology are determined by the behaviours in the primary deformation zone. There are two regions in the secondary deformation zone (OCD). The sliding region is above the sticking region which is close to the cutting tooltip. The frictional behaviour in the secondary deformation zone has a significant influence on the tool wear and the cutting temperature.

2.2 Analytical model for cutting forces

The most important physical quantity in metal cutting is the cutting force. The investigation on tool wear and machined surface integrity can be carried out according to the variations of cutting forces. Based on the experimental observations and assumptions, a few analytical models of cutting force are proposed for orthogonal cutting in literature. The most popular model in metal cutting mechanics is the shear plane model proposed by MERCHANT & ERNST (1941). Many analytical models of orthogonal cutting are developed based on this fundamental work. In the shear plane model, the chip is assumed to be generated by shearing

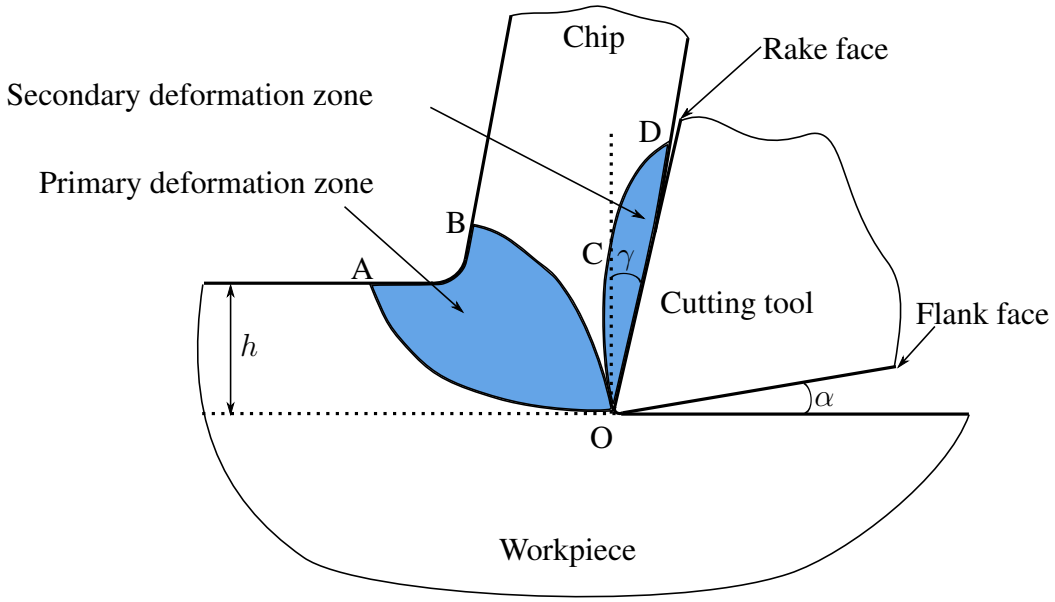


Figure 2.2. Deformation zones in orthogonal cutting.

along the inclined shear plane in the primary deformation zone with an angle of ϕ as shown in Fig. 2.3. According to the Merchant and Ernst's theory, the shear angle can be computed by assuming that the cutting work is minimum. When the shear angle is determined, the cutting force can be calculated based on the cutting conditions including the rake angle and the chip size.

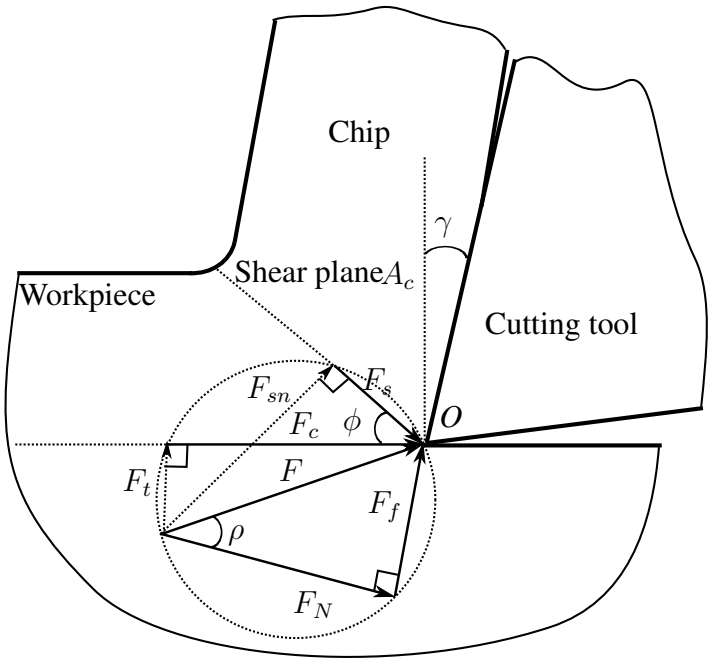


Figure 2.3. Merchant model of cutting force.

According to the shear plane model, the cutting force can be computed from the geometric relationships within the cutting zones. As shown in Fig. 2.3, the resultant force F acting on the cutting tool can be decomposed as the normal force F_N and the friction force F_f by projecting the resultant force F onto the rake face. ρ is the friction angle between chip and tool and defined as

$$\rho = \arctan(F_f/F_N). \tag{2.1}$$

The resultant force F can also be decomposed as the forces F_s (which is along the shear plane) and F_{sn} (which is normal to the shear plane). The shear force F_s can be computed as

$$F_s = F\cos(\phi + \rho - \gamma), \tag{2.2}$$

where ϕ is the angle between the shear plane and the horizontal direction. γ is the rake angle. Based on the theory of shear deformation, the shear force F_s can also be approximated by the shear strength τ_s and the size of chip A_c

$$F_s = \frac{\tau_s A_c}{\sin\phi}, \tag{2.3}$$

where A_c is size of the shear plane within the chip. By combining equation (2.2) and equation (2.3), the resultant force F can be formulated as

$$F = \frac{\tau_s A_c}{\sin\phi}\frac{1}{\cos(\phi + \rho - \gamma)}. \tag{2.4}$$

According to Fig. 2.3, the resultant force F can be decomposed as the cutting force F_c and the thrust force F_t as well. Thus, the cutting force F_c along the cutting direction can be formulated as

$$F_c = F\cos(\rho - \gamma) \tag{2.5}$$

$$= \frac{\tau_s A_c}{\sin\phi}\frac{\cos(\rho - \gamma)}{\cos(\phi + \rho - \gamma)}. \tag{2.6}$$

To minimize the cutting force with respect to the shear angle ϕ, the equation in terms of the angle ϕ can be obtained by taking the derivative of F_c to ϕ and forcing it equal to zero

$$2\phi - \rho + \gamma = \frac{\pi}{2}. \tag{2.7}$$

After the angle ϕ is obtained, the forces mentioned above (i.e. F_c, F_s and F) can be computed accordingly.

2.3 High speed machining

The concept of high speed machining is first introduced in the fundamental tests by C. Salomon in the 1930s. It was found that the cutting temperature would reach the maximum

when the cutting speed increases to a certain value (Fig. 2.4). Upon further increasing the cutting speed, the cutting temperature would start to drop, see Fig. 2.4. Thus the high speed machining usually refers to the machining with a cutting speed beyond this range. Since the cutting heat will be taken away by the chips in high speed machining, the cutting force as well as the thermal effect on the machined surface will be reduced. As a result, it will increase the machined surface integrity and the lifetime of the cutting tool. Additionally, the cooling fluid can be eliminated, which reduces the subsequent pollution to the environment. Due to its advantages, the high speed machining has been widely used in the aerospace industry, automotive industry and so on.

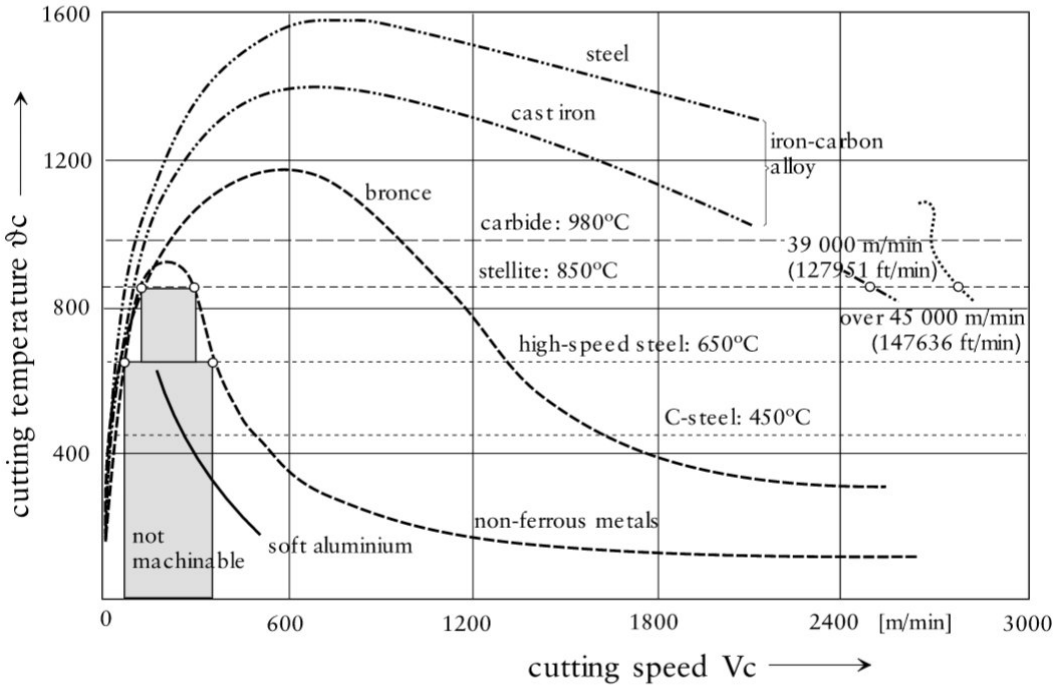


Figure 2.4. Machining temperature in milling at high cutting speeds (taken from SCHULZ (1999)).

During the high speed machining process, large deformations take place at the shear plane in a very short time interval, which leads to a high strain rate in the primal shear zone. Different deformation situations in the primal shear zone will induce different chip morphology, either continuous chip or serrated chip. The magnitude of the deformation will finally be determined by both the thermal diffusivity and ductility of the workpiece material. For instance, the continuous chips will be generated for aluminum alloys because of their high thermal diffusivity and good ductility. However, the serrated chips will be generated for some difficult-to-cut materials, such as the titanium alloys. This work will focus on the modelling of the serrated chip formation process in high speed machining of the titanium alloy.

2.4 Serrated chip formation mechanism

The serrated chip formation mechanisms have been widely studied based on cutting experiments, such as in KOMANDURI & VON TURKOVICH (1981), SUTTER & LIST (2013) and MA ET AL. (2017). As shown in Fig. 2.5, the serrated chip formation process is investigated by the quick stop photomicrograph at different stages of chip formation in GENTE ET AL. (2001). Although the two pictures are taken at different cutting speeds, the chip formation process is still apparent. This is owing to the fact that cutting tool stops at the two important positions, see Fig. 2.5. The colored lines refer to different parts of the boundary of chip segment.

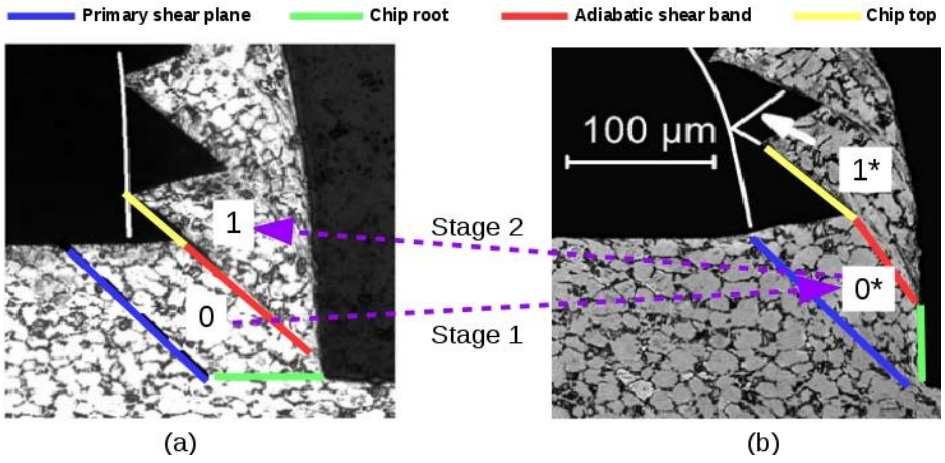


Figure 2.5. Quick stop photomicrograph at different stages of chip formation by GENTE ET AL. (2001).

According to the morphology of the longitudinal cross-section of the generated chips, the formation of one chip segment can be divided into two stages:

- In the first stage, the undeformed chip segment 0 in Fig. 2.5(a) will be highly compressed to the status of chip segment 0* in Fig. 2.5(b). During this stage, the ductile fracture will occur at the root of the chip due to the strain concentration (the green line in Fig. 2.5(a) becomes the green line in Fig. 2.5(b)), which drives the separation of the chip from the workpiece, and the adiabatic shear band will initiate at the chip root.
- In the second stage, the chip segment 0* in Fig. 2.5(b) will slip along the primary shear plane (blue line in 2.5(b)) and finally take its final shape with the same status of the chip segment 1 in Fig. 2.5(a). During this stage, the blue line in 2.5(b) becomes the red and the yellow lines in 2.5(a). That indicates the adiabatic shear band will propagate along the primary shear plane, and the ductile fracture will occur at the top of the shear band on the chip upper surface (the yellow line) when the chip raises to some position. Thus, one segment of the serrated chip is completely generated.

In summary, there will be large plastic deformations, adiabatic shear band formation as well as ductile fracture, both at the root and on the upper surface of the chip, during

the chip formation process. The adiabatic shear band is formed due to the thermal softening. It exhibits large deformation and low strength resistance, which promotes the large slip between two chip segments. The physics of this process has to be captured by the constitutive model in numerical simulation. In this work, a finite plasticity framework, a temperature related thermal softening model and a stress state based ductile fracture model are applied in order to model this process. This model will be introduced in Chapter 5.

Chapter 3

Continuum Mechanics

In continuum mechanics, the material in structures is treated as a continuous medium, where its microscale internal states are characterized by the macroscale field quantities, such as the density, displacement and temperature. The continuum body is imagined as being composed of the material points. The deformation of the body due to external loading can thus be measured by the kinematic variables at each material point. Furthermore, the deformation process of the body is governed by the universal balance laws and material property. To apply the material property into the balance principles, the constitutive model that relates the kinematics and the material responses has to be applied.

3.1 Kinematics

To describe the kinematics of a large deformation process, it is necessary to distinguish the deformation of the body over time by the reference configuration and the current configuration, as shown in Fig. 3.1. The reference configuration refers to the undeformed initial body Ω_0 whereas the current configuration refers to the deformed body Ω_t at time t. The quantities defined in the reference configuration will be denoted by capital letters whereas the quantities defined in the current configuration will be denoted by small letters. As shown in Fig. 3.1, the position vector of a material point P is denoted as $\boldsymbol{X}$ at time $t = 0$ in reference configuration Ω_0 of the body and $\boldsymbol{x}$ in current configuration Ω_t of the body.

The current position of the material point P can be described by a mapping relation from its initial position

$$\boldsymbol{x} = \boldsymbol{\varphi}(\boldsymbol{X}, t) \tag{3.1}$$

for all material points $\boldsymbol{X} \in \Omega_0$ and for all times t, where the vector field $\boldsymbol{\varphi}$ is defined as the motion of the body, see HOLZAPFEL (2002). The motion $\boldsymbol{\varphi}$ can be seen as a one-to-one mapping of the position vector between the two configurations of the body.
The displacement vector $\boldsymbol{u}$ of the material point P is determined by the difference of position vectors

$$\boldsymbol{u} = \boldsymbol{x}(\boldsymbol{X}, t) - \boldsymbol{X}. \tag{3.2}$$

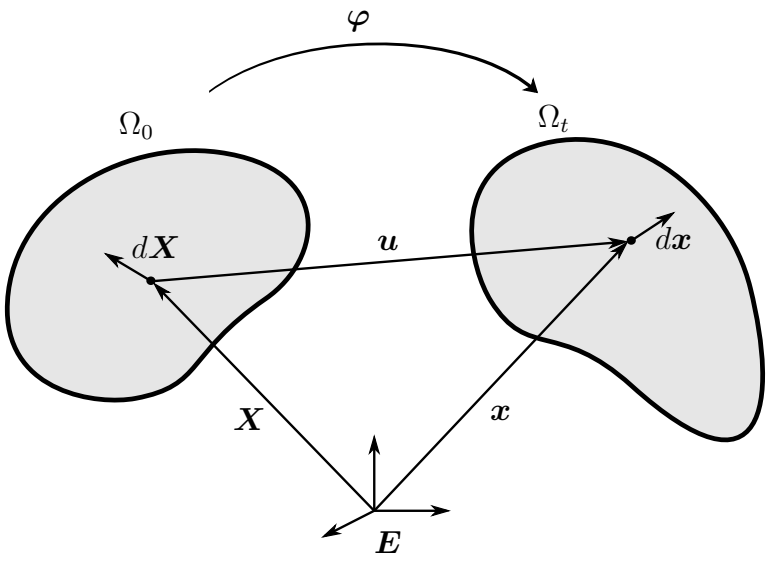


Figure 3.1. Motion of material body Ω.

By taking the time derivatives of displacement vector, the velocity field and the acceleration field can be computed as

$$\boldsymbol{v} = \frac{d\boldsymbol{u}}{dt} = \frac{d\boldsymbol{x}(\boldsymbol{X}, t)}{dt}, \quad \boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \frac{d^2\boldsymbol{x}(\boldsymbol{X}, t)}{dt^2}. \tag{3.3}$$

In continuum mechanics, the deformation of an infinitesimal vector element $d\boldsymbol{X}$ in the body is measured by the deformation gradient $\boldsymbol{F}$ through

$$\boldsymbol{F} = \frac{d\boldsymbol{x}(\boldsymbol{X}, t)}{d\boldsymbol{X}}. \tag{3.4}$$

This fundamental relation describes a mapping of the infinitesimal line element between two distinct configurations. Since points in two configurations are involved, the deformation gradient $\boldsymbol{F}$ is called a two-point tensor. Based on the definition of $\boldsymbol{F}$, the mapping of an infinitesimal volume element can be written as

$$dv = JdV, \quad J = \det(\boldsymbol{F}) > 0, \tag{3.5}$$

where J is known as the Jacobian determinant, dV and dv denote the infinitesimal volume element in the reference and current configurations, respectively. Different with the mapping of infinitesimal line and volume elements, the mapping of the infinitesimal surface element is given by

$$\boldsymbol{n}da = J\boldsymbol{F}^{-T} \cdot \boldsymbol{N}dA, \tag{3.6}$$

in which $\boldsymbol{n}$ and $\boldsymbol{N}$ are the unit normal vectors of the infinitesimal surface elements da in the current configuration and dA in the reference configuration, respectively. Since the rigid body motion is included in the deformation gradient $\boldsymbol{F}$, the strain measures are defined by

the alternative approaches to describe the deformation of the body. A widely used strain measure is the Green-Lagrange strain defined in the reference configuration as

$$\boldsymbol{E} = \frac{1}{2}(\boldsymbol{C} - \boldsymbol{1}), \tag{3.7}$$

where $\boldsymbol{C} = \boldsymbol{F}^T\boldsymbol{F}$ is the right Cauchy-Green tensor defined in the reference configuration. Accordingly, the Euler-Almansi tensor is defined in the current configuration as

$$\boldsymbol{e} = \frac{1}{2}(\boldsymbol{1} - \boldsymbol{b}^{-1}), \tag{3.8}$$

where $\boldsymbol{b} = \boldsymbol{F}\boldsymbol{F}^T$ is the left Cauchy-Green tensor defined in the current configuration.
By applying the polar decomposition, the deformation gradient can be decomposed into the pure stretch and the pure rotation

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{R}, \tag{3.9}$$

where $\boldsymbol{R}$ is the rotation tensor with $\det(\boldsymbol{R}) = 1$, $\boldsymbol{U}$ and $\boldsymbol{V}$ are known as the right and left stretch tensor, respectively. It can be observed that the right and left Cauchy-Green tensors can be computed as

$$\boldsymbol{C} = \boldsymbol{U}^2, \quad \boldsymbol{b} = \boldsymbol{V}^2. \tag{3.10}$$

By using the spectral decomposition, the right and left Cauchy-Green tensors can be reformulated as

$$\boldsymbol{C} = \sum_{i=1}^{3} \lambda_i^2 \boldsymbol{N}_i \otimes \boldsymbol{N}_i, \quad \boldsymbol{b} = \sum_{i=1}^{3} \lambda_i^2 \boldsymbol{n}_i \otimes \boldsymbol{n}_i, \tag{3.11}$$

where λ_i is the eigenvalue characterizing the amount of stretch, $\boldsymbol{N}_i$ and $\boldsymbol{n}_i$ are the eigenvectors.
To study the deformation rate of the continuum body, the material and spatial velocity gradients are defined. The material time derivative of deformation gradient gives the material velocity gradient

$$\dot{\boldsymbol{F}} = \frac{\partial \dot{\boldsymbol{x}}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{v}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}}. \tag{3.12}$$

The spatial velocity gradient is defined by the derivative of the spatial velocity with respect to the spatial coordinate

$$\boldsymbol{l} = \frac{\partial \dot{\boldsymbol{x}}(\boldsymbol{X}, t)}{\partial \boldsymbol{x}} = \dot{\boldsymbol{F}}\boldsymbol{F}^{-1}. \tag{3.13}$$

The rate of deformation tensor is defined by the symmetric part of the spatial velocity gradient

$$\boldsymbol{d} = \frac{1}{2}(\boldsymbol{l} + \boldsymbol{l}^T). \tag{3.14}$$

For a time dependent problem, the time derivatives of strain measures are required. By applying equation (3.13) and equation (3.14), the material time derivative of Green-Lagrange strain yields

$$\dot{\boldsymbol{E}} = \frac{d}{dt}\left[\frac{1}{2}(\boldsymbol{F}^T\boldsymbol{F} - \mathbf{1})\right] = \frac{1}{2}(\dot{\boldsymbol{F}}^T\boldsymbol{F} + \boldsymbol{F}^T\dot{\boldsymbol{F}}) = \boldsymbol{F}^T\boldsymbol{d}\boldsymbol{F}, \tag{3.15}$$

which indicates a *pull back* of the rate of deformation tensor into the reference configuration. The Lie-derivative is introduced for the spatial tensor, e.g. the Euler-Almansi strain tensor $\boldsymbol{e}$, as

$$\mathcal{L}_v(\boldsymbol{e}) = \boldsymbol{F}^{-T}\left[\frac{d}{dt}(\boldsymbol{F}^T\boldsymbol{e}\boldsymbol{F})\right]\boldsymbol{F}^{-1} = \boldsymbol{F}^{-T}\dot{\boldsymbol{E}}\boldsymbol{F}^{-1} = \boldsymbol{d}, \tag{3.16}$$

in which the Euler-Almansi strain is transformed into the reference configuration by a *pull back* operation, which is followed by the material time derivative and the *push forward* into the current configuration again. The Lie-derivative does not involve the differentiation of the basis vectors and leads to an objective time derivative.

3.2 Stress and stress power

The deformation of a continuum body gives rise to the interactions between material parts within the body, which can be described by means of stress. Physically, stress is defined by the internal force per unit area. In continuum mechanics, the internal force is postulated as the force acting on an imaginary internal surface of the deformable body. The Cauchy's stress theorem states that there exist unique second order tensors $\boldsymbol{\sigma}$ and $\boldsymbol{P}$ such that

$$\boldsymbol{t}(\boldsymbol{x}, t) = \boldsymbol{\sigma}(\boldsymbol{x}, t)\boldsymbol{n}, \tag{3.17}$$

$$\boldsymbol{T}(\boldsymbol{X}, t) = \boldsymbol{P}(\boldsymbol{X}, t)\boldsymbol{N}, \tag{3.18}$$

where the $\boldsymbol{T}$ and $\boldsymbol{t}$ are the traction vectors denoting the forces measured per unit surface area in the reference configuration and current configuration respectively, $\boldsymbol{n}$ and $\boldsymbol{N}$ are the unit normal vectors on the surfaces, $\boldsymbol{\sigma}$ is the Cauchy stress tensor defined in the current configuration and $\boldsymbol{P}$ is the first Piola-Kirchhoff stress tensor relating both configurations. The first Piola-Kirchhoff stress tensor is a two-point tensor and can be transformed to the Cauchy stress tensor by

$$\boldsymbol{\sigma} = J^{-1}\boldsymbol{P}\boldsymbol{F}^T. \tag{3.19}$$

A stress tensor defined in the reference configuration is the second Piola-Kirchhoff stress tensor, which does not have a physical interpretation but is very useful,

$$\boldsymbol{S} = \boldsymbol{F}^{-1}\boldsymbol{P} = J\boldsymbol{F}^{-1}\boldsymbol{\sigma}\boldsymbol{F}^{-T}. \tag{3.20}$$

Another stress tensor often used in plasticity is the Kirchhoff stress tensor, which is defined in current configuration as

$$\boldsymbol{\tau} = J\boldsymbol{\sigma}. \tag{3.21}$$

To describe the energy produced by the deformation of the body, the stress power is usually applied, where the rate of internal mechanical work done by internal stresses is defined as the stress power P_{int}, such as in current configuration

$$P_{int} = \int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{d} dv. \tag{3.22}$$

Since the stress power is a scalar and invariant to different configurations, it can be formulated in reference configurations as well

$$P_{int} = \int_{\Omega_0} J\boldsymbol{\sigma} : \boldsymbol{d} dV = \int_{\Omega_0} \boldsymbol{\tau} : \boldsymbol{d} dV = \int_{\Omega_0} \boldsymbol{P} : \dot{\boldsymbol{F}} dV = \int_{\Omega_0} \boldsymbol{S} : \dot{\boldsymbol{E}} dV \tag{3.23}$$

where the strain-stress pairs appeared in the stress power are conjugate variables.

3.3 Balance principles

In this section, the fundamental balance principles which govern the deformation of a continuum body such as the chip formation process are presented.

Conservation of mass
For a closed system, the total mass of the body is constant over time during the deformation process. Thus, the time derivative of total mass of the body is zero

$$\dot{m} = \frac{d}{dt} \int_{\Omega_0} \rho_0 dV = \frac{d}{dt} \int_{\Omega} \rho dv = 0, \tag{3.24}$$

where ρ_0 is the initial density and ρ is the current density. By applying the equation (3.5) to the last term of the above equation, we can obtain

$$\frac{d(\rho J)}{dt} = J(\dot{\rho} + \rho \mathrm{div} \dot{\boldsymbol{x}}) = 0, \tag{3.25}$$

which leads to the local rate form of the continuity mass equation immediately

$$\dot{\rho} + \rho \mathrm{div} \dot{\boldsymbol{x}} = 0. \tag{3.26}$$

Balance of linear momentum
The linear momentum is defined as the product of mass and velocity. The balance principle of linear momentum states that the time derivative of the linear momentum is equal to the external body force and the surface traction, which is formulated in the current configuration as

$$\frac{d}{dt} \int_{\Omega} \rho \dot{\boldsymbol{u}} dv = \int_{\Omega} \rho \hat{\boldsymbol{b}} dv + \int_{\partial\Omega} \boldsymbol{t} da, \tag{3.27}$$

where $\hat{\boldsymbol{b}}$ and $\boldsymbol{t}$ are the body force and surface traction, respectively. By using the Cauchy's stress theorem and the divergence theorem, the integral over surface can be transformed into the volume integral

$$\int_{\partial\Omega} \boldsymbol{t} da = \int_{\partial\Omega} \boldsymbol{\sigma}^T \cdot \boldsymbol{n} da = \int_{\Omega} \mathrm{div}\boldsymbol{\sigma} dv. \tag{3.28}$$

Thus, equation (3.27) can be reformulated as

$$\int_{\Omega} \rho \ddot{\boldsymbol{u}} dv = \int_{\Omega} \rho \hat{\boldsymbol{b}} dv + \int_{\Omega} \mathrm{div}\boldsymbol{\sigma} dv, \tag{3.29}$$

which is also written as

$$\int_{\Omega} (\rho \ddot{\boldsymbol{u}} - \rho \hat{\boldsymbol{b}} - \mathrm{div}\boldsymbol{\sigma}) dv = 0. \tag{3.30}$$

Since the balance law for linear momentum holds for arbitrary volume of the body, the local form reads

$$\rho \ddot{\boldsymbol{u}} - \rho \hat{\boldsymbol{b}} - \mathrm{div}\boldsymbol{\sigma} = 0. \tag{3.31}$$

Rewriting the above strong form into the reference configuration, it reads

$$\rho_0 \ddot{\boldsymbol{u}} - \rho_0 \hat{\boldsymbol{b}} - \mathrm{Div}\boldsymbol{P} = 0. \tag{3.32}$$

Balance of angular momentum

The angular momentum is defined by the cross product of the linear momentum vector and the relative position vector

$$\frac{d}{dt} \int_{\Omega} (\boldsymbol{x} - \boldsymbol{x}_0) \times \rho \dot{\boldsymbol{u}} dv = \int_{\Omega} (\boldsymbol{x} - \boldsymbol{x}_0) \times \rho \hat{\boldsymbol{b}} dv + \int_{\partial\Omega} (\boldsymbol{x} - \boldsymbol{x}_0) \times \boldsymbol{t} da, \tag{3.33}$$

where $\boldsymbol{x}_0$ is a fixed reference position vector. By applying the divergence theorem, the conservation of mass and the balance of linear momentum to the above equation, the balance of angular momentum leads to the constraint of symmetric Cauchy stress tensor, i.e.

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^T. \tag{3.34}$$

Conservation of energy

For a pure mechanical process, the mechanical energy of the system is conserved. By applying the Cauchy's stress theorem $\boldsymbol{t} = \boldsymbol{\sigma}\boldsymbol{n}$, the divergence theorem and the balance of linear momentum to the external mechanical power, the conservation of mechanical energy can be derived as

$$\frac{d}{dt} \underbrace{\int_{\Omega} \frac{1}{2} \rho \boldsymbol{v}^2 dv}_{K} + \underbrace{\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{d} dv}_{P_{int}} = \underbrace{\int_{\Omega} \rho \hat{\boldsymbol{b}} \cdot \boldsymbol{v} dv + \int_{\partial\Omega} \boldsymbol{t} \cdot \boldsymbol{v} da}_{P_{ext}}, \tag{3.35}$$

in which the left hand side contains the rate of kinetic energy K and stress (internal mechanical) power P_{int} whereas the right hand side denotes the external mechanical power P_{ext}. For a general deformation process, the laws of thermodynamics have to be fulfilled. The first law of thermodynamics postulates the balance of total energy

$$\frac{d}{dt}\int_{\Omega}(\frac{1}{2}\rho\boldsymbol{v}^2+e)dv=\int_{\Omega}(\rho\hat{\boldsymbol{b}}\cdot\boldsymbol{v}+r)dv+\int_{\partial\Omega}(\boldsymbol{t}\cdot\boldsymbol{v}-\boldsymbol{q}\cdot\boldsymbol{n})da, \tag{3.36}$$

where e is the specific internal energy per unit volume, r is the heat source per unit volume and time, $\boldsymbol{q}$ is the heat fluxes through surface per unit time. Combing the mechanical energy balance in equation (3.35) with equation (3.36) leads to the reduced form of the energy balance

$$\frac{d}{dt}\underbrace{\int_{\Omega}edv}_{U}=\underbrace{\int_{\Omega}\boldsymbol{\sigma}:\boldsymbol{d}dv}_{P_{int}}+\underbrace{\int_{\Omega}(r-\mathrm{div}\boldsymbol{q})dv}_{Q}, \tag{3.37}$$

which states that the rate of internal energy U is equal to the sum of internal mechanical power P_{int} and the thermal power Q. Since the energy balance holds for arbitrary volume of body, the local form reads

$$\dot{e}=\boldsymbol{\sigma}:\boldsymbol{d}-\mathrm{div}\boldsymbol{q}+r. \tag{3.38}$$

The transformation between the mechanical energy and the thermal energy is governed by the first and second law of thermodynamics.

Entropy inequality

The second law of thermodynamics governs the direction of energy transfer, i.e. total entropy production per unit time is never negative for all of thermodynamic processes. The entropy production per unit time can be described by the change rate of entropy and the thermal power divided by temperature, which leads to

$$\frac{d}{dt}\int_{\Omega}\eta dv+\int_{\partial\Omega}\frac{\boldsymbol{q}}{\theta}\cdot\boldsymbol{n}da-\int_{\Omega}\frac{r}{\theta}dv\geqslant 0, \tag{3.39}$$

where η is the specific entropy per volume in current configuration and θ is temperature. This inequality is known as the Clausius-Duhem inequality. By applying the first law of thermodynamics in equation (3.37) to the above inequality and neglecting the heat source, the local Clausius-Planck inequality in the current configuration is obtained

$$D_{int}=\boldsymbol{\sigma}:\boldsymbol{d}-\dot{e}+\theta\dot{\eta}\geqslant 0, \tag{3.40}$$

in which D_{int} is known as the internal dissipation. By the use of the Legendre transformation to the specific Helmholz free energy ψ per unit deformed volume

$$\psi=e-\theta\eta, \tag{3.41}$$

the second law of thermodynamics can be reformulated in terms of the Helmholtz free energy as

$$D_{int} = \boldsymbol{\sigma} : \boldsymbol{d} - \dot{\psi} - \dot{\theta}\eta \geqslant 0. \tag{3.42}$$

Additionally, by applying the Legendre transformation to equation (3.38), the first law of thermodynamics transforms to

$$\dot{\psi} = \boldsymbol{\sigma} : \boldsymbol{d} - \mathrm{div}\boldsymbol{q} + r - \eta\dot{\theta} - \theta\dot{\eta}. \tag{3.43}$$

Chapter 4

Optimal Transportation Meshfree (OTM) Method

To cope with the topology change of the body due to large deformation and fracture in the metal cutting process, the meshfree numerical solution scheme is applied in this work. As a recently developed Galerkin type meshfree approximation scheme, the Optimal Transportation Meshfree (OTM) method shows good performances to simulate the large deformation and fracture problems, see LI ET AL. (2010) and LI ET AL. (2015).
In the original OTM method, the kinetic energy term in the formulation of Hamilton's principle is treated by the optimal transportation theory, see VILLANI (2003) and VILLANI (2008), where the Euler-Lagrangian equations for each nodal point are obtained after time and spatial discretisations. The stabilized OTM formulation is derived from the weak form in an updated Lagrangian framework by WEISSENFELS & WRIGGERS (2018). Because of its advantage in algorithmic robustness and its convenience in fracture modelling, the stabilized OTM method is applied as a meshfree solution scheme in this work to model the chip formation.

4.1 The weak form

To solve the Partial Differential Equations (PDEs) defined in the balance principles, the initial boundary value problem (IBVP) is first specified. The deformation of the body is governed by the balance law of linear momentum

$$\rho\ddot{\boldsymbol{u}} - \rho\hat{\boldsymbol{b}} - \mathrm{div}\boldsymbol{\sigma} = 0, \tag{4.1}$$

where displacement $\boldsymbol{u}$ is the primary variable and $\hat{\boldsymbol{b}}$ is the prescribed body force. The initial conditions, the Dirichlet boundary conditions and the Neumann boundary conditions for the body are given respectively by

$$\boldsymbol{u}(t=0) = \boldsymbol{u}_0 \ \text{ in } \Omega, \tag{4.2}$$
$$\boldsymbol{v}(t=0) = \boldsymbol{v}_0 \ \text{ in } \Omega, \tag{4.3}$$
$$\boldsymbol{u} = \hat{\boldsymbol{u}} \ \text{ on } \partial_u\Omega, \tag{4.4}$$
$$\boldsymbol{\sigma}\boldsymbol{n} = \hat{\boldsymbol{t}} \ \text{ on } \partial_t\Omega. \tag{4.5}$$

By multiplying the differential equation (4.1) with a test function $\delta\boldsymbol{u}$ and integrating over the domain, the weak form of the differential equation can be obtained as

$$\int_\Omega \delta\boldsymbol{u} \cdot (\rho\ddot{\boldsymbol{u}} - \rho\hat{\boldsymbol{b}} - \mathrm{div}\boldsymbol{\sigma})dv = 0. \tag{4.6}$$

By applying $\mathrm{div}(\delta\boldsymbol{u} \cdot \boldsymbol{\sigma}) = \delta\boldsymbol{u} \cdot \mathrm{div}\boldsymbol{\sigma} + \mathrm{grad}\delta\boldsymbol{u} : \boldsymbol{\sigma}$, we obtain

$$\int_\Omega \delta\boldsymbol{u} \cdot \rho\ddot{\boldsymbol{u}}dv + \int_\Omega \mathrm{grad}\delta\boldsymbol{u} : \boldsymbol{\sigma}dv = \int_\Omega \delta\boldsymbol{u} \cdot \rho\hat{\boldsymbol{b}}dv + \int_\Omega \mathrm{div}(\delta\boldsymbol{u} \cdot \boldsymbol{\sigma})dv. \tag{4.7}$$

By using the divergence theorem to the last term of the above equation and by splitting the boundary into force boundary $\partial_t\Omega$ and displacement boundary$\partial_u\Omega$, it results in the weak form of equilibrium in the current configuration, see WRIGGERS (2008),

$$\int_\Omega \delta\boldsymbol{u} \cdot \rho\ddot{\boldsymbol{u}}dv + \int_\Omega \mathrm{grad}\delta\boldsymbol{u} : \boldsymbol{\sigma}dv = \int_\Omega \delta\boldsymbol{u} \cdot \rho\hat{\boldsymbol{b}}dv + \int_{\partial_t\Omega} \delta\boldsymbol{u} \cdot \hat{\boldsymbol{t}}da, \tag{4.8}$$

where the displacements $\boldsymbol{u}$ are the primal variables. The density, the gravity acceleration and the Cauchy stress correspond to ρ, $\hat{\boldsymbol{b}}$, and $\boldsymbol{\sigma}$, respectively. The surface traction $\hat{\boldsymbol{t}}$ is prescribed at the force boundary.
Note that the strong form in equation (4.1) involves the computation of second derivatives of displacement since stress is computed from a constitutive equation in terms of strain. By performing integration by parts in equation (4.6), the second derivatives are reduced to the first derivatives in the weak form in equation (4.8).

4.2 Spatial discretisation

As shown in Fig. 4.1, the spatial domain of the body is discretized by two sets of points in OTM: the material points, which are used as integration points, and the nodal points, which carry the position information of the body. To relate a material point with the nearest nodal points, the support domain is constructed on the material point. In this way, the nodal point is involved in multiple support domains of material points. At the same time, the material points related to a nodal point construct an influence domain for this nodal point as shown in Fig. 4.1.

At each material point, the displacements and the test functions are approximated based on the nodal values within its support domain using proper shape functions

$$\boldsymbol{u}_p = \sum_{I=1}^{n_{np}} N_I(\boldsymbol{x}_p)\boldsymbol{u}_I, \quad \delta\boldsymbol{u}_p = \sum_{I=1}^{n_{np}} N_I(\boldsymbol{x}_p)\delta\boldsymbol{u}_I, \quad \mathrm{grad}(\delta\boldsymbol{u}_p) = \sum_{I=1}^{n_{np}} \boldsymbol{B}_I(\boldsymbol{x}_p)\delta\boldsymbol{u}_I, \tag{4.9}$$

where n_{np} specifies the number of nodes within the support domain of material point p, $N_I(\boldsymbol{x}_p)$ is the shape function of material point p evaluated as node I, and the matrix $\boldsymbol{B}_I(\boldsymbol{x}_p)$ contains the derivatives of the shape functions at node I. The support domain is updated in

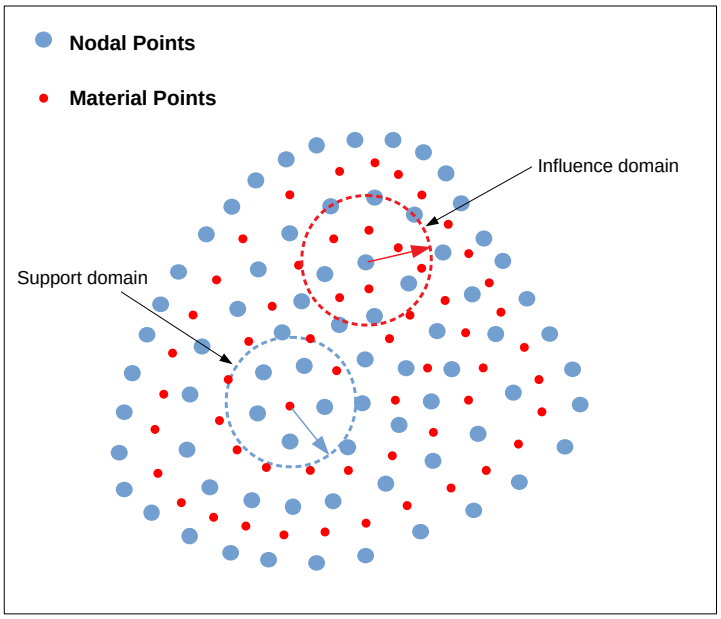


Figure 4.1. Spatial discretisation by material points and nodal points in OTM.

every time step, thus the element distortion can be eliminated during the large deformation process.

By applying equations (4.9), the weak form in equation (4.8) can be discretized by the assemble of material points

$$\underbrace{\left[\bigcup_{p=1}^{n_{mp}}\sum_{I}^{n_{np}}\sum_{J}^{n_{np}} N_I(\boldsymbol{x}_p)\mathbf{1}N_J(\boldsymbol{x}_p)m_p\right]}_{\boldsymbol{M}}\cdot\ddot{\boldsymbol{u}} = \underbrace{\bigcup_{p=1}^{n_{mp}}\sum_{I}^{n_{np}}\left[N_I(\boldsymbol{x}_p)\hat{\boldsymbol{b}}_p m_p - \boldsymbol{B}_I^T(\boldsymbol{x}_p)\boldsymbol{\sigma}_p v_p\right]}_{\boldsymbol{f}-\boldsymbol{P}(\boldsymbol{u})}, \tag{4.10}$$

where n_{mp} is the total number of material points in the body, $\ddot{\boldsymbol{u}}$ is the global nodal acceleration vector, m_p denotes the mass of the material point p and v_p its current volume. To guarantee the conservation of total mass, the mass of a material point is assumed to be constant during the calculation. It is worth noting that $\bigcup_{p=1}^{n_{mp}}$ is used here instead of $\sum$, which denotes the global assembly process of all material point contributions. At the same time, since the overlap of support domain is allowed in OTM, there are more relationships between the domains of material point.

The above discretized formulation can be abbreviated as

$$\boldsymbol{M}\ddot{\boldsymbol{u}} = \boldsymbol{f} - \boldsymbol{P}(\boldsymbol{u}), \tag{4.11}$$

where $\boldsymbol{M}$ stands for the consistent mass matrix, $\boldsymbol{f}$ contains the prescribed body force and $\boldsymbol{P}$ denotes the internal force vector. This discretized dynamic equation will be solved by updating the nodal point data and the material point data.

4.3 Time integration

To solve the dynamic problem in equation (4.11), a proper time integration scheme has to be selected. In this work, the central difference time integration is applied. The central difference time integration can be derived from the explicit Newmark scheme, which will be introduced in this part.

Explicit Newmark method

The Newmark method relates the variables at time t_{n+1} with those at time t_n by

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \boldsymbol{v}_n + (\frac{1}{2} - \beta)\Delta t^2 \boldsymbol{a}_n + \beta \Delta t^2 \boldsymbol{a}_{n+1}, \tag{4.12}$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + (1-\gamma)\Delta t \boldsymbol{a}_n + \gamma \Delta t \boldsymbol{a}_{n+1}, \tag{4.13}$$

where β and γ are the parameters. The explicit Newmark scheme is recovered by setting $\beta = 0$.

Central difference scheme

Following the explicit scheme, the central difference time integration can be further obtained by setting $\gamma = 0.5$ in the above relations. The displacements at time step t_n and t_{n+1} can thus be updated as

$$\boldsymbol{u}_n = \boldsymbol{u}_{n-1} + \Delta t \boldsymbol{v}_{n-1} + \frac{1}{2}\Delta t^2 \boldsymbol{a}_{n-1}, \tag{4.14}$$

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \boldsymbol{v}_n + \frac{1}{2}\Delta t^2 \boldsymbol{a}_n. \tag{4.15}$$

The velocity at time step t_n can be updated as

$$\boldsymbol{v}_n = \boldsymbol{v}_{n-1} + \frac{(\boldsymbol{a}_n + \boldsymbol{a}_{n-1})\Delta t}{2}. \tag{4.16}$$

Rewrite equation (4.14), we can get

$$\boldsymbol{v}_{n-1} + \frac{\Delta t}{2}\boldsymbol{a}_{n-1} = \frac{1}{\Delta t}(\boldsymbol{u}_n - \boldsymbol{u}_{n-1}). \tag{4.17}$$

Insert equation (4.17) to equation (4.16), it reads

$$\Delta t \boldsymbol{v}_n = \boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \frac{\Delta t^2}{2}\boldsymbol{a}_n. \tag{4.18}$$

Combine equation (4.18) and equation (4.15), the velocity at time t_n can be represented in terms of displacements at time t_{n+1} and t_{n-1}

$$\boldsymbol{v}_n = \frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_{n-1}}{2\Delta t}. \tag{4.19}$$

Insert equation (4.19) to equation (4.18), the acceleration at time t_n can be represented as

$$\boldsymbol{a}_n = \frac{(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n) - (\boldsymbol{u}_n - \boldsymbol{u}_{n-1})}{\Delta t^2}. \tag{4.20}$$

Time integration in OTM

To apply the central difference time integration, one approach is inserting the acceleration $\boldsymbol{a}_n$ in equation (4.20) to the discretized form in equation (4.10) at time t_n, where the displacements at time t_{n+1} will be the unknown variables. Another approach is computing the acceleration $\boldsymbol{a}_n$ at time t_n first by solving

$$\boldsymbol{M}\boldsymbol{a}_n = \boldsymbol{f}_n - \boldsymbol{P}(\boldsymbol{u}_n). \tag{4.21}$$

Then, the displacement and velocity at time t_{n+1} can be updated by using the acceleration $\boldsymbol{a}_n$. Rewrite equation (4.20), the displacement at time t_{n+1} reads

$$\boldsymbol{u}_{n+1} = 2\boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \Delta t^2 \boldsymbol{a}_n. \tag{4.22}$$

When $n = 0$, the special initialisation is required in equation (4.22). Based on the second order accurate Taylor series expansion, the displacement $\boldsymbol{u}_{-1}$ can be obtained using the initial values

$$\boldsymbol{u}_{-1} = \boldsymbol{u}_0 - \Delta t \boldsymbol{v}_0 + \frac{\Delta t^2}{2}\boldsymbol{a}_0. \tag{4.23}$$

To efficiently implement the central difference time integration scheme in equation (4.22), a current velocity is defined as

$$\hat{\boldsymbol{v}}_{n+1} = \begin{cases} \boldsymbol{v}_n + \frac{\Delta t}{2}\boldsymbol{a}_n, n = 0 \\ \hat{\boldsymbol{v}}_n + \Delta t \boldsymbol{a}_n, n \geqslant 1. \end{cases} \tag{4.24}$$

Accordingly, the displacement at time t_{n+1} is updated as

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \hat{\boldsymbol{v}}_{n+1}. \tag{4.25}$$

The accuracy of the central difference time integration is of second order. This explicit scheme is only conditionally stable with limited time step size $\Delta t < \Delta t_{crit}$, where Δt_{crit} is determined by the smallest time interval that wave propagates across the element. For a detailed investigation about the time increment limit, see MIMOUNA & TCHELEPI (2019). In this work, the critical time increment is determined by

$$\Delta t_{crit} = \Delta l \sqrt{\frac{E(1-\nu)}{\rho(1-2\nu)(1+\nu)}}, \tag{4.26}$$

where Δl is the characteristic length that can be assumed as the minimal distance between the nodes, E is the Young's modulus and ν is the Poison's ratio.

4.4 Update of the nodal point data

If the classical consistent mass matrix in equation (4.10) is maintained, it still needs to solve a algebraic equation system in this explicit scheme, which obviously conflicts with the primary

advantage of using explicit schemes. To calculate the acceleration efficiently, the concept of a lumped mass matrix is applied to the equilibrium equation, see HUGHES (1987). By using of the row-sum technique, the full mass matrix $\boldsymbol{M}$ becomes a diagnosed matrix, which leads to a direct computation of nodal mass, see WEISSENFELS & WRIGGERS (2018),

$$m_I^n = \sum_p^{n_{mp}^I} N_I^n(\boldsymbol{x}_p^n) m_p, \tag{4.27}$$

where m_I^n is the mass at node I, n_{mp}^I is the number of material point related to the node I within its influence domain. Finally the global equation system (4.11) can be decomposed into a set of independent nodal equations

$$m_I^n \boldsymbol{a}_I^n = \boldsymbol{r}_I^n, \tag{4.28}$$

in which the nodal residual vector $\boldsymbol{r}_I^n$ can be formulated as

$$\boldsymbol{r}_I^n = \sum_p^{n_{mp}^I} \left[N_I^n(\boldsymbol{x}_p^n) \hat{\boldsymbol{b}}_p^n m_p - \boldsymbol{B}_I^{nT}(\boldsymbol{x}_p^n) \boldsymbol{\sigma}_p^n v_p \right]. \tag{4.29}$$

In order to penalize the inaccurate behaviour within each single support domain, a stabilisation term is added to the nodal residual vector, see WEISSENFELS & WRIGGERS (2018),

$$\boldsymbol{r}_I^{n-stab} = \boldsymbol{r}_I^n - \epsilon \sum_p^{n_{mp}^I} N_I^n(\boldsymbol{x}_p^n) \boldsymbol{e}_{I,p}^n, \tag{4.30}$$

where the last term on the right hand side enforces the error to be zero by the penalty regularisation. $\boldsymbol{e}_{I,p}^n$ is the error due to underintegration and calculated based on the positions of nodal point I and material point p by

$$\boldsymbol{e}_{I,p}^n = \frac{\boldsymbol{x}_I^n - \boldsymbol{x}_p^n - (\tilde{\boldsymbol{x}}_I^n - \tilde{\boldsymbol{x}}_p^n)}{\|\boldsymbol{x}_I^{n-1} - \boldsymbol{x}_p^{n-1}\|}, \tag{4.31}$$

where $\boldsymbol{x}_I^n - \boldsymbol{x}_p^n$ is the distance vector between nodal point I and material point p computed from OTM algorithm, $\tilde{\boldsymbol{x}}_I^n - \tilde{\boldsymbol{x}}_p^n$ is the distance vector updated by the constant increment of deformation gradient $\Delta \boldsymbol{F}_p^n$

$$\tilde{\boldsymbol{x}}_I^n - \tilde{\boldsymbol{x}}_p^n = \Delta \boldsymbol{F}_p^n (\boldsymbol{x}_I^{n-1} - \boldsymbol{x}_p^{n-1}). \tag{4.32}$$

The difference of the distance vectors between two formulations characterizes the error due to underintegration in equation (4.31). After calculating the nodal acceleration $\boldsymbol{a}_I^n$ by equation (4.28), the nodal displacement $\boldsymbol{u}_I^{n+1}$ is updated according to equation (4.25). Finally, the nodal coordinates at time t_{n+1} are computed as

$$\boldsymbol{x}_I^{n+1} = \boldsymbol{x}_I^0 + \boldsymbol{u}_I^{n+1}, \tag{4.33}$$

where $\boldsymbol{x}_I^0$ are the initial nodal coordinates.

4.5 Update of the material point data

The position of the material point p at the time step $n+1$ is updated by the shape functions at time step n and the nodal point coordinates at time step $n+1$ within its support domain

$$\boldsymbol{x}_p^{n+1} = \sum_I^{n_{np}^p} N_I^n(\boldsymbol{x}_p^n)\boldsymbol{x}_I^{n+1}, \tag{4.34}$$

where n_{np}^p is the number of nodal points within the support domain of material point p. The support domain will be updated at each time step, thus the number of the nodal point with the support domain is changing in the large deformation.

To capture the kinematic information in the large deformation, the updated Lagrangian formulation is employed in the OTM framework. The deformation gradient of material point p is updated in terms of the current and the incremental value of deformation gradient as

$$\boldsymbol{F}_p^{n+1} = \Delta\boldsymbol{F}_p^{n+1}\boldsymbol{F}_p^n, \tag{4.35}$$

in which the incremental deformation gradient can be calculated based on the nodal displacement increments

$$\Delta\boldsymbol{F}_p^{n+1} = \mathbf{1} + \sum_I^{n_{np}^p} \frac{\partial N_I^n(\boldsymbol{x}_p^n)}{\partial \boldsymbol{x}_p^n}\Delta\boldsymbol{u}_I^{n+1}. \tag{4.36}$$

The volume and density at material point p are updated accordingly based on the increment of deformation gradient at material point $\Delta\boldsymbol{F}_p^{n+1}$

$$v_p^{n+1} = \det(\Delta\boldsymbol{F}_p^{n+1})v_p^n, \quad \rho^{n+1} = \frac{m_p}{v_p^{n+1}}. \tag{4.37}$$

4.6 Local maximum entropy shape function

In meshfree methods, the basis function has to be constructed for an arbitrary number of nodes with arbitrary locations. The Local Maximum Entropy (LME) approximation function, see ARROYO & ORTIZ (2006), is taken as a basis for the numerical solution of PDEs in the style of meshfree Galerkin in OTM. This approximation scheme continuously bridges two important limits: the Delaunay triangulation and the maximum entropy statistical inference. It belongs to the convex approximation schemes which are based on shape functions that are positive and interpolate affine functions exactly. An important property of the LME shape function is the weak Kronecker-delta property at the boundary.

Given a fixed material point $\boldsymbol{x}_p$ and its associated nodal points in the support domain, the

LME shape function $N_I(\boldsymbol{x}_p)$ can be obtained by solving a constrained optimisation problem

$$\text{Minimum: } f_\beta[N_I(\boldsymbol{x}_p)] = \beta U[N_I(\boldsymbol{x}_p)] - H[N_I(\boldsymbol{x}_p)], \tag{4.38}$$

$$\text{Subject to: } \quad N_I(\boldsymbol{x}_p) \geqslant 0, \tag{4.39}$$

$$\sum_I^{n_{np}} N_I(\boldsymbol{x}_p) = 1, \tag{4.40}$$

$$\sum_I^{n_{np}} N_I(\boldsymbol{x}_p)\boldsymbol{x}_I = \boldsymbol{x}_p, \tag{4.41}$$

where β is a parameter to control the degree of locality, $U[N_I(\boldsymbol{x}_p)]$ is the function to describe the locality

$$U[N_I(\boldsymbol{x}_p)] = \sum_I^{n_{np}} N_I(\boldsymbol{x}_p)|\boldsymbol{x}_p - \boldsymbol{x}_I|^2, \tag{4.42}$$

and $H[N_I(\boldsymbol{x}_p)]$ is the function to describe the entropy

$$H[N_I(\boldsymbol{x}_p)] = -\sum_i^{n_{np}} N_I(\boldsymbol{x}_p)\log N_I(\boldsymbol{x}_p). \tag{4.43}$$

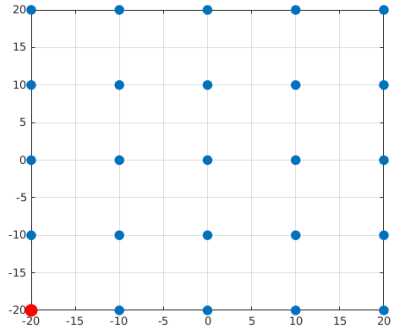
By enforcing the first order completeness condition (4.41) using the Lagrangian multiplier method and the partition of unity condition (4.40) with normalisation, the unique solution of $N_I(\boldsymbol{x}_p)$ for this optimisation problem is, see ARROYO & ORTIZ (2006),

$$N_I(\boldsymbol{x}_p) = \frac{Z_I(\boldsymbol{x}_p)}{Z}, \quad Z_I = e^{-\beta|\boldsymbol{x}_p - \boldsymbol{x}_I|^2 + \lambda(\boldsymbol{x}_p - \boldsymbol{x}_I)}, \quad Z = \sum_I^{n_{np}} Z_I, \tag{4.44}$$

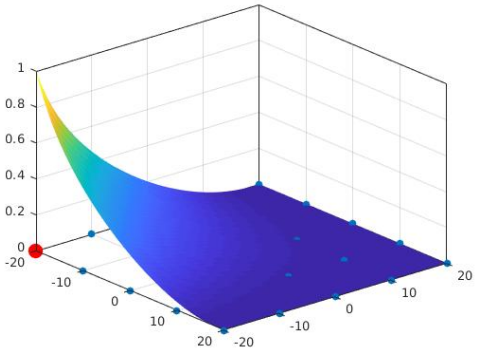
where λ is the Lagrangian multiplier and can be computed by solving the constraint of first order completeness: $\mathbf{r}(\mathbf{x}, \lambda) = \sum_I N_I(\mathbf{x})(\mathbf{x} - \mathbf{x}_I) = \mathbf{0}$ within a local Newton-Raphson scheme. The parameter β is calculated by $\beta = \frac{\gamma}{h^2}$, where γ is suggested to be in the range of 0.8 to 4, and h is the characteristic nodal spacing.
For the nodal point set as shown in Fig. 4.2, the LME shape function at nodal point $(-20, -20)$ is evaluated within the domain by setting different values of β as shown in Fig. 4.2. It can be seen that β controls the degree of locality of the shape function.

4.7 Imposing contact forces in OTM

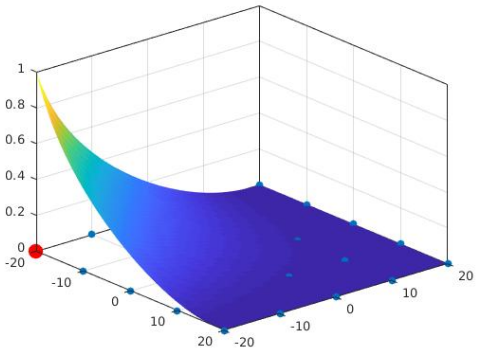
Since the frictional contact between the cutting tool and workpiece plays an essential role in the metal cutting process, the contact forces have to be computed within the OTM scheme. In contact mechanics, see WRIGGERS (2006), the surfaces involved in the contact interface are distinguished by the master surface and the slave surface as shown in Fig. 4.3. During



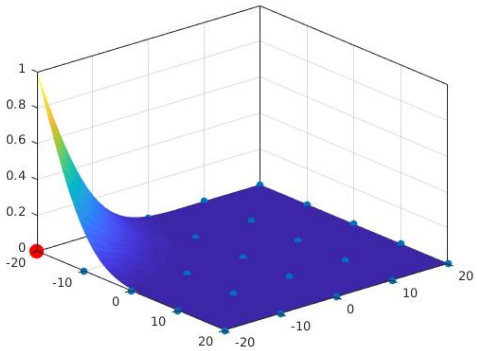
(a) Nodal points in a support domain



(b) $\beta = 10^{-10}$



(c) $\beta = 10^{-6}$



(d) $\beta = 10^{-2}$

Figure 4.2. The support domain and the LME shape functions at node $(-20, -20)$.

the contact process, the contact surfaces have to fulfill the non-penetration condition, which is defined by projecting the slave node onto the master surface

$$g^N = (\boldsymbol{x}_s - \boldsymbol{x}_m) \cdot \boldsymbol{n}_m \geq 0, \tag{4.45}$$

where g^N is the normal gap, $\boldsymbol{x}_s$ is the coordinate of slave node, $\boldsymbol{x}_m$ is the orthogonal projection of $\boldsymbol{x}_s$ on the master surface Γ_m, and $\boldsymbol{n}_m$ is the normal vector associated with the master body.

By using the penalty method, the normal contact force and the stick tangential contact force are determined as

$$\boldsymbol{t}^N = c^N \boldsymbol{g}^N, \quad \boldsymbol{t}^T = c^T \boldsymbol{g}^T, \tag{4.46}$$

where c^N and c^T are the penalty parameters. In the tangential direction, one has to distinguish between stick and slip. In the stick case, the relative displacement $\boldsymbol{g}^T$ and relative velocity $\dot{\boldsymbol{g}}^T$

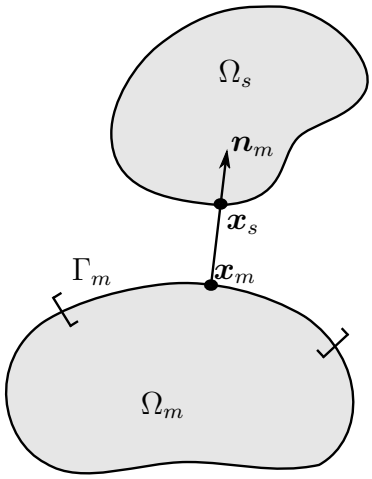


Figure 4.3. The contact between cutting tool and workpiece.

in tangential direction have to be zero. In the slip state, the Columb friction law is assumed to determine tangential contact force

$$\boldsymbol{t}^T = -\mu t^N \frac{\dot{\boldsymbol{g}}^T}{\|\dot{\boldsymbol{g}}^T\|}, \tag{4.47}$$

where μ is the frictional coefficient. The contact forces for the two deformable bodies are identical in the value but opposite in the direction. More details on computational contact modelling can be found in WRIGGERS (2006).
In this work, the cutting tool is treated as the master body and the workpiece is the slave body. The gap can be obtained by projecting the nodes of workpiece onto the tool surface. In the explicit OTM scheme, the predictor-corrector strategy is applied to impose the contact constraint. If the nodal residual $\boldsymbol{r}_I^n$ of equation (4.30) violated the non-penetration condition, the contact force is imposed onto the nodal residual of this active node. By choosing the penalty parameter as $c^N = \frac{m}{\Delta t^2}$ in equation (4.46), the normal contact force is given by

$$\boldsymbol{t}_I^N = \frac{m_I}{\Delta t^2} \boldsymbol{g}_I^N. \tag{4.48}$$

When there is normal penetration of $\boldsymbol{g}_I^N$ at node I, the nodal displacement corrected by this normal contact force $\boldsymbol{t}_I^N$ will be equal to $\boldsymbol{g}_I^N$ within one time step, which means the normal penetration will be zero after the correction. For active nodes, either stick or slip occurs in the tangential direction. The stick contact force enforces the relative velocity to be zero

$$\boldsymbol{t}_{I,stick}^T = m_I \frac{\boldsymbol{v}_I^{n+1} - \boldsymbol{v}_m}{\Delta t}, \tag{4.49}$$

where $\boldsymbol{v}_m$ is the velocity of the master surface (cutting tool surface) and $\boldsymbol{v}_I^{n+1}$ is the velocity of node I. The frictional force in slip case is determined based on a constitutive law, such as the Columb friction law

$$\boldsymbol{t}_{I,slip}^T = \mu \|\boldsymbol{t}_I^N\| \boldsymbol{n}_I^T, \tag{4.50}$$

where the unit directional vector $\boldsymbol{n}_I^T$ is calculated as $\frac{\boldsymbol{t}_{I,stick}^T}{\|\boldsymbol{t}_{I,stick}^T\|}$. The tangential force exerted by the master surface on the slave nodal point is finally determined by

$$\boldsymbol{t}_I^T = \min(\boldsymbol{t}_{I,slip}^T, \boldsymbol{t}_{I,stick}^T). \tag{4.51}$$

It is worth noting that the stick force calculated above depends on the time increment Δt. If very small time steps are used, the magnitude of the stick force will be very large and thus the stick case will not occur. However, the relative velocity will be decreased during slip and the stick will occur after some steps, which compensates the dependence of time step. The total contact force is given by

$$\boldsymbol{t}_I = \boldsymbol{t}_I^N + \boldsymbol{t}_I^T. \tag{4.52}$$

By imposing the total contact force $\boldsymbol{t}_I$ to the nodal residual $\boldsymbol{r}_I^n$, the nodal acceleration will be updated as

$$\boldsymbol{a}_I^n = \hat{\boldsymbol{a}}_I^n + \frac{\boldsymbol{t}_I}{m_I}, \tag{4.53}$$

where $\hat{\boldsymbol{a}}_I^n$ is the nodal acceleration obtained by the original nodal residual in the predictor stage, see equation (4.30).

4.8 Fracture modelling in OTM

Ductile fracture is a fundamental mechanism in the metal cutting process, thus modelling of fracture in the OTM discretisation has to be investigated. Since the support domain is constructed on each material point in OTM, the overlap between the neighbouring support domains are induced. It would lead to difficulties in modelling of crack propagation in this discretisation. There are two approaches that can be applied to resolve the fracture geometry within the domain, one is the nodal point split approach where crack propagates along with nodal points and the other is the material point split approach where crack propagates along with material points. Both of the approaches require the update of support domain such that the relationship between material points and nodal points from both sides of a crack can be dismissed.

As a material point split approach, the material point erosion, see PANDOLFI ET AL. (2013) and LI ET AL. (2012), treats the fracture at material point by a regional failure within an ε-neighbourhood of this material point as shown in Fig. 4.4.

The material point erosion is an extension of eigenerosion in finite element method (PANDOLFI & ORTIZ, 2012) to OTM, and the eigenerosion stems from the theory of eigenfracture by SCHMIDT ET AL. (2009). In material point erosion, the crack will propagate along with the material points located in the neighbourhood of material point. The free energy cost at material point p for fracture is approximated by

$$E_p = G_c \Delta A = G_0 \frac{V_\varepsilon}{2\varepsilon}, \tag{4.54}$$

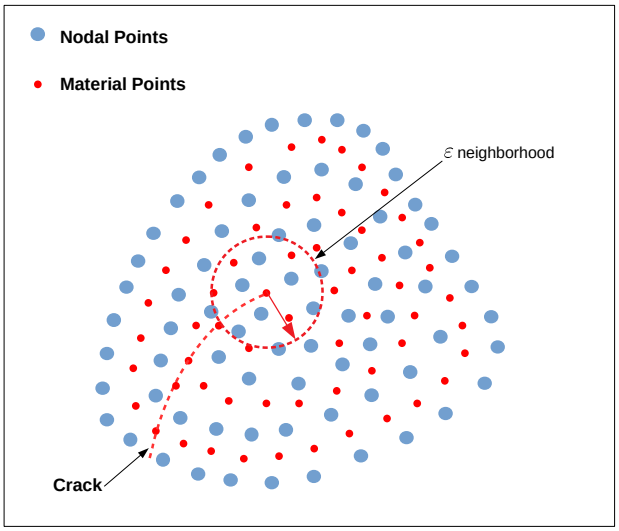


Figure 4.4. The material point erosion in OTM.

where G_c is the critical energy release rate, ΔA is the increment of new cracked area, G_0 is a material parameter, ε is the size of neighbourhood, and V_ε is the volume of ε-neighbourhood. Thus the fracture criteria for material point p is formulated as, see LI ET AL. (2015),

$$\psi_p \cdot v_p \geq G_c \Delta A, \tag{4.55}$$

where ψ_p is the specific free energy per unit volume at material point p and v_p is the volume of the material point p.
Once the fracture condition is triggered, the material point will be ruptured, where the support domain of this material point is dismissed and its contribution to the nodal residual is set to be zero. The relationship between nodal points and material points across the crack will be automatically dismissed during the update of the support domain by searching. By this approach, the crack propagation and the complex patterns of ductile fracture in 3D can be captured, see LI ET AL. (2015) for instance.

Chapter 5

Johnson-Cook Model in Finite Plasticity

In the serrated chip formation process, the material will undergo large plastic deformation. At the same time, the ductile fracture exists in the cutting zone and drives the separation of the chip from the workpiece. To capture the physical phenomena in the chip formation process, the proper constitutive model and the fracture model has to be applied. In this chapter, the finite plasticity with the Johnson-Cook flow stress model, see DE SOUZA NETO ET AL. (2011) and JOHNSON & COOK (1983), is formulated to capture the finite plastic deformation in the cutting zone. The temperature evolution is computed by assuming the thermal effect as the adiabatic process. Furthermore, the chip separation from the workpiece as well as the serrated chip morphology on the chip upper surface is treated as the ductile fracture with the Johnson-Cook fracture model, see JOHNSON & COOK (1985). The Johnson-Cook is validated by the notched bar tensile tests.

5.1 Finite plasticity

The finite plasticity framework is formulated based on the multiplicative decomposition of the deformation gradient into elastic and plastic portions, see (LEE & LIU, 1967),

$$\boldsymbol{F} = \boldsymbol{F}^e \boldsymbol{F}^p, \tag{5.1}$$

where $\boldsymbol{F}^e$ and $\boldsymbol{F}^p$ are the elastic and plastic deformation gradients, respectively. In this multiplicative split, it is assumed that there is an intermediate configuration that can be obtained from the current configuration by purely elastic unloading, see Fig. 5.1.

By defining the elastic and plastic velocity gradients

$$\boldsymbol{l}^e = \dot{\boldsymbol{F}}^e \boldsymbol{F}^{e^{-1}}, \quad \tilde{\boldsymbol{L}}^p = \dot{\boldsymbol{F}}^p \boldsymbol{F}^{p^{-1}}, \tag{5.2}$$

the plastic velocity gradient in the intermediate configuration can be reformulated as

$$\tilde{\boldsymbol{L}}^p = \frac{d}{dt}\left(\boldsymbol{F}^{e^{-1}}\boldsymbol{F}\right)\boldsymbol{F}^{p^{-1}} = \boldsymbol{F}^{e^{-1}}(\underbrace{\dot{\boldsymbol{F}}\boldsymbol{F}^{-1}}_{\boldsymbol{l}} - \underbrace{\dot{\boldsymbol{F}}^e\boldsymbol{F}^{e^{-1}}}_{\boldsymbol{l}^e})\boldsymbol{F}^e, \tag{5.3}$$

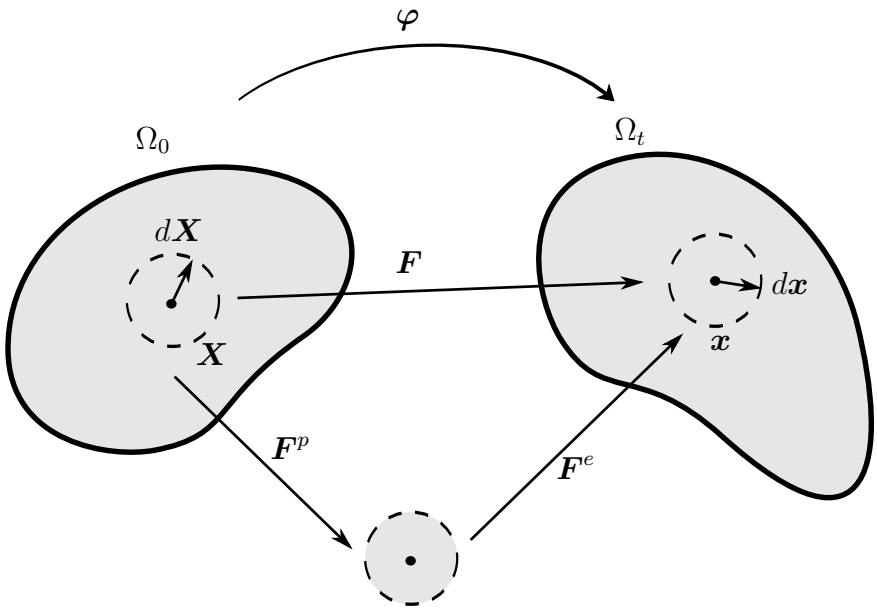


Figure 5.1. Multiplicative decomposition of the deformation gradient.

where the expression $\dot{\boldsymbol{F}}^{-1} = -\boldsymbol{F}^{-1}\dot{\boldsymbol{F}}\boldsymbol{F}^{-1}$ is applied and the variable with tilde refers to the intermediate configuration. Following the above expressions and the equation (3.13), the spatial velocity gradient can be formulated as

$$\boldsymbol{l} = \boldsymbol{l}^e + \boldsymbol{F}^e\tilde{\boldsymbol{L}}^p\boldsymbol{F}^{e^{-1}} = \boldsymbol{l}^e + \boldsymbol{l}^p, \tag{5.4}$$

where $\boldsymbol{l}^p = \boldsymbol{F}^e\tilde{\boldsymbol{L}}^p\boldsymbol{F}^{e^{-1}}$ defines the plastic velocity gradient in the current configuration. According to the equation (3.14), the symmetric part of velocity gradient can be additively decomposed as

$$\boldsymbol{d} = \boldsymbol{d}^e + \boldsymbol{d}^p, \tag{5.5}$$

where the plastic part can thus be formulated as

$$\boldsymbol{d}^p = \operatorname{sym}\left(\boldsymbol{F}^e\tilde{\boldsymbol{L}}^p\boldsymbol{F}^{e^{-1}}\right) = \operatorname{sym}\left(\boldsymbol{l}^p\right). \tag{5.6}$$

For the sake of later use, the Lie-derivative of elastic left Cauchy Green tensor is given by, see WRIGGERS (2008),

$$\mathcal{L}_v\boldsymbol{b}^e = -\boldsymbol{F}^e(\tilde{\boldsymbol{L}}^p + \tilde{\boldsymbol{L}}^{pT})\boldsymbol{F}^{eT} = -2\operatorname{sym}\left(\boldsymbol{l}^p\boldsymbol{b}^e\right). \tag{5.7}$$

By postulating a zero plastic spin $\boldsymbol{w}^p = \boldsymbol{0}$ in $\boldsymbol{l}^p = \boldsymbol{d}^p + \boldsymbol{w}^p$, the relationship between the plastic velocity gradient and the Lie-derivative of elastic left Cauchy Green tensor can be formulated as

$$\boldsymbol{d}^p = -\frac{1}{2}\mathcal{L}_v\boldsymbol{b}^e(\boldsymbol{b}^e)^{-1}. \tag{5.8}$$

To derive the constitutive equations, the form of free energy has to be first postulated. By assuming the hyperelastic law, the free energy can be expressed in terms of the elastic left Cauchy Green tensor $\boldsymbol{b}^e$ and the isotropic plastic hardening variable ξ

$$\Psi = \Psi(\boldsymbol{b}^e, \xi), \tag{5.9}$$

where$\Psi(\boldsymbol{b}^e, \xi)$ is the specific free energy in the reference configuration. The time derivative of the free energy is derived, after some straightforward manipulations, as

$$\dot{\Psi}(\boldsymbol{b}^e, \xi) = \frac{\partial \Psi}{\partial \boldsymbol{b}^e} : \dot{\boldsymbol{b}}^e + \frac{\partial \Psi}{\partial \xi}\dot{\xi} \tag{5.10}$$

$$= 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e : \boldsymbol{d} + 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e : \mathcal{L}_v \boldsymbol{b}^e (\boldsymbol{b}^e)^{-1} + \frac{\partial \Psi}{\partial \xi}\dot{\xi}, \tag{5.11}$$

where the expressions $\dot{\boldsymbol{b}}^e = \boldsymbol{l}^e \boldsymbol{F}^e \boldsymbol{F}^{eT} + (\boldsymbol{l}^e \boldsymbol{F}^e \boldsymbol{F}^{eT})^T = \boldsymbol{l}^e \cdot \boldsymbol{b}^e = \boldsymbol{b}^e \cdot (\boldsymbol{d} - \boldsymbol{d}^p)$ and the equation (5.8) have been applied.
By ignoring the kinematic hardening, the dissipation inequality is formulated as

$$D_{int} = \boldsymbol{\tau} : \boldsymbol{d} - \frac{d}{dt}\Psi(\boldsymbol{b}^e, \xi) \geqslant 0, \tag{5.12}$$

where the stress power $\boldsymbol{\tau} : \boldsymbol{d}$ is evaluated in the reference volume in equation (3.23), thus the free energy $\Psi(\boldsymbol{b}^e, \xi)$ is in the reference configuration as well. Substituting the time derivative of free energy in equation (5.11) to the dissipation inequality (5.12), it reads

$$D_{int} = \left(\boldsymbol{\tau} - 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e\right) : \boldsymbol{d} + \left(2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e\right) : \left(-\frac{1}{2}\mathcal{L}_v \boldsymbol{b}^e (\boldsymbol{b}^e)^{-1}\right) + \left(-\frac{\partial \Psi}{\partial \xi}\right)\dot{\xi} \geqslant 0. \tag{5.13}$$

Since the above equation must hold for all admissible processes, a standard argument then gives the following constitutive equations

$$\boldsymbol{\tau} = 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e, \quad R = -\frac{\partial \Psi}{\partial \xi}, \tag{5.14}$$

where R is the thermodynamic force conjugate with isotropic hardening variable ξ. The reduced dissipation inequality is thus expressed as

$$D^r_{int} = \boldsymbol{\tau} : \underbrace{\left(-\frac{1}{2}\mathcal{L}_v \boldsymbol{b}^e (\boldsymbol{b}^e)^{-1}\right)}_{\boldsymbol{d}^p} + R\dot{\xi} \geqslant 0, \tag{5.15}$$

which describes a restriction for the evolution equation of plastic flow.
To fulfill the above inequality, the dissipation potential $\Phi = \Phi(\boldsymbol{\tau}, R)$ and the associated plastic flow are postulated for the rate of plastic flow (SIMO, 1992),

$$\boldsymbol{d}^p = \dot{\gamma}\frac{\partial \Phi}{\partial \boldsymbol{\tau}}, \quad \dot{\xi} = \dot{\gamma}\frac{\partial \Phi}{\partial R}, \tag{5.16}$$

where γ is the plastic multiplier that denoting the amount of accumulated plastic strain. By applying equation (5.8), the plastic evolution equation can be reformulated in terms of elastic left Cauchy Green tensor as

$$-\frac{1}{2}\mathcal{L}_v \boldsymbol{b}^e = \dot{\gamma}\frac{\partial \Phi}{\partial \boldsymbol{\tau}}\boldsymbol{b}^e, \tag{5.17}$$

where the multiplier γ has to fulfill the Kuhn-Tucker conditions

$$\dot{\gamma} \geqslant 0,\ \Phi(\boldsymbol{\tau}, R) \leqslant 0,\ \dot{\gamma}\Phi(\boldsymbol{\tau}, R) = 0. \tag{5.18}$$

5.2 Temperature evolution in the adiabatic process

When the thermomechanical coupling is considered in the deformation process, the free energy $\Psi(\boldsymbol{b}^e, \theta, \xi)$ will depend not only on strain variables, but also on temperature θ. By applying the formulation in equation (5.8), the rate of free energy is formulated as

$$\dot{\Psi}(\boldsymbol{b}^e, \xi, \theta) = 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e : \boldsymbol{d} - 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e : \boldsymbol{d}^p + \frac{\partial \Psi}{\partial \xi}\dot{\xi} + \frac{\partial \Psi}{\partial \theta}\dot{\theta}. \tag{5.19}$$

Thus, the dissipation inequality can be evaluated as

$$D_{int} = \boldsymbol{\tau} : \boldsymbol{d} - \dot{\Psi} - \dot{\theta}\eta \tag{5.20}$$

$$= \left(\boldsymbol{\tau} - 2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e\right) : \boldsymbol{d} + \left(2\frac{\partial \Psi}{\partial \boldsymbol{b}^e}\boldsymbol{b}^e\right) : \boldsymbol{d}^p + \left(-\frac{\partial \Psi}{\partial \xi}\right)\dot{\xi} - \left(\eta + \frac{\partial \Psi}{\partial \theta}\right)\dot{\theta} \geqslant 0. \tag{5.21}$$

The constitutive equation for the specific entropy η can be derived from the above dissipation inequality as

$$\eta = -\frac{\partial \Psi}{\partial \theta}. \tag{5.22}$$

Insert the constitutive equations (5.14) to equation (5.13), the reduced dissipation can be formulated as

$$D^r_{int} = \boldsymbol{\tau} : \boldsymbol{d}^p + R\dot{\xi} \geqslant 0. \tag{5.23}$$

By applying the above reduced form, i.e. equation (5.23), to the equation (5.19), it leads to

$$\dot{\Psi} = \boldsymbol{\tau} : \boldsymbol{d} - D^r_{int} - \eta\dot{\theta}. \tag{5.24}$$

Combing the equation (5.24) with the energy balance equation $\dot{\Psi} = \boldsymbol{\tau} : \boldsymbol{d} - \eta\dot{\theta} - \theta\dot{\eta} - \mathrm{div}(\boldsymbol{q}) + r$, the evolution of entropy is obtained as

$$\theta\dot{\eta} = D^r_{int} - \mathrm{div}(\boldsymbol{q}) + r. \tag{5.25}$$

Additionally, the time derivative of entropy can be expressed according to its definition as

$$\theta\dot{\eta} = -\theta\frac{\partial^2 \Psi(\boldsymbol{b}^e, \theta, \xi)}{\partial \theta^2}\dot{\theta} - \theta\frac{\partial^2 \Psi(\boldsymbol{b}^e, \theta, \xi)}{\partial \theta \partial \boldsymbol{b}^e} : \dot{\boldsymbol{b}}^e, \tag{5.26}$$

$$= C_p\dot{\theta} - H, \tag{5.27}$$

where $C_p = -\theta \frac{\partial^2 \Psi}{\partial \theta^2}$ is the specific heat capacity, the term $\frac{\partial^2 \Psi(\boldsymbol{b}^e, \theta, \xi)}{\partial \theta \partial \boldsymbol{b}^e}$ is known as the Gough-Joule effect denoting the entropy change due to elastic deformation and H is the latent heat that can be neglected. Combing equation (5.25) with equation (5.27), the temperature evolution equation can be formulated as

$$C_p \dot{\theta} = D^r_{int} - \operatorname{div}(\boldsymbol{q}) + r, \tag{5.28}$$

$$= \boldsymbol{\tau} : \boldsymbol{d}^p + R\dot{\xi} - \operatorname{div}(\boldsymbol{q}) + r. \tag{5.29}$$

In continuum mechanics, if the heat flux across the surface in a body and the heat source within this body vanish, then a thermodynamic process is said to be adiabatic. Thus, the temperature evolution equation for the adiabatic process can be approximated as

$$\dot{\theta} = \frac{\boldsymbol{\tau} : \boldsymbol{d}^p + R\dot{\xi}}{C_p}. \tag{5.30}$$

Since the strain rate is very high in the cutting zones, the heat conduction can be neglected and thus the adiabatic process can be assumed in metal cutting simulation. In this work, the adiabatic heating induced by the plastic deformation is considered as the cause for temperature increase. Hence, the temperature evolution is estimated by the plastic stress power

$$\dot{\theta} = \beta \frac{\sigma_{eq} \dot{\gamma}}{C_p}, \tag{5.31}$$

where σ_{eq} is the equivalent stress, and β is the Taylor-Quinney coefficient that characterizes the amount of plastic work transformed into heat, see TAYLOR & QUINNEY (1934).

5.3 Algorithmic treatment

To implement the plasticity model, the evolution equations in the continuous setting have to be treated in a discrete way. By applying the exponential mapping integrator to the equation (5.17), the following discrete evolution law of $\boldsymbol{b}^e$ can be formulated

$$\boldsymbol{b}^{e,n+1} = \exp(-2\Delta\gamma \frac{\partial \Phi}{\partial \boldsymbol{\tau}}) \boldsymbol{b}^{e,n}. \tag{5.32}$$

To integrate the evolution equations, the elastic predictor-plastic corrector return mapping algorithm, see DE SOUZA NETO ET AL. (2011), is usually applied. In this case, the deformation is treated as an elastic one in the elastic trial step and then the kinematic variables are corrected according to the plastic flow in the return mapping step.
In the elastic trial step, the trial value of the elastic left Cauchy Green tensor is computed by

$$\boldsymbol{b}^{etr,n+1} = \Delta \boldsymbol{F}^{n+1} \boldsymbol{b}^{e,n} (\Delta \boldsymbol{F}^{n+1})^T. \tag{5.33}$$

The spectral representation of the left Cauchy Green tensor reads

$$\boldsymbol{b}^{etr} = \boldsymbol{Q} \boldsymbol{\Lambda}^{etr} \boldsymbol{Q}^T \tag{5.34}$$

with $\boldsymbol{Q}$ the rotation matrix composed of the eigenvectors and $\boldsymbol{\Lambda}^{etr}$ the diagonal matrix composed of the eigenvalues (stretches). The logarithmic strain, also known as Henky strain, can be computed from the stretches

$$\boldsymbol{\varepsilon}_H^{etr} = \frac{1}{2}\log(\boldsymbol{\Lambda}^{etr}). \tag{5.35}$$

In the return mapping step, the real variable can be obtained by correcting the elastic variables. The real elastic left Cauchy Green tensor can be computed according to the evolution equation (5.32) as

$$\boldsymbol{b}^{e,n+1} = \exp(-2\Delta\gamma\frac{\partial\Phi}{\partial\boldsymbol{\tau}})\boldsymbol{b}^{etr,n+1}. \tag{5.36}$$

Applying the logarithmic strain to the above equation, the update of the logarithmic elastic strain can be accordingly formulated as

$$\boldsymbol{\varepsilon}_H^{e,n+1} = \boldsymbol{\varepsilon}_H^{etr,n+1} - \Delta\gamma\frac{\partial\Phi}{\partial\boldsymbol{\tau}}. \tag{5.37}$$

It can be observed that the update of elastic strain is similar to the small strain case. In this way, the evolution equation in terms of $\boldsymbol{b}^e$ is transformed into the formulation in terms of the plastic logarithmic strain

$$\dot{\boldsymbol{\varepsilon}}_H^p = \dot{\gamma}\frac{\partial\Phi}{\partial\boldsymbol{\tau}}. \tag{5.38}$$

5.4 Johnson-Cook flow stress model

To explicitly express the constitutive equations and the evolution equations, the free energy function and the yield potential have to be defined. By assuming the linear elastic law, the hyperelastic free energy $\Psi(\boldsymbol{b}^e)$ is additively decoupled into the volumetric part $\Psi^{vol}(J^e)$ and the isochoric part $\Psi^{dev}(\boldsymbol{b}_{iso}^e)$

$$\Psi^{vol}(J^e) = \frac{1}{2}K[\log(J^e)]^2, \quad \Psi^{dev}(\boldsymbol{b}_{iso}^e) = G[\frac{1}{2}\log(\boldsymbol{b}_{iso}^e)] : [\frac{1}{2}\log(\boldsymbol{b}_{iso}^e)], \tag{5.39}$$

with $J^e = \|\boldsymbol{F}^e\|$ and $\boldsymbol{b}_{iso}^e = J^{-\frac{2}{3}}\boldsymbol{b}^e$. K and G are the bulk and shear modulus, respectively. The Kirchhoff stress can be computed in terms of the logarithmic strain as

$$\boldsymbol{\tau} = 2\frac{\partial\Psi}{\partial\boldsymbol{b}^e}\boldsymbol{b}^e = K\mathrm{tr}(\boldsymbol{\varepsilon}_H^e)\boldsymbol{I} + 2G\left[\boldsymbol{\varepsilon}_H^e - \frac{1}{3}\mathrm{tr}(\boldsymbol{\varepsilon}_H^e)\boldsymbol{I}\right]. \tag{5.40}$$

By assuming the von Mises yield criteria, the yield potential is expressed in terms of the Kirchhoff stress as

$$\Phi(\boldsymbol{\tau}, R) = \sqrt{\frac{3}{2}}\|\mathrm{dev}(\boldsymbol{\tau})\| - \sigma_Y, \tag{5.41}$$

where σ_Y is the flow stress.
To consider the strain hardening, the strain rate hardening and the thermal softening effects, the multiplicative decomposed power form of the flow stress is usually applied for metals. A commonly used model is the Johnson-Cook hardening law, see JOHNSON & COOK (1983), which is applied in this work,

$$\sigma_Y = [A + B(\varepsilon^p_{eq})^n][1 + C\ln(\frac{\dot{\varepsilon}^p_{eq}}{\dot{\varepsilon}^p_{e0}})][1 - (\frac{\theta - \theta_r}{\theta_m - \theta_r})^m], \tag{5.42}$$

in which A is the initial yield stress, $\dot{\varepsilon}^p_{e0}$ is the reference plastic strain rate, θ_r is the room temperature, θ_m is the melting temperature, B, C, n and m are additional material parameters. The temperature related term in this model is responsible for the thermal softening, which drives the adiabatic shear band formation during the serrated chip formation process. Finally, the evolution equations for the plastic strain and for the isotropic hardening strain are given by

$$\dot{\boldsymbol{\varepsilon}}^p_H = \dot{\gamma}\frac{\partial \Phi}{\partial \boldsymbol{\tau}} = \dot{\gamma}\sqrt{\frac{3}{2}}\frac{\mathrm{dev}(\boldsymbol{\tau})}{\|\mathrm{dev}(\boldsymbol{\tau})\|}, \quad \dot{\xi} = \dot{\gamma}\frac{\partial \Phi}{\partial R} = \dot{\gamma}, \tag{5.43}$$

where γ is the plastic multiplier that characterizes the magnitude of equivalent plastic strain increment $\Delta\varepsilon^p_{eq}$ in the flow rule. The equivalent plastic strain is defined as

$$\varepsilon^p_{eq} = \sqrt{\frac{2}{3}}\|\boldsymbol{\varepsilon}^p\|. \tag{5.44}$$

The Cauchy stress $\boldsymbol{\sigma}$ is obtained by back transforming the Kirchhoff stress using the same rotation tensor $\boldsymbol{Q}$ as in equation (5.34)

$$\boldsymbol{\sigma} = \boldsymbol{Q}J\boldsymbol{\tau}\boldsymbol{Q}^T. \tag{5.45}$$

To solve the evolution equations, the local Euler backward time integration scheme is used. Based on the elastic predictor-plastic corrector return mapping algorithm, the corresponding values for the next time step $n + 1$ can be calculated. For the details of the algorithm, see Box 1.

5.5 Johnson-Cook fracture model

In the serrated chip formation process, the ductile fracture will occur at different locations under complex loading conditions. The material separation from the workpiece is definitely promoted by the ductile fracture in the vicinity of the tooltip, where the material is highly stressed and the strain is highly concentrated. At the same time, the ductile fracture will occur at the chip upper surface as well, which will lead to the serrated chip or totally segmented chip. Hence the ductile fracture model should be able to predict the fracture locations with different stress states. In the Johnson-Cook fracture model, the critical equivalent plastic strain is expressed in terms of the stress triaxiality, the strain rate and the temperature. The ductile fracture will be triggered once the accumulated equivalent plastic strain ε^p_{eq} reaches the critical value ε^p_{eqf}, see JOHNSON & COOK (1985),

Box 1. Return mapping algorithm for the finite plasticity

Given: $\Delta \boldsymbol{F}$, $\boldsymbol{b}_n^e$, ξ_n and θ_n.

Find: $\boldsymbol{\sigma}_{n+1}$, $\boldsymbol{b}_{n+1}^e$, ξ_{n+1} and θ_{n+1}.

Step 1: Trial elastic

$$\boldsymbol{b}_{n+1}^{etr} := \Delta \boldsymbol{F} \boldsymbol{b}_n^e \Delta \boldsymbol{F}^T = \sum_{i=1}^{3} \lambda_i^2 \boldsymbol{n}_i^{tr} \otimes \boldsymbol{n}_i^{tr}$$

$$\hat{\boldsymbol{\varepsilon}}_{n+1}^{etr} := \sum_{i=1}^{3} \ln \lambda_i \boldsymbol{n}_i^{tr} \otimes \boldsymbol{n}_i^{tr} = \boldsymbol{Q} \boldsymbol{\varepsilon}_{n+1}^{etr} \boldsymbol{Q}^T$$

$$p^{tr} := K(\mathrm{tr}\boldsymbol{\varepsilon}_H^{etr}),\ \boldsymbol{s}^{tr} = 2G(\boldsymbol{\varepsilon}_H^{etr} - \frac{1}{3}\mathrm{tr}\boldsymbol{\varepsilon}_H^{etr} \cdot 1)$$

$$q^{tr} := \sqrt{\frac{3}{2}} \|\boldsymbol{s}^{tr}\|,\ \sigma_Y = \sigma_Y(\xi_n, \theta_n)$$

Step 2: Yield condition and return mapping

if $q^{tr} \leqslant \sigma_Y(\xi_n, \theta_n)$ **then**

Elastic,

$$p_{n+1} = p^{tr},\ \boldsymbol{s}_{n+1} = \boldsymbol{s}^{tr}$$

else

Plastic, Newton loop to calculate $\Delta\gamma$.

$$p_{n+1} = p^{tr},\ \boldsymbol{s}_{n+1} = \frac{\sigma_Y(\xi_{n+1})}{q^{tr}} \boldsymbol{s}^{tr}$$

end if

Step 3: Update stress and elastic strain:

$$\boldsymbol{\tau}_{n+1} = \boldsymbol{s}_{n+1} + p_{n+1},\ \boldsymbol{\sigma}_{n+1} = J^{-1} \boldsymbol{Q} \boldsymbol{\tau}_{n+1} \boldsymbol{Q}^T,$$
$$\theta_{n+1} = \theta_n + \beta \frac{\Delta\gamma q^{tr}}{\rho C_p},\quad \boldsymbol{\varepsilon}_{n+1}^e = \frac{1}{2\mu} \boldsymbol{s}_{n+1} + \frac{1}{3} \frac{p^{tr}}{K} \mathbf{1},$$
$$\boldsymbol{b}_{n+1}^e = \boldsymbol{Q} \exp(\boldsymbol{\varepsilon}_{n+1}^e) \boldsymbol{Q}^T.$$

$$\varepsilon_{eq}^p \geqslant \varepsilon_{eqf}^p = [d_1 + d_2 \exp(d_3 \eta_s)][1 + d_4 \ln(\frac{\dot{\varepsilon}_{eq}^p}{\dot{\varepsilon}_{e0}^p})][1 + d_5 \frac{\theta - \theta_r}{\theta_m - \theta_r}], \tag{5.46}$$

where d_1, d_2, d_3, d_4 and d_5 are the material parameters, η_s is the stress triaxiality defined as

the ratio between the hydrostatic pressure p and the von Mises stress σ_v

$$\eta_s = \frac{p}{\sigma_v}, \quad p = \frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma}). \tag{5.47}$$

Instead of artificially setting a separation line in the workpiece, both the chip separation from the workpiece and the serrated morphology on the chip upper surface are treated as ductile fracture, which is described by the Johnson-Cook fracture model in this work.

5.6 Validation of the Johnson-Cook material model

In this part, the Johnson-Cook flow stress model and the fracture model are validated by comparing the simulation with the experimental results in the literature. The tension tests on the notched bar are usually conducted to calibrate the Johnson-Cook model and to predict the ductile fracture under tension loading conditions, see CORONA & ORIENT (2014). The geometry of the notched bar is shown in Fig. 5.2. Different stress states will be recovered in the notched zone with different notch radii R. Three values of 0.32, 0.128 and 0.064 are assigned to the notch radius R in this work, which leads to three different ratios of the notch radius to the radius of the bar.

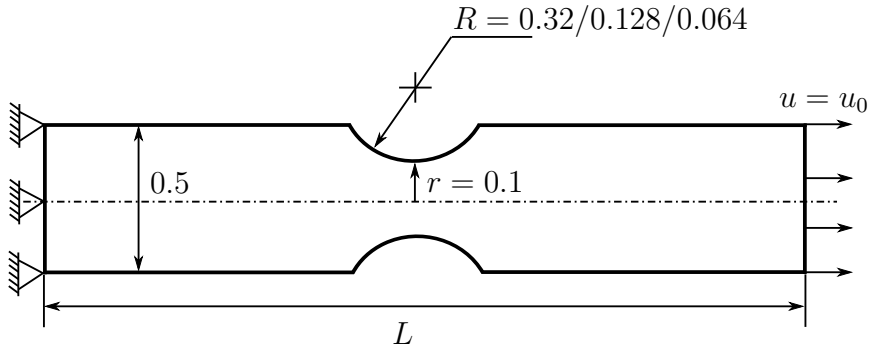


Figure 5.2. The geometry of the notched bar.

The notched bar is fixed at the left end and a displacement boundary is imposed on the right end. The load deflection curves of the notched bar tension tests are computed by the use of the stabilized OTM method and are shown in Fig. 5.3. It can be observed that the load deflection curves, as well as the fracture locations, predicted using the Johnson-Cook model have a good agreement with the experimental results.

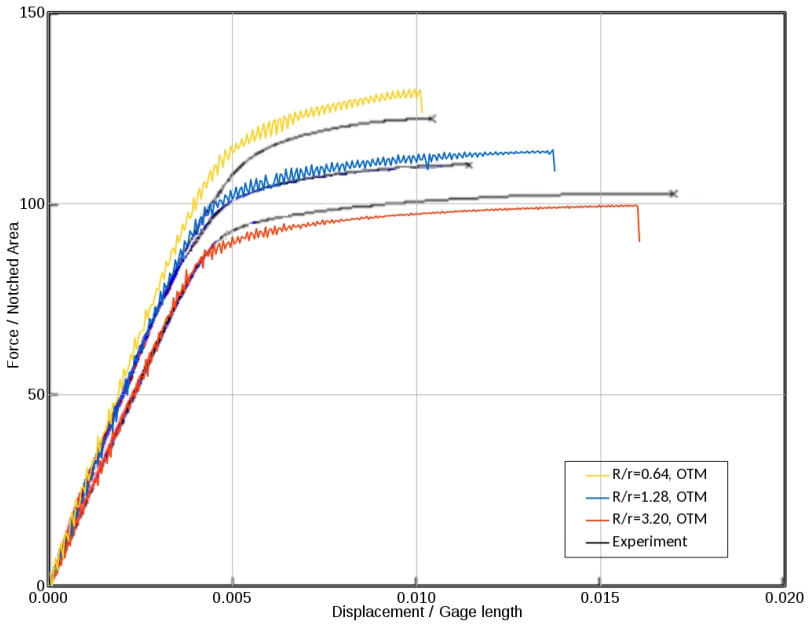


Figure 5.3. Load deflection curve of the notched bar (experimental results taken from (CORONA & ORIENT, 2014)).

Chapter 6

Metal Cutting Simulation Using Johnson-Cook Material Model

In this chapter, the 3D stabilized OTM framework with plane strain assumption is applied to the modelling of the serrated chip formation process in orthogonal cutting of a rectangular block. The plastic deformation and ductile fracture of the workpiece are described by the Johnson-Cook flow stress model and the Johnson-Cook fracture model, respectively. A supplementary condition for the stress triaxiality is proposed to improve the fracture model for serrated chip generation. The frictional contact force exerted by the cutting tool is imposed on the workpiece using a predictor-corrector strategy. The effectiveness of the numerical model is demonstrated by comparing the predicted chip morphology, cutting force and chip formation process with experimental results[1].

In this work, it is shown that the Johnson-Cook flow stress model leads to an under prediction of the thermal effect for shear band formation. Hence, a larger magnitude of the Taylor-Quinney coefficient in the temperature evolution equation has to be applied to reproduce thermal softening effects more realistically. However, this larger magnitude of the Taylor-Quinney coefficient is physically incorrect. Instead of the pure plastic deformation approach and the separation layer approach in the literature, both the chip separation from the workpiece and the serrated morphology on the chip upper surface are treated as the ductile fracture. This enables a more realistic modelling of the chip formation. As a well-validated ductile fracture model, such as in CORONA & ORIENT (2014) and ROTH & MOHR (2014), the Johnson-Cook fracture model (JOHNSON & COOK, 1985) shows limitations in capturing the fracture on the chip upper surface. Thus, a supplementary condition of positive stress triaxiality is introduced in this work to improve the performance of the Johnson-Cook fracture model. This condition allows more accurate representation and measurements of the chip size, i.e. chip spacing, peak and valley. By combining the stabilized OTM method with the material point erosion approach, both the chip separation from the workpiece and the fracture at the chip upper surface are successfully captured.

[1] Parts of this chapter are published in HUANG ET AL. (2019).

6.1 Parameter setting

6.1.1 The geometry of cutting tool and workpiece

During the metal cutting process, the deformations of the workpiece and the chip are directly driven by the cutting tool which moves in the horizontal direction with a specific cutting speed and cutting depth as shown in Fig. 6.1. The workpiece is a rectangular block with the length and height of $300\mu m$ and $120\mu m$, respectively. The cutting depth is set as $100\mu m$.

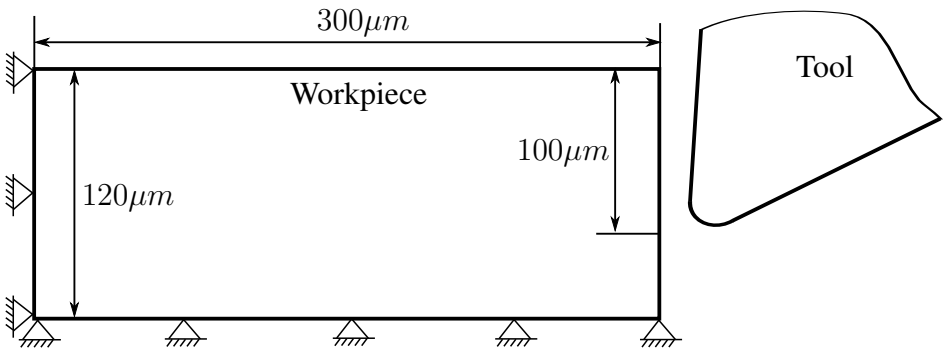


Figure 6.1. The geometry of the workpiece.

The friction behaviour between the cutting tool and the workpiece has a large effect on the cutting process. In this work, the cutting tool is assumed to be a rigid body. The boundary of the cutting tool is described by analytical functions. As shown in Fig. 6.2, the geometry of the cutting tool is defined based on the coordinate system with the original point O at the center of the round tooltip. The cutting tool consists of three parts: the rake face, the round tooltip and the flank face. The angle of the rake face with respect to the vertical axis is defined as the rake angle α_r. The angle of the flank face with respect to the horizontal axis corresponds to the flank angle α_f. To globally distinguish which part of the cutting tool will come into contact, the surface of the cutting tool is divided into three parts according to the polar angle

- Tooltip: $\pi - \alpha_r \leqslant \alpha < 1.5\pi + \alpha_f$,
- Rake face: $0.25\pi + 0.5\alpha_f - 0.5\alpha_r \leqslant \alpha < \pi - \alpha_r$,
- Flank face: the rest.

For each part of the cutting tool, the geometrical boundary can be analytically expressed in the $x - y$ plane as

- The tooltip

$$x = x_c + R\cos\alpha, \tag{6.1}$$

$$y = y_c + R\sin\alpha, \tag{6.2}$$

- The rake face

$$y - y_r = \tan(0.5\pi - \alpha_r)(x - x_r), \tag{6.3}$$

- The flank face

$$y - y_f = \tan\alpha_f(x - x_f), \tag{6.4}$$

where (x_c, y_c) is the center of round tool tip, (x_r, y_r) is the intersection between the tool tip and the rake face, (x_f, y_f) is the intersection between the tool tip and the flank face. The rake angle of cutting tool is set as 0° and tool radius as $2\mu m$. The friction coefficient is set to be 0.8 in the tool-chip contact modelling.

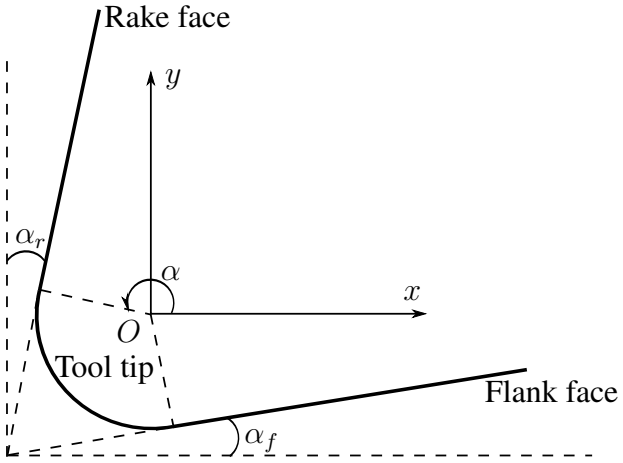


Figure 6.2. Geometry of cutting tool.

6.1.2 Material parameters

The Ti-6Al-4V alloy is employed as the workpiece material. The material properties of the constitutive models in equation (5.42) and equation (5.46) are listed in Table 6.1. The initial temperature θ_r and the melting temperature θ_m for the workpiece are set to be 25°C and 1630°C, respectively.

6.2 The effect of stress triaxiality

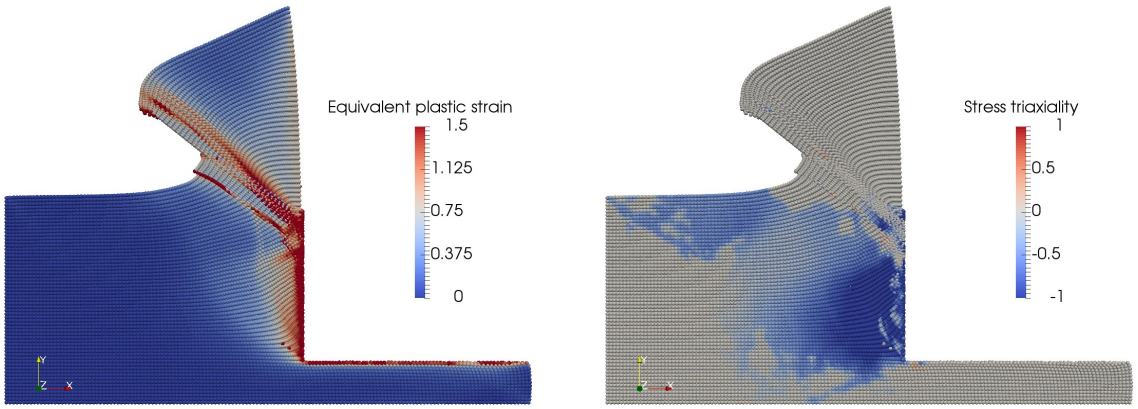
In the chip formation process, very complex stress states occur in the zones of large deformation. The stress triaxiality is employed in the ductile fracture model to distinguish tension and compression stress states. As shown in Fig. 6.3(a), the first segment of the serrated chip is generated as a result of localized plastic deformations in the shear band and ductile fracture in the chip if the original Johnson-Cook fracture model of equation (5.46) is applied. The stress triaxiality is plotted in Fig. 6.3(b), where the positive stress

Table 6.1. Material parameters of Ti6Al4V (LEE & LIN, 1998) (MIGUÉLEZ ET AL., 2013) (ZHANG ET AL., 2011).

Parameter	Symbol	Value	Unit
Density	ρ	4420	Kg/m^3
Elastic modulus	E	114	GPa
Poisson's ratio	ν	0.33	$-$
Specific heat capacity	C_p	560	$J/kg°C$
Room temperature	θ_r	25	$°C$
Melting temperature	θ_m	1630	$°C$
Initial yield stress for JC flow stress model	A	782	MPa
Hardening modulus for JC flow stress model	B	498	MPa
Work-hardening exponent for JC flow stress model	n	0.28	$-$
Strain rate dependency coefficient for JC flow stress model	C	0.028	$-$
Thermal softening exponent for JC flow stress model	m	1	$-$
JC fracture model parameter	d_1	0.4	$-$
JC fracture model parameter	d_2	0.25	$-$
JC fracture model parameter	d_3	-0.5	$-$
JC fracture model parameter	d_4	0.014	$-$
JC fracture model parameter	d_5	3.87	$-$

triaxiality indicates the tension stress state whereas the negative stress triaxiality indicates the compression stress state. It can be seen that the plastic strain is very large within the shear band and the stress triaxiality within the shear band is close to zero. The chip separation from the workpiece and the serrated morphology on the chip upper surface are successfully captured by the numerical model. However, the chip segment in Fig. 6.3(a) is totally disconnected from the workpiece with more than one ductile crack generated within the shear band, which does not agree with the continuous shear band observed in the experiment as shown in Fig. 6.4. This shows the limitation of the Johnson-Cook fracture model in prediction of ductile fracture at a low level of stress triaxiality.

From the micromechanical point of view, the ductile crack at the macro scale (ductile fracture) is initiated when the nucleation, the growth and the coalescence of microcavities (ductile damage) evolve to a certain extent, see GROSS & SEELIG (2017). In addition, the nucleation and growth rate of a microcavity is an increasing function of stress triaxiality, see LEMAITRE (2012). In other words, the ductile fracture will be prevented at the location with negative stress triaxiality. In the Johnson-Cook fracture model of equation (5.46), the stress triaxiality is employed in exponential form. However, when the stress triaxiality is negative or close to zero, the critical equivalent plastic strain calculated by equation (5.46) can also be reached due to extremely large plastic deformations in the shear band, which will lead to the over failure in the zone with large plastic deformation but under high compression, such as the fracture in Fig. 6.3. To restrict the ductile fracture only in the area with zero or positive stress triaxiality, in this work, the supplementary condition is applied to improve the



(a) Equivalent plastic strain distribution

(b) Stress triaxiality distribution

Figure 6.3. The chip morphology without restriction of η_s ($v_c = 10m/s$, $\beta = 2$).

performance of the Johnson-Cook fracture model in equation (5.46), i.e.

$$\eta_s \geqslant 0. \tag{6.5}$$

With this assumption, both the chip separation from the workpiece at the chip root and the fracture on the chip upper surface as shown in Fig. 6.4 are well captured by the fracture model as shown in Fig. 6.5(a). At the same time, the highly compressed chip material within the shear band is still retained even though a large plastic deformation exists, see Fig. 6.5(a). The continuous shear band in Fig. 6.5(a) has a good agreement with the experimental observation in Fig. 6.4. The potential ductile fracture locations, both at the root and on the upper surface of the chip, are clearly captured by the positive stress triaxiality in Fig. 6.5(b). Hence, the serrated morphology on the chip upper surface clearly shows the peaks and valleys, which are observed in the experimental results of the chip formation process.

6.3 The effect of Taylor-Quinney coefficient

The adiabatic shear band formation is driven by thermal softening, which is described by the temperature related thermal softening term in the Johnson-Cook flow stress model of equation (5.42). The higher the temperature, the stronger the thermal softening effect. The Taylor-Quinney coefficient β in the temperature evolution equation (5.31) indicates the amount of plastic work transformed into heat. Therefore, the magnitude of β is directly related to the thermal softening effect and has to be in the range between 0 and 1. As shown in Fig. 6.6(a), a continuous chip is generated even when the maximum value of 1 is assigned to β, which means that the whole energy is transformed into the thermal energy. This is physically incorrect and shows that the thermal softening is under-predicted by the Johnson-Cook flow stress model. A smaller value of β will lead to a lower temperature in the cutting zone, which restrains the thermal softening and the strain localisation in front of the tooltip. If a larger

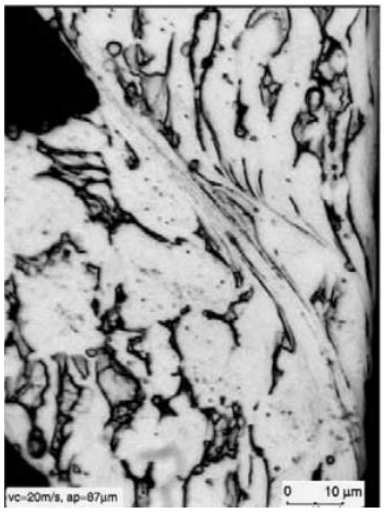
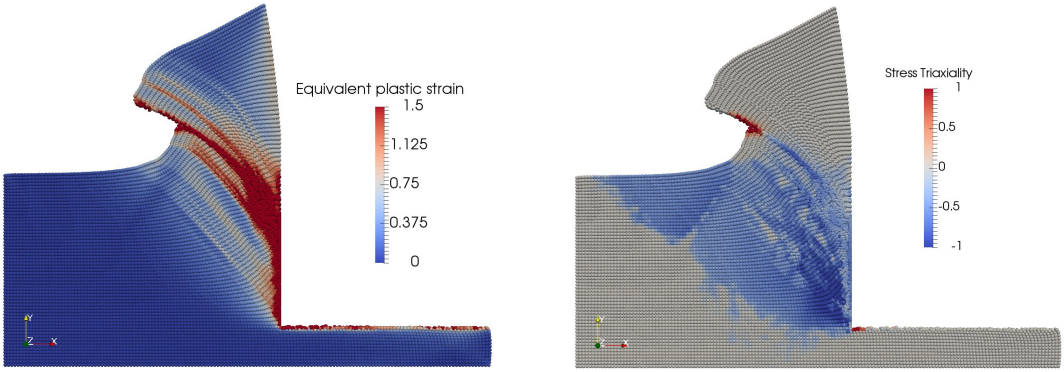


Figure 6.4. Shear band in experiment (BÄKER ET AL., 2002)



(a) Equivalent plastic strain distribution

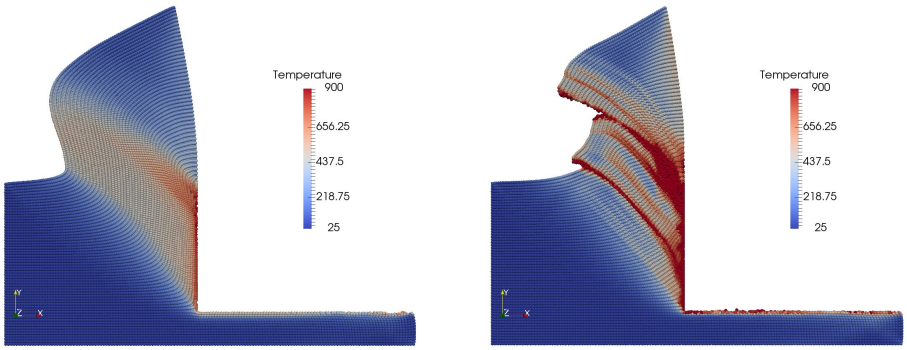
(b) Stress triaxiality distribution

Figure 6.5. The chip morphology with restriction of η_s ($v_c = 10m/s$, $\beta = 2$).

value of β is applied, the serrated chip, which is observed in experiments, is generated with a higher temperature in the cutting zone as shown in Fig. 6.6(b). This indicates that the thermal softening drives the shear band formation in metal cutting. Additionally, it shows the limitations of the Johnson-Cook flow stress model.

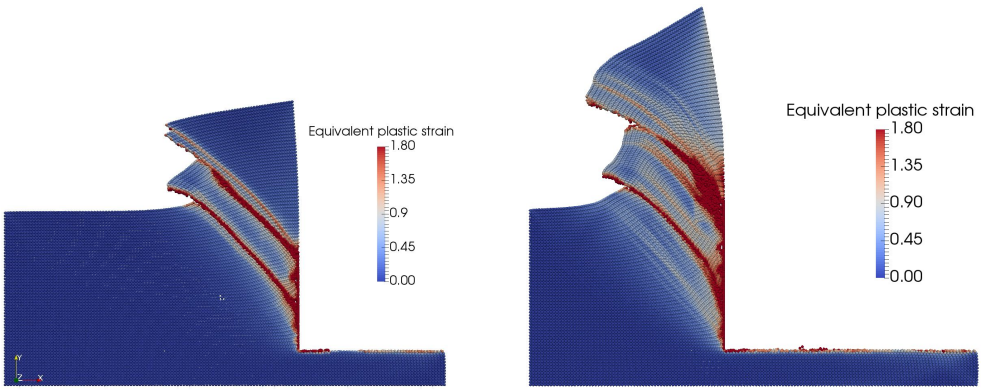
6.4 The effect of cutting speed

According to the experimental results, the cutting speed has a large effect on the chip morphology, see YE ET AL. (2013), e.g. chip spacing and chip size. As shown in Fig. 6.7, the chip spacing increases with the increase of the cutting speed, whereas the angle between the shear band and tool rake face decreases with an increase of the cutting speed, which has been observed in the experiments of reference YE ET AL. (2013).

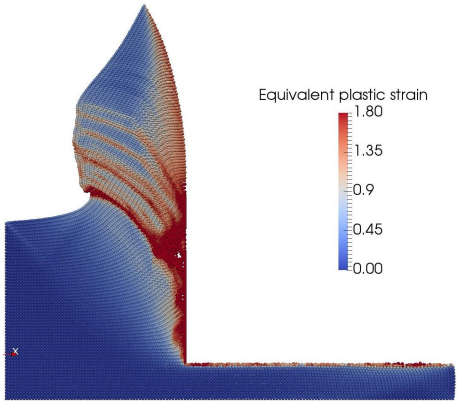


(a) $\beta = 1$ (b) $\beta = 2$

Figure 6.6. The effect of β on chip morphology ($v_c = 10m/s$).



(a) $v_c = 5m/s$ (b) $v_c = 10m/s$

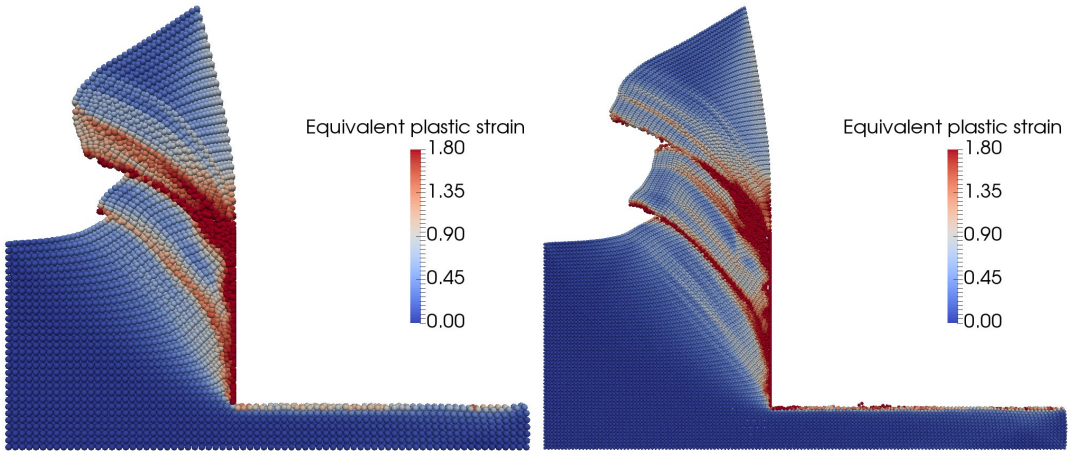


(c) $v_c = 15m/s$

Figure 6.7. The effect of cutting speed on chip morphology ($\beta = 2$).

6.5 The effect of spatial discretisation

To capture the adiabatic shear band formation, the spatial domain needs to be discretized with enough material points and nodal points. The chip formation process is simulated with a different number of material points as shown in Fig. 6.8. When fewer material points are used, the first shear band generates branches after its initiation, which leads to smaller chip spacing. With the increase of the material point, the first shear band becomes clear and narrow. Furthermore, the chip spacing converges to a constant value, see Fig. 6.9. Due to the high computational cost, the spatial domain is discretized with 107,975 material points in the previous and the following sections.



(a) with 27,000 material points (b) with 107,975 material points

Figure 6.8. The effect of spatical discretisation ($v_c = 10m/s$, $\beta = 2$).

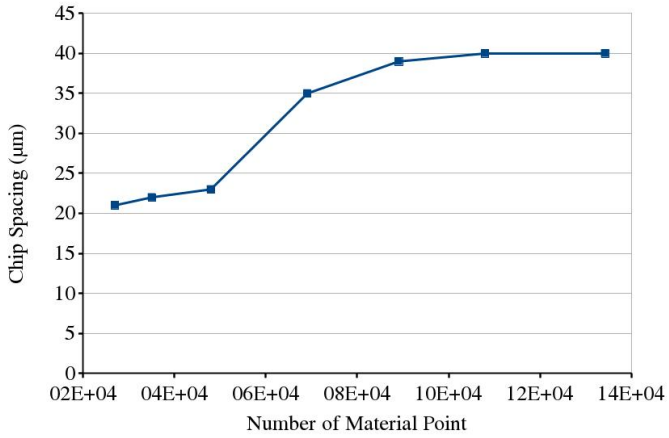


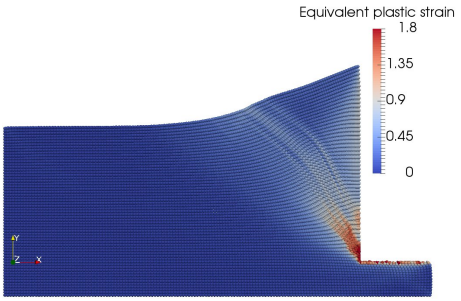
Figure 6.9. The effect of spatical discretisation ($v_c = 10m/s$, $\beta = 2$).

6.6 The serrated chip formation process

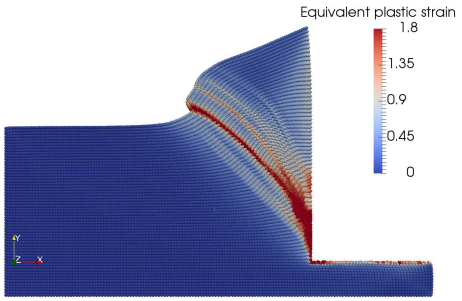
In the end, the whole modelling of the chip formation process is shown using the above model improvements (positive stress triaxiality and $\beta = 2$). The results are also compared with the experimental observations. Fig. 6.10 shows the calculated serrated chip formation process in orthogonal cutting with the cutting speed of $10m/s$, where only the side view of the 3D simulation is shown. The equivalent plastic strain is plotted at different tool displacements u_t. Fig. 6.10(a) to Fig. 6.10(c) shows the generation of the first chip segment whereas Fig. 6.10(d) to Fig. 6.10(f) shows the second one. At the beginning of each segment generation, see Fig. 6.10(a) and Fig. 6.10(d), the plastic strain concentrates at the tooltip and the shear band initiates from the root of the chip, which agrees with the Stage 1 of chip formation process observed in the experiment, see Fig. 2.5. Then the shear band will propagate quickly along the primary shear zone and the entire adiabatic shear band is formed as shown in Fig. 6.10(b) and Fig. 6.10(e). Due to the fracture on the chip upper surface and the rising up of the chip, the serrated chip is generated as shown in Fig. 6.10(c) and Fig. 6.10(f), which agrees with the Stage 2 of the chip formation process in Fig. 2.5. Additionally, the serrated chip morphology and the shear band in Fig. 6.10(f) have been observed in the experiment by SUTTER & LIST (2013) in Fig. 6.11. The chip spacing, measured at the second chip segment, in Fig. 6.10(f) is about $40\mu m$, which is very close to the experimentally measured value of about $45\mu m$ in YE ET AL. (2013) at the same cutting conditions.

Fig. 6.12 shows the stress triaxiality development in the serrated chip formation process shown in Fig. 6.10. At the beginning of each segment generation, see Fig. 6.12(a) and Fig. 6.12(c), the undeformed part of the chip segment is highly compressed and the stress triaxiality is negative. At the same time, positive stress triaxiality occurs both on the upper surface and at the root of the chip, which indicates the potential ductile fracture locations. Due to the raising up of the chip along with the shear band, the ductile fracture occurs at the locations with positive stress triaxiality as shown in Fig. 6.12(b) and Fig. 6.12(d).

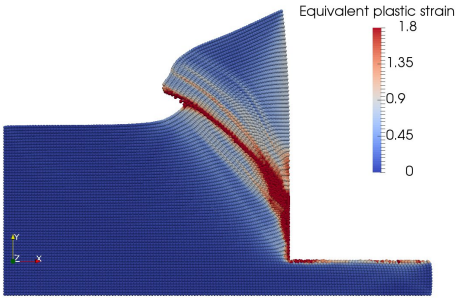
The cutting force development in this serrated chip formation process is also investigated and shown in Fig. 6.13. This force is calculated by summing up the normal contact forces between the chip segment and the tool rake face. It can be seen that the cutting force fluctuates around an average value (the dashed line in Fig. 6.13) during the cutting process, which corresponds well to the serrated chip formation process from Fig. 6.10(a) to Fig. 6.10(f). At the beginning of the first chip segment generation, the cutting force is very large due to the high compression in the entire tool-chip contact interface. Then, the cutting force drops significantly when the shear band starts to propagate. When the first chip segment raises up to some position, the generation of the second chip segment is initiated. Thus the cutting force starts to increase, which is followed by the second cycle of fluctuation. This fluctuation behaviour agrees well with the experimentally measured cutting force in HARZALLAH ET AL. (2018) as shown in Fig. 6.14. In addition, it is worth noting that the overlap between adjacent segments' generation can be observed. Hence the cutting force in each fluctuation cycle is not applied by one unique segment.



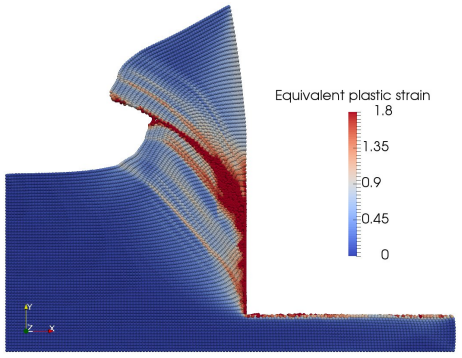
(a) $u_t = 0.05mm$



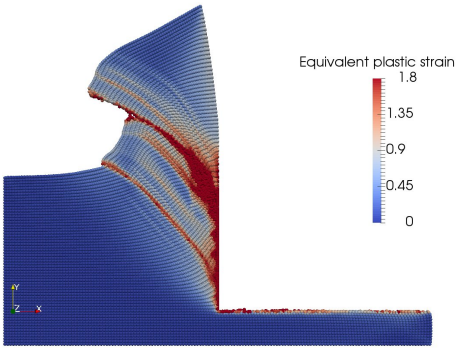
(b) $u_t = 0.08mm$



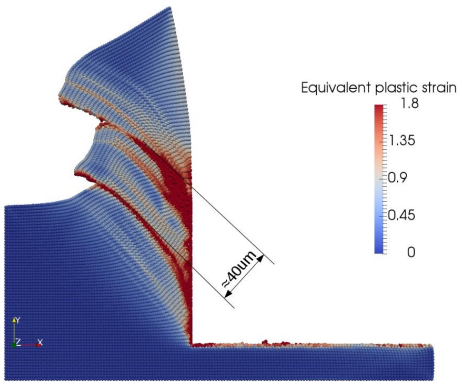
(c) $u_t = 0.10mm$



(d) $u_t = 0.14mm$



(e) $u_t = 0.15mm$



(f) $u_t = 0.17mm$

Figure 6.10. The side view of serrated chip formation process ($v_c = 10m/s$, $\beta = 2$).

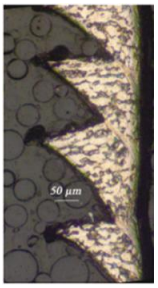
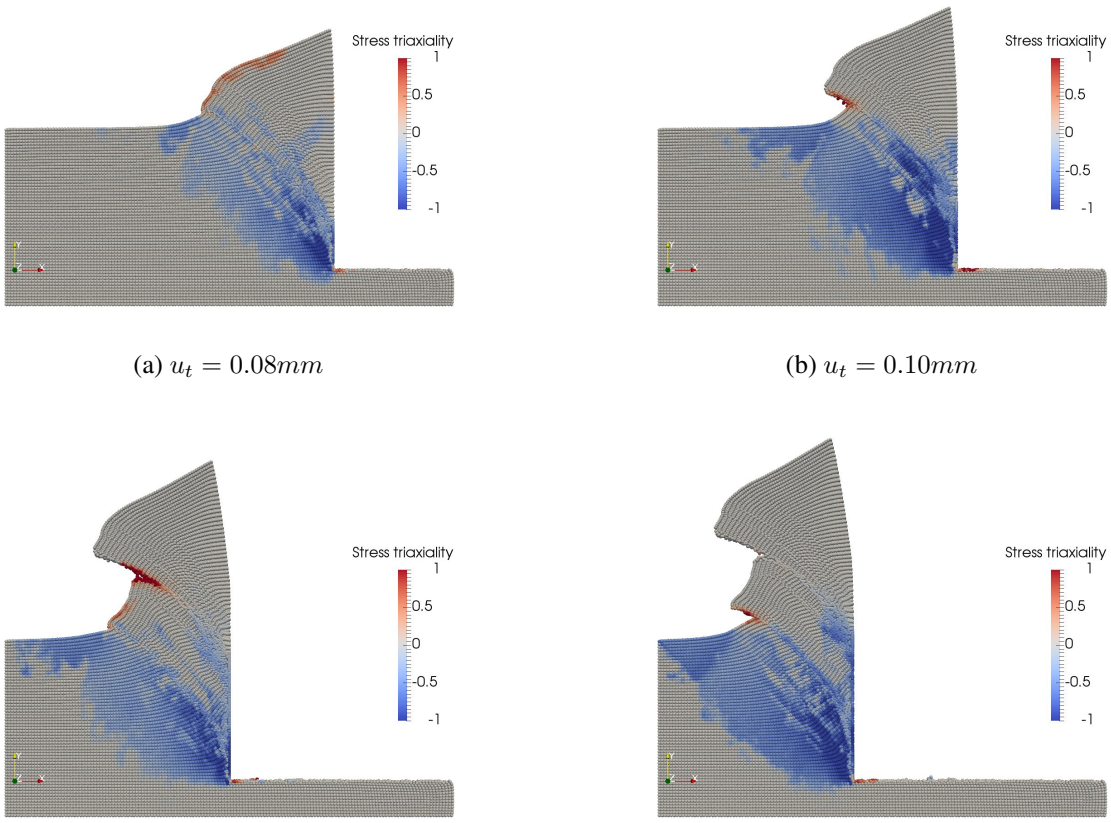


Figure 6.11. Chip morphology for Ti6Al4V by SUTTER & LIST (2013).



(a) $u_t = 0.08mm$ (b) $u_t = 0.10mm$

(c) $u_t = 0.15mm$ (d) $u_t = 0.17mm$

Figure 6.12. The stress triaxiality development in the serrated chip formation process ($v_c = 10m/s$, $\beta = 2$).

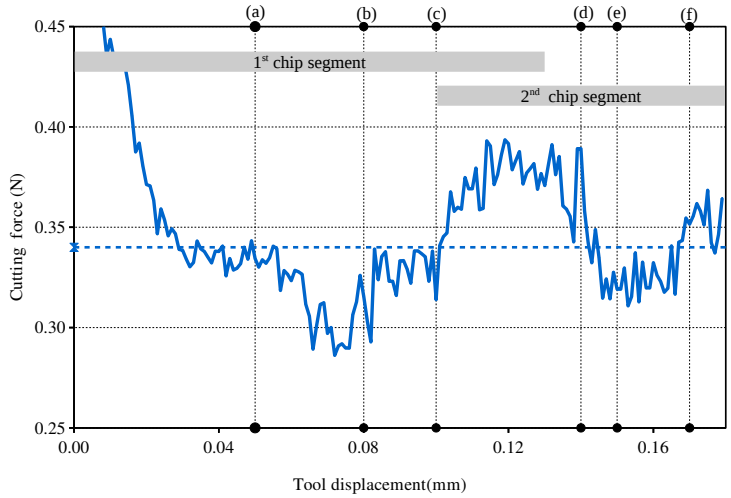


Figure 6.13. Cutting force development in the serrated chip formation process.

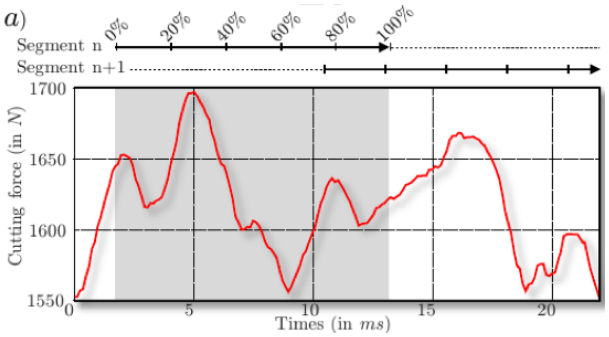


Figure 6.14. Cutting force in experiment by HARZALLAH ET AL. (2018).

Chapter 7

Machine Learning (ML) based Material Model

7.1 Introduction

The phenomenological material model such as the Johnson-Cook model shows good performance for some simple loading conditions, e.g. the tension of the notched bar (CORONA & ORIENT, 2014). However, it is not always close to the experimental results when the structure undergoes complex deformations such as the shear dominated loadings. Thus, the parameters of the material model need to be further calibrated by conducting more experiments. In some cases, the material model has to be modified to account for more physical effects, exemplarily see CAO (2017). Comparing with the classical material models, the data-driven material models show many advantages, such as the direct utilisation of experimental data and the possibility to improve accuracy when additional data are available.

To replace the classical material modelling approach, several data-driven material modelling approaches have been proposed, such as the model-free data-driven computing paradigm, see KIRCHDOERFER & ORTIZ (2016), KIRCHDOERFER & ORTIZ (2018), EGGERSMANN ET AL. (2019) and STAINIER ET AL. (2019), the manifold learning approach, see IBAÑEZ ET AL. (2017), IBAÑEZ ET AL. (2018) and IBAÑEZ ET AL. (2019), and the self-consistent clustering approach for heterogeneous materials, see LIU ET AL. (2016) and SHAKOOR ET AL. (2019). In addition to the aforementioned data-driven approaches, the artificial neural network acting as a Machine Learning (ML) approach has been applied to approximate the constitutive model based on data as well, see GHABOUSSI & SIDARTA (1998), HASHASH ET AL. (2004), and LEFIK & SCHREFLER (2003). However, learning and predicting history dependent material models such as plasticity is still challenging.

In this chapter, a ML based material modelling framework is proposed for both elasticity and plasticity. The ML based hyperelasticity model is directly developed with the Feed forward Neural Network (FNN). The ML based plasticity model is developed by using two approaches: the Recurrent Neural Network (RNN) and a novel method called Proper Orthogonal Decomposition Feed forward Neural Network (PODFNN).

Before formulating the ML based plasticity model, the data collection strategy for the plasticity model is proposed. To simplify the data collection process from experiments, only

strain components act as input and only stress components represent the output of the ML based model. Instead of using the previous strain and stress as history variables, in this work, the accumulated absolute total strain is applied as the history variable in the input to distinguish different loading paths. This variable can be computed from preexisting inputs without additional effort, e.g. from experiments. Due to its history dependence, the strain-stress sequence data for plasticity are collected from different loading-unloading paths based on the concept of sequence for plasticity. Since the isotropic plasticity can be formulated in the principle space, the training sequence data are collected only from the multi-axial tests, which simplifies the data collection process.
In the RNN based plasticity model, strain components act as input and stress components act as the output of the model, where the loading history is recorded automatically by the internal states of the network. Whereas in PODFNN based plasticity model, the multi-dimensional stress sequence is decoupled leading to independent one dimensional coefficient sequences by means of the POD. In this case, the neural network based model with multiple outputs is replaced by multiple independent neural network models in which each possesses a one-dimensional output. The input of the PODFNN based model contains strain components and their history variables. This decoupled approach leads to less training time and better training performance.
To apply the ML based material model in finite element analysis, the tangent matrix is derived by the automatic symbolic differentiation tool AceGen, see KORELC & WRIGGERS (2016). The effectiveness and generalisation of the presented models are investigated by a series of numerical examples using both 2D and 3D finite element analysis.[1]

7.2 Data-driven material modelling framework

To develop a data-driven material model by means of machine learning technology, three steps are necessary: data collection, machine learning and validation, see Fig. 7.1. As a fundamental ingredient for data-driven models, the data, representing the material behaviour, have to be collected firstly. According to the specific problem, the training data can be collected from experiments and simulations.
In this work, strain-stress data are employed as the input and output of the data-driven model. For the plasticity model, the strain-stress data will be collected for specific loading paths and stored as sequences. Depending on the problem, the training data usually have to be preprocessed utilizing data scaling, data decomposition and data arrangement.
The second step is to fit the constitutive equation related to the data by means of the ML technology. The Artificial Neural Network (ANN) as a machine learning technology will be employed in this work. The hyperparameters of the neural network based model have to be selected according to the data and the accompanying accuracy requirements. Once the model is trained, the describing parameters will be used and stored for the material description of the developed model.
The final step is to validate the accurate reproduction of the ML based material model. To this end, the ML based material model is compared with a standard material model within

[1] Parts of this chapter are published in HUANG ET AL. (2020).

several finite element applications. By deriving the tangent matrix and residual vector, the ML based model can be incorporated into a FEM code. The performance of the developed model will be evaluated by benchmark tests. If the accuracy of the material model can not meet the necessary requirements, the model hyperparameters will be optimized or supplemental data will be collected. Therefore, the machine learning based framework is an open system and the accuracy of the developed model can be improved iteratively during its application.

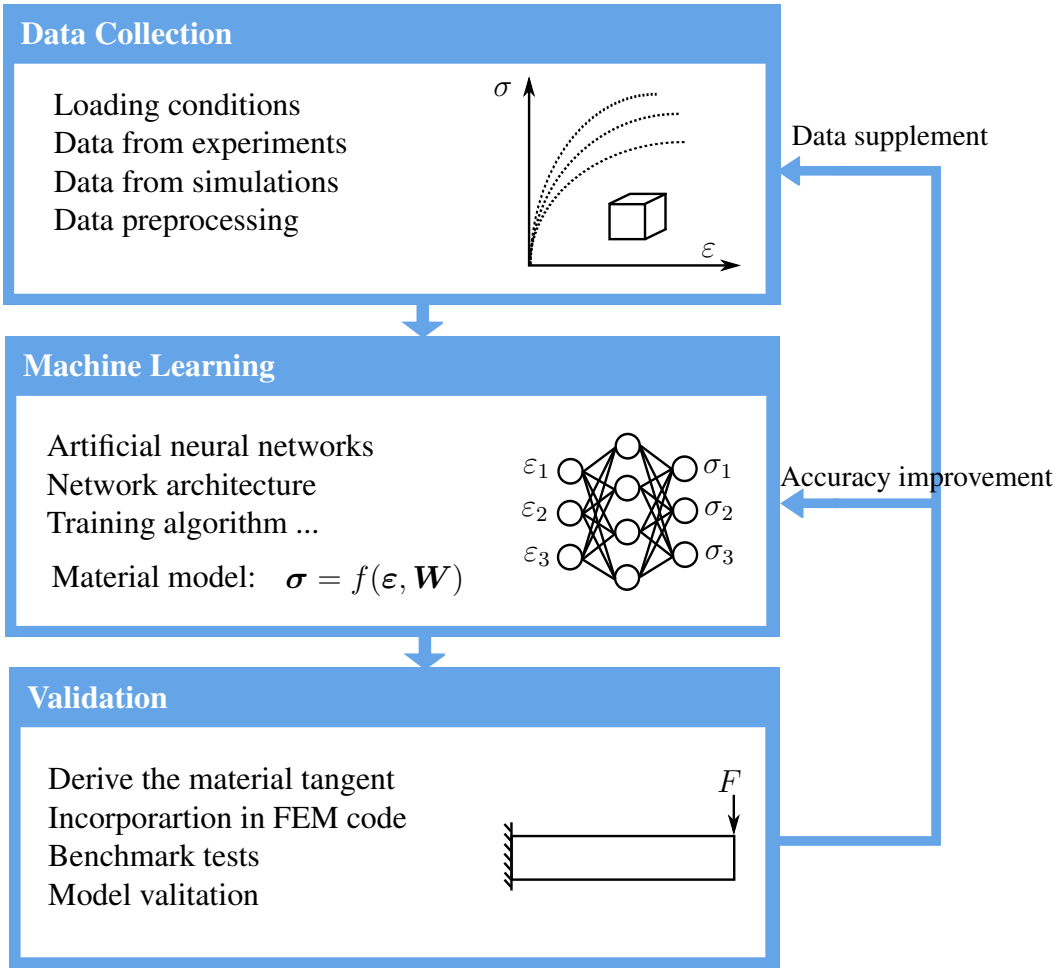


Figure 7.1. The data-driven material modelling framework.

7.3 ML based hyperelasticity

Before the plasticity model is developed in detail, a ML based hyperelasticity model is presuited by utilizing feed forward neural networks in this section.

7.3.1 Feed forward Neural Network (FNN)

Feed forward neural network is a fundamental ML technology. A deep FNN is composed of several connected layers of artificial neurons and biases, where the data are fed into an input layer and then flows through some hidden layers. The output is finally predicted at an output layer, as shown in Fig. 7.2. The neurons from different layers are fully connected through the weights w. In the prediction phase, the data flow in one way from the input layer to the output layer. In the training phase, the global error defined by the mean-squared differences between the target value and the FNN output will be back-propagated through the hidden layers. This step is performed in order to update the weights, where the objective is to minimize the global error.

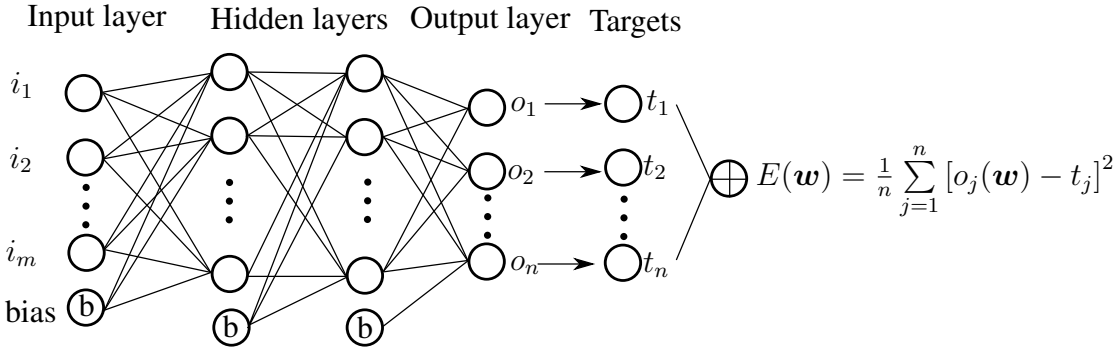


Figure 7.2. The feed forward neural network.

At each neuron, an activation function is attached, see Fig. 7.3. The output of each neuron is computed by multiplying the outputs from the previous layer with the corresponding weights. For the neuron j in the layer k, the data of the previous layer $k-1$ are summed up and then altered by an activation function. The output of the neuron j in layer k is computed as

$$o_j^k = f_s\left(\sum_{i=1}^{N} w_{ij} o_i^{k-1} + b_i^{k-1}\right), \tag{7.1}$$

where N is the number of neurons in the previous layer $k-1$, w_{ij} is the weight connecting neurons i and j, o_i^{k-1} is the output of the neuron i in layer $k-1$ whereas b_i^{k-1} is its bias. A common choice for the activation function is the sigmoid function

$$f_s(x) = \frac{2}{1+e^{-2x}} - 1. \tag{7.2}$$

The specific architecture of the FNN, such as the number of layers and the number of neurons in each layer, has to be determined according to the complexity of the data set.

7.3.2 Back propagation algorithm

In the training phase, the weighs of the neural network will be initialized firstly, see NGUYEN & WIDROW (1990), which is followed by the weights updating using a training algorithm

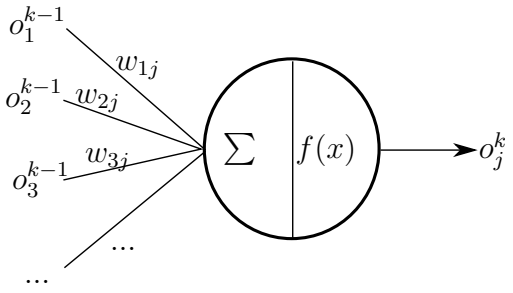


Figure 7.3. The artificial neuron.

such that the global error is minimized. The global error, also named as loss function or network performance, is defined according to the difference between the network prediction and the target data as shown in Fig. 7.2. The mean squared error is used to measure the loss

$$E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N}[o_i(\boldsymbol{w}) - t_i]^2 = \frac{1}{N}\sum_{i=1}^{N} e_i, \tag{7.3}$$

where N is the number of outputs, o_i is the i-th output, $\boldsymbol{w}$ is the vector that contains the weights of the neural network, and t_i is the i-th target value. Training a feed forward neural network is an optimisation problem, where the global error is treated as the objective function. To train the feed forward neural network, the gradient based algorithm can be applied in the optimisation process. Since the hidden layers are located parallel with the output layer, the so-called back propagation technique will be used to compute the gradients in terms of the weights. Fig. 7.4 shows the connections of 3 neurons in the output and hidden layers in FNN, where the partial derivative of global loss E with respect to weight w_{ij}^n will be computed by back propagating the error from the output layer to the hidden layers.

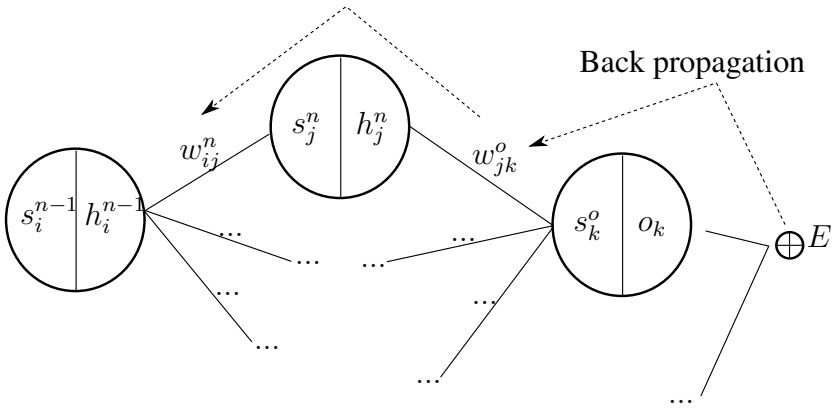


Figure 7.4. The back propagation algorithm.

Gradient with respect to weights of output layer

As depicted in Fig. 7.4, the gradient of global loss E with respect to weight w^o_{jk} of output layer can be expressed by the chain rule

$$\frac{\partial E}{\partial w^o_{jk}} = \frac{\partial E}{\partial o^k} \cdot \frac{\partial o^k}{\partial s^o_k} \cdot \frac{\partial s^o_k}{\partial w^o_{jk}}, \tag{7.4}$$

where $o_k = f_s(s^o_k)$ is the output of the neuron k in output layer, s^o_k is the summed input of the neuron k expressed as $s^o_k = \sum_{j=1}^{numh} h^n_j w^o_{jk}$ with $numh$ the number of neuron in the hidden layer. According to the expression of s^o_k, its derivatives with respect to weight can be computed as

$$\frac{\partial s^o_k}{\partial w^o_{jk}} = h^n_j, \tag{7.5}$$

in which h^n_j is the output of neuron j in the hidden layer n. Replace the first two terms of the right hand side of equation (7.4) with

$$\delta^o_k = \frac{\partial E}{\partial s^o_k} = \frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial s^o_k}, \tag{7.6}$$

the gradient of global loss in equation (7.4) with respect to weights of output layer can be represented as

$$\frac{\partial E}{\partial w^o_{jk}} = \delta^o_k h^n_j. \tag{7.7}$$

Gradient with respect to weights of hidden layer

To compute the gradient of loss E with respect to weight w^n_{ij} as shown in Fig. 7.4, the contributions of hidden neuron h^n_j to all of the output neurons have to be accounted for in the chain rule

$$\frac{\partial E}{\partial w^n_{ij}} = \left[\sum_{k=1}^{numo} \left(\frac{\partial E}{\partial s^o_k} \cdot \frac{\partial s^o_k}{\partial h^n_j} \right) \right] \cdot \frac{\partial h^n_j}{\partial s^n_j} \cdot \frac{\partial s^n_j}{\partial w^n_{ij}}, \tag{7.8}$$

where $numo$ is the number of neuron in the output layer, s^n_j is the summed input of neuron j in the n-th hidden layer and w^n_{ij} is the weight of the connection between neuron i and neuron j. Substituting equation (7.6) into equation (7.8) and simplifying the expression, it will lead to

$$\frac{\partial E}{\partial w^n_{ij}} = \left(\sum_{k=1}^{numo} \delta^o_k \cdot w^o_{jk} \right) \cdot \frac{\partial h^n_j}{\partial s^n_j} \cdot h^{n-1}_i \tag{7.9}$$

$$= \delta^n_j \cdot h^{n-1}_i, \tag{7.10}$$

where h^{n-1}_i is the output of neuron i in the $(n-1)$-th hidden layer.

Training algorithm of FNN

To solve the optimisation problem in neural network training, a proper optimizer has to be chosen to update the weights. The Stochastic Gradient Descent (SGD), also known as incremental gradient descent, is an iterative method that can be used to get the minimized performance function and network weights. In SGD, the network weights are updated by

$$w^{t+1} = w^t - \eta \cdot \frac{\partial E(\boldsymbol{w})}{\partial w^t}, \tag{7.11}$$

where t is the iteration number and η is the learning rate. The SGD algorithm has a simple formulation, however, it is difficult to select the most appropriate learning rate. The performance function often converges to a local optimal resulting in a significant error in the application.
The most common choice of the training algorithm for the FNN is the Levenberg-Marquardt algorithm, which has the second-order training speed without requirement for computing the Hessian matrix. The Levenberg-Marquardt algorithm, see HAGAN (1994), stems from the quasi-Newton method that starts from an expansion of the gradient of loss function

$$\frac{\partial E}{\partial w_i^{n+1}} = \frac{\partial E}{\partial w_i^n} + \frac{\partial^2 E}{\partial (w_i^n)^2} \Delta w_i^n + O(\|\Delta w_i^n\|^2), \tag{7.12}$$

where the second order derivatives contribute to the Hessian matrix

$$H_{ij} = \frac{\partial^2 E}{\partial w_i^n \partial w_j^n}. \tag{7.13}$$

In quasi-Newton method, the large memory of $O(n^2)$ is required to compute and store the inverse of Hessian matrix for the high dimensional loss function. To avoid this problem, the Hessian matrix is approximated in the Levenberg-Marquardt algorithm as

$$\boldsymbol{H} = \boldsymbol{J}^T \boldsymbol{J}, \tag{7.14}$$

where $\boldsymbol{J}$ is the Jacobian matrix that contains first derivatives of network errors e_i with respect to the weights

$$J_{ij} = \frac{\partial e_i}{\partial w_j^n}. \tag{7.15}$$

The Jacobian matrix can be obtained from the back propagation process directly, which is much cheaper than computing the Hessian matrix. In addition, the gradient of the loss function thus can be computed as

$$\frac{\partial E}{\partial w_j^n} = \sum_{i=1}^{n} e_i \frac{\partial e_i}{\partial w_j^n}, \tag{7.16}$$

or in matrix notation

$$\nabla E = \boldsymbol{J}^T \boldsymbol{e}. \tag{7.17}$$

Finally, the weight will be updated as

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n - (\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I})^{-1}\boldsymbol{J}^T\boldsymbol{e}, \tag{7.18}$$

where the vector $\boldsymbol{w}^{n+1}$ contains the weights in iteration $n+1$, μ is a parameter to adaptively control the speed of convergence. When μ is zero, it reproduces the quasi-Newton method, which is faster and more accurate near minimum. When μ is large, it becomes the gradient descent algorithm.

7.3.3 Data collection for the ML based hyperelasticity

To approximate hyperelastic behaviour by the FNN for finite element applications, the first task is to determine the input and output variables for the neural network. Since the loading and the unloading curve coincide for the elastic deformation, as shown in Fig. 7.5, the relationship between the strain space and stress space can be seen as a one-to-one mapping. Hence, the strain-stress mapping can be approximated by the FNN without considering the loading history.

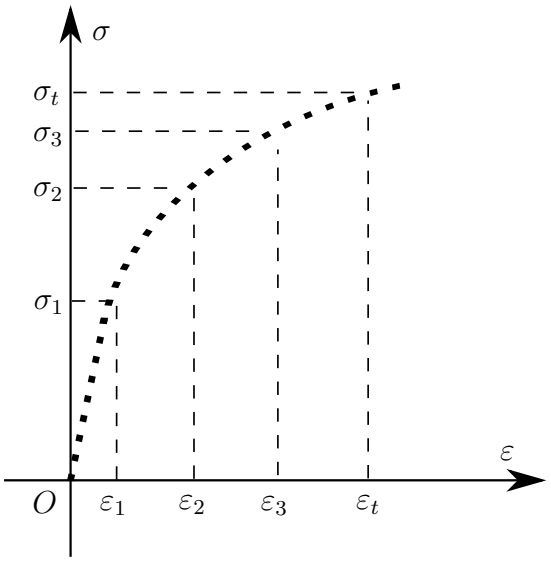


Figure 7.5. The loading and unloading curve for hyperelasticity.

As a ML based material modelling approach, it has to be compatible not only with the unknown material law but also with the well-established material law. Thus, the well-known material model can be applied to validate the developed ML based model by comparing the simulation results. Instead of using experimental data, the training data for hyperelasticity are collected here from the non-linear neo-Hookean model, which is applied as the target model to learn

$$\boldsymbol{\sigma} = \frac{1}{2}\frac{\lambda}{J}(J^2 - 1)\boldsymbol{I} + \frac{\mu}{J}(\boldsymbol{b} - \boldsymbol{I}), \tag{7.19}$$

where the Cauchy stress $\boldsymbol{\sigma}$ and the left Cauchy Green tensor $\boldsymbol{b}$ are symmetric tensors. For the 2D problem, the inputs of the model can be chosen as the strain components $(J, b_{11}, b_{22}, b_{12})$,

whereas the outputs are chosen as the stress components $(\sigma_{11}, \sigma_{22}, \sigma_{12})$. According to the number of input and the output, the architecture of the FNN can be determined as 4-n-3 for instance, where one hidden layer with n neurons is applied for this hyperelastic law. The input data are generated by taking equally spaced points within the given range of strain space. The stresses as output data can be computed from the neo-Hookean model in equation (7.19) accordingly.
Before training the FNN, the generated data are scaled to the range $(-1, 1)$ such that training is accelerated. Then the neural network is trained until the stopping criteria is reached. After training, the weights $\boldsymbol{w}$ and bias $\boldsymbol{b_s}$ will be saved as the model parameters. The ML based hyperelasticity model is thus expressed as

$$\boldsymbol{\sigma}^{NN} = FNN(\boldsymbol{b}, J, \boldsymbol{w}, \boldsymbol{b}_s), \tag{7.20}$$

where $\boldsymbol{\sigma}^{NN}$ is the predicted Cauchy stress by the FNN.

7.3.4 The residual and tangent

The ML based model can be used in the same way as the classical constitutive model in the finite element analysis. The residual for the static problem is given by

$$\boldsymbol{R}(\boldsymbol{u}) = \boldsymbol{f} - \int_\Omega \boldsymbol{B}^T \boldsymbol{\sigma}^{NN} d\Omega, \tag{7.21}$$

$$\boldsymbol{f} = \int_\Omega N\rho\hat{\boldsymbol{b}} dv - \int_{\partial\Omega} N\hat{\boldsymbol{t}} da, \tag{7.22}$$

in which $\boldsymbol{B}$ is the gradient of shape function N, ρ is the density, $\boldsymbol{\sigma}^{NN}$ is the stress computed from the ML based model, $\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{t}}$ are the body force and the surface traction respectively. Due to the non-linearity, the Newton Rapson iterative solution scheme is applied. The tangent matrix is computed by taking the derivative of residual in terms of displacement

$$\boldsymbol{K}_T = \frac{\partial \boldsymbol{R}(\boldsymbol{u})}{\partial \boldsymbol{u}}. \tag{7.23}$$

The derivation of the tangent matrix for the neural network based model requires the computation of derivatives by the chain rule, which will be complex if the number of neurons is very large. In this work, the automatic differentiation tool AceGen, see KORELC & WRIGGERS (2016), based on the symbolic computing in Mathematica is applied, by which the tangent matrix and residual vector can be derived automatically.

7.3.5 Testing the FNN based hyperelasticity in FEM

The material parameters for the neo-Hookean model used in the training data collection are set as $E = 700N/mm^2, \nu = 0.499$. An FNN with the architecture of 4-10-3 is applied, with 4 neurons in the input layer, 10 neurons in the hidden layer and 3 neurons in the output layer. The Levenberg-Marquardt algorithm, see HAGAN (1994), is applied as the training optimizer. After 14082 training iterations, the mean squared error decreased to 0.0326, which costs a training time of $6h40m55s$.

The first example is the uniaxial compression test of a plate in 2D. As shown in Fig. 7.6, the pressure is imposed on the top surface of the plate, the bottom of the plate is fixed in the vertical direction. The distributed load is given as $q_0 = -20MPa$.

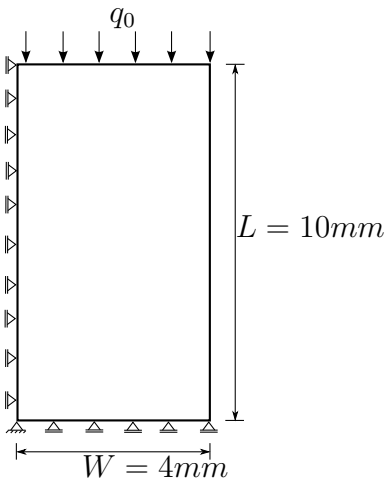
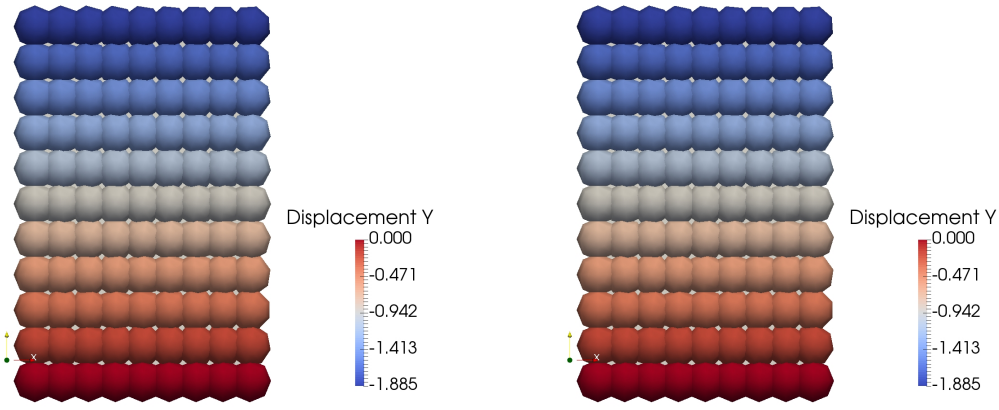


Figure 7.6. Compression of the plate.

The final deformation of the plate computed with the ML based model is compared with the outcome of using the neo-Hookean model in equation (7.19). It can be seen from Fig. 7.7 that the displacements in the vertical direction are very close. The spheres displayed in the numerical examples in Fig. 7.7 and in the remaining figures of this chapter refer to the nodes of the finite element model. The computation time with the analytical hyperelastic model is $8.14s$, whereas the computation time with the ML based model is $9.75s$ on the same computer.



(a) Neo-Hookean model

(b) With FNN based model

Figure 7.7. Deformed state of the plate.

In order to further validate the generalisation, a second test case, the Cook's membrane problem, is conducted. The tapered beam is clamped at the left end and loaded at the right

end by a constant distributed vertical load $q_0 = 5Mpa$, as depicted in Fig. 7.8. The geometric domain of the structure is discretized by 40 quadratic 9-node quadrilateral elements leading to 189 nodes.

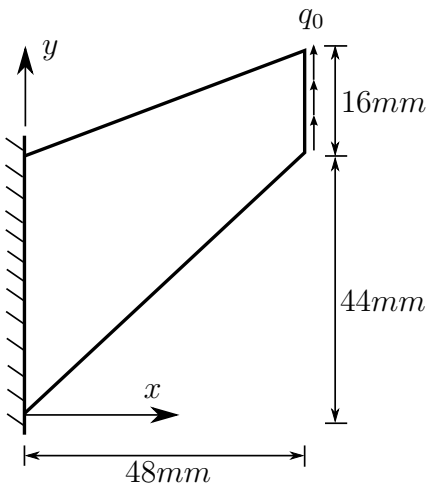
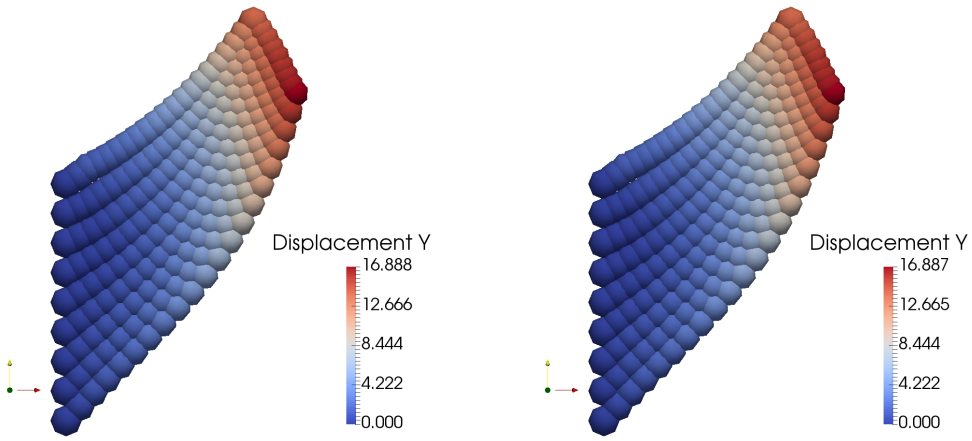


Figure 7.8. Cook's membrane problem.

With the same model as trained in the first test, the final deformation of the membrane is computed and compared. As shown in Fig. 7.9, the vertical displacement in both cases are very close to each other, which highlights the proficient generalisation capabilities of the ML based elasticity model. The computation time with the analytical hyperelastic model is $15.72s$, whereas the computation time with the ML based model is $19.88s$ on the same computer.



(a) Neo-Hookean model

(b) With FNN based model

Figure 7.9. Deformed state of the Cook's membrane.

To this end, it proves that the FNN works well for approximating the nonlinear elastic behaviour as shown in the above results. Elastic deformation is a history independent process

and the stress depends only on current kinematic variables. However, for plastic material behaviour, the response of the deformed material depends not only on the current deformation but also on its loading history. Since the FNN does not have any inherent ability to record loading history, the current approach needs to be improved. Furthermore, the collection process of the training data for plasticity needs to be different from elasticity.

7.4 Data collection strategy for plasticity

The aim of this part is to develop a data-driven material model that can be used to computationally reproduce the plastic material behaviour by means of machine learning tools. Collecting data from experiments is a key part for the data-driven material model. In experiments, only the total strain and stress data of a specimen can be collected, which means the classical concept of elastic-plastic splitting to total strain cannot be applied in the data-driven model. This leads to the questions: how to build the data-driven model using the total strain and stress data available from experiment? And what kind of experiments have to be conducted to collect data?

7.4.1 Concept of sequence for plasticity

Since the plastic flow depends not only on the current stress state but also on the loading history, the plastic deformation is a history dependent process. For 1D plastic deformations, the loading and unloading curves do not coincide as shown in Fig. 7.10, where the strain and stress data are time series of data sets and can be seen as sequences.

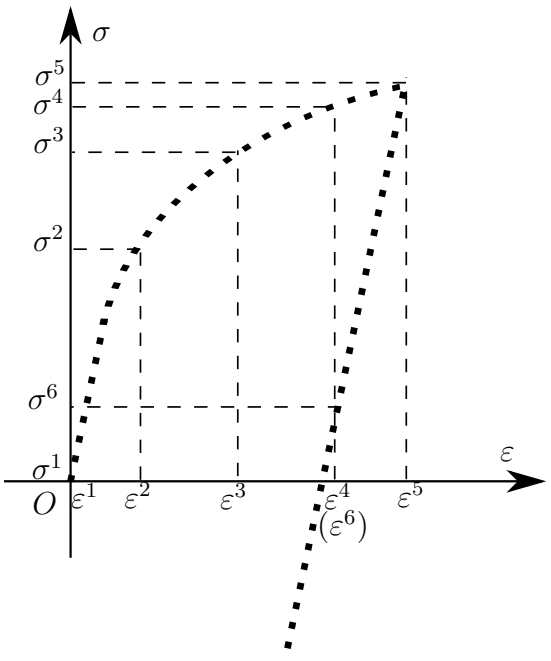


Figure 7.10. The loading and unloading curve for plasticity.

Along the loading-unloading path in Fig. 7.10, the strain and stress data sets of one material point have a strict sequential order and can be written as two sets of corresponding sequences

$$\{\varepsilon^1, \varepsilon^2, \varepsilon^3, ..., \varepsilon^t, ...\} \Leftrightarrow \{\sigma^1, \sigma^2, \sigma^3, ..., \sigma^t, ...\}, \tag{7.24}$$

where ε^t and σ^t are the total strain and stress at time t collected from the experiment. Thus, the basic data unit for a plasticity model is not a strain-stress pair but a strain-stress sequence pair. Each strain-stress sequence pair refers to one loading-unloading path and thus sequence data collected from different loading-unloading paths are required to train a data-driven plasticity model.

In machine learning, the classical equation of plasticity will be replaced with a ML based plasticity model driven by experimental data as

$$\boldsymbol{\sigma}^t = f^{ML}(\boldsymbol{\varepsilon}^t, \boldsymbol{h}^t), \tag{7.25}$$

where $\boldsymbol{h}^t$ is a history variable for distinguishing loading history and $\boldsymbol{\varepsilon}^t$ is the total strain. In this data-driven model, both the input and the output are sequence data. To build a ML based plasticity model, the history data as well as the strain-stress data have to be obtained from experiments.

The choice of history variable is crucial for a successful prediction of sequences. In the literature, the stress and strain in the last step are applied as the history information together with the current strain in the input, see HASHASH ET AL. (2004). However, the previous strain and previous stress are not enough to distinguish the loading history in real applications. Because any error in the predicted previous stress by the ML tool will introduce extra error into the system in an accumulate way. In this work, the accumulated absolute strain is applied in the input as history variable. The accumulated absolute strain component $\varepsilon^t_{acc,j}$ at time step t can be computed for the j-th strain component as

$$h^t_j := \varepsilon^t_{acc,j} = \begin{cases} \sum_{i=3}^{t} |\varepsilon_j^{i-1} - \varepsilon_j^{i-2}|, & t \geqslant 3, \\ 0, & t = 1, 2, \end{cases} \tag{7.26}$$

where ε_j^{i-1} and ε_j^{i-2} are strains at time step $i-1$ and $i-2$ respectively. This history variable has to be computed for each total strain component.$|\varepsilon_j^{i-1} - \varepsilon_j^{i-2}|$ is the absolute increment of strain component j from time step $i-2$ to $i-1$, which is necessary to distinguish the loading history when tension and compression loadings are mixed, such as in the loading-unloading paths.

For the 1D case, depicted in Fig. 7.10, a monotonic loading (e.g. from σ^1 to σ^4) and a mixed loading-unloading (e.g. from σ^1 to σ^6) may lead to the same total strain (e.g. $\varepsilon^4 = \varepsilon^6$). To distinguish monotonic loading from mixed loading-unloading, the absolute value of strain increment $|\varepsilon^{i-1} - \varepsilon^{i-2}|$ is applied in equation (7.26), which leads to different accumulated

absolute strain for different paths, e.g.

$$\varepsilon^6_{acc} = \sum_{i=3}^{6} |\varepsilon^{i-1} - \varepsilon^{i-2}| \tag{7.27}$$

$$= \sum_{i=3}^{4} |\varepsilon^{i-1} - \varepsilon^{i-2}| + \sum_{i=5}^{6} |\varepsilon^{i-1} - \varepsilon^{i-2}| \tag{7.28}$$

$$= \varepsilon^4_{acc} + \sum_{i=5}^{6} |\varepsilon^{i-1} - \varepsilon^{i-2}| > \varepsilon^4_{acc}. \tag{7.29}$$

Note that $|\varepsilon^{i-1} - \varepsilon^{i-2}|$ is applied instead of $|\varepsilon^{i} - \varepsilon^{i-1}|$ in equation (7.26), since $\sum_{i=3}^{t} |\varepsilon^{i} - \varepsilon^{i-1}|$ is equal to ε^t for monotonic loading, and it is not a history variable but the current input. The advantage of applying the accumulated absolute strain as the history variable is that it can be obtained from the existing experimental data without the effort to collect them additionally.

7.4.2 Loading paths to collect data from experiments

To collect the strain-stress sequence data from experiments, the loading paths required in experiments have to be investigated. Since isotropic plasticity can be formulated in the principle strain-stress space, the ML based plasticity model can be formulated in the principle space as well, where the input and output of the model are principle strain and principle stress respectively. Therefore only the principle strain and principle stress are required to be collected from the experiments.

To collect the principle strain and principle stresses, the multi-axial loading tests can be conducted on specially designed specimens, such as the biaxial experiments conducted by MOHR ET AL. (2010) and the loading paths suggested by GOEL ET AL. (2011). The von Mises yield surface covering different stress states is shown in Fig. 7.11 for the 2D case. In order to learn the plasticity behaviour by e.g. artificial neural networks, yielding as well as hardening effects have to be captured implicitly. To fully describe the stress states in the deformed structures, the data have to be collected from several tests under different loading-unloading paths. However, only biaxial tests for 2D and triaxial tests for 3D are required to collect data in principle space.

In experiments, the principle strain and stress data can be collected within a homogeneous region at one point within a specimen, which can be described by a quadrilateral region for 2D case depicted in Fig. 7.12. The biaxial loading-unloading paths in this region can be characterized by the displacements of the edges connected to point A.

For each biaxial loading-unloading path, the node A moves from its original position to A' for loading and then goes back to original position for unloading, during which the displacements (u_x, u_y) will increase linearly from zero to a specific value obeying $\sqrt{u_x^2 + u_y^2} = r_i, (i = 1, 2, 3...)$ and then decrease to zero. $max(r_i)$ has to be large enough to capture as much loading range of plasticity deformation as possible. As shown in Fig. 7.13, the loading-unloading paths are just determined by setting a radius r_i and different values of the angles ϕ. Since the unloading can start from different positions, multiple circles with radius r_i have to be applied in data collection.

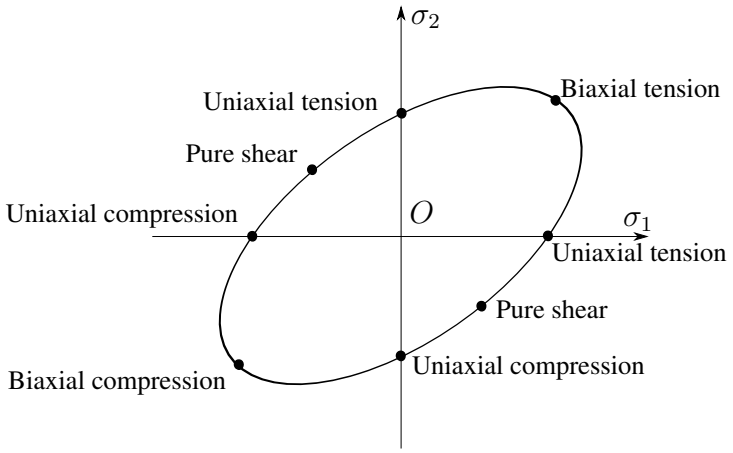


Figure 7.11. Stress states in 2D.

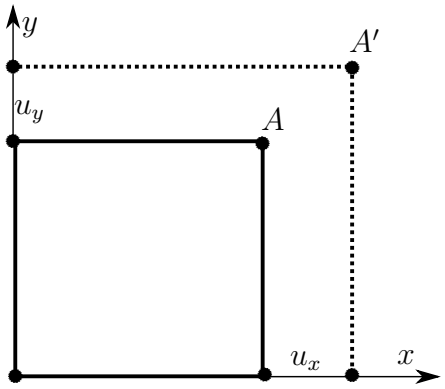


Figure 7.12. Quadrilateral region within a specimen to collect data with biaxial loading.

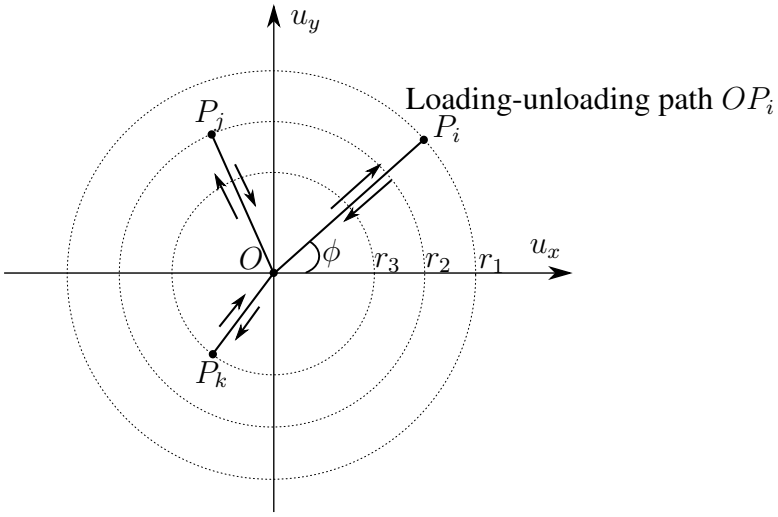


Figure 7.13. Loading-unloading paths for data collection in 2D.

For the 3D case, data can be collected from a cubic region, as shown in Fig. 7.14, where the triaxial loading-unloading paths at this point can be characterized by the displacement (u_x, u_y, u_z) at the point A.

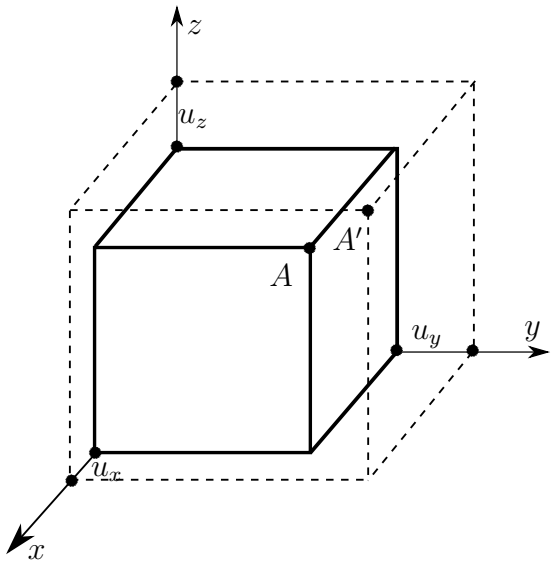


Figure 7.14. Cubic region within a specimen to collect data with triaxial loading.

The loading-unloading path for the 3D case can be generated in the spherical coordinate system as shown in Fig. 7.15, where the displacement of node A obeys $\sqrt{u_x^2 + u_y^2 + u_z^2} = r_i, (i = 1, 2, 3, ...)$. The loading path OP is distinguished by the angles ϕ and θ with radius r_i. By looping the loading path OP around the sphere, the whole range of stress states can be included.

For example, along the loading path OP_i in Fig. 7.15, the strain-stress sequence data will be collected firstly as

$$\hat{\boldsymbol{\varepsilon}} = \begin{bmatrix} \varepsilon_1^1 & \varepsilon_1^2 & \dots & \varepsilon_1^t & \dots \\ \varepsilon_2^1 & \varepsilon_2^2 & \dots & \varepsilon_2^t & \dots \\ \varepsilon_3^1 & \varepsilon_3^2 & \dots & \varepsilon_3^t & \dots \end{bmatrix}_{3\times np}, \quad \boldsymbol{\sigma}_{opi} = \begin{bmatrix} \sigma_1^1 & \sigma_1^2 & \dots & \sigma_1^t & \dots \\ \sigma_2^1 & \sigma_2^2 & \dots & \sigma_2^t & \dots \\ \sigma_3^1 & \sigma_3^2 & \dots & \sigma_3^t & \dots \end{bmatrix}_{3\times np}, \tag{7.30}$$

where np is the number of data point on the loading path OP_i, $\varepsilon_i^t (i = 1, 2, 3)$ are the principle strains at time t measured in the cubic region within the specimen, $\sigma_i^t (i = 1, 2, 3)$ are the principle stresses at time t in that region, $\hat{\boldsymbol{\varepsilon}}$ is the strain sequence and $\boldsymbol{\sigma}_{opi}$ is the stress sequence of path OP_i. Then the history data, accumulated absolute strain ε_{acc}^t, are computed from the strain sequence $\hat{\boldsymbol{\varepsilon}}$ using equation (7.26). The final strain sequence data of path OP_i are obtained by combining the total strain sequence and the history variable sequence as

$$\boldsymbol{\varepsilon}_{opi} = \begin{bmatrix} \varepsilon_1^1 & \varepsilon_1^2 & \dots & \varepsilon_1^t & \dots \\ \varepsilon_2^1 & \varepsilon_2^2 & \dots & \varepsilon_2^t & \dots \\ \varepsilon_3^1 & \varepsilon_3^2 & \dots & \varepsilon_3^t & \dots \\ \varepsilon_{acc,1}^1 & \varepsilon_{acc,1}^2 & \dots & \varepsilon_{acc,1}^t & \dots \\ \varepsilon_{acc,2}^1 & \varepsilon_{acc,2}^2 & \dots & \varepsilon_{acc,2}^t & \dots \\ \varepsilon_{acc,3}^1 & \varepsilon_{acc,3}^2 & \dots & \varepsilon_{acc,3}^t & \dots \end{bmatrix}_{6\times np}, \tag{7.31}$$

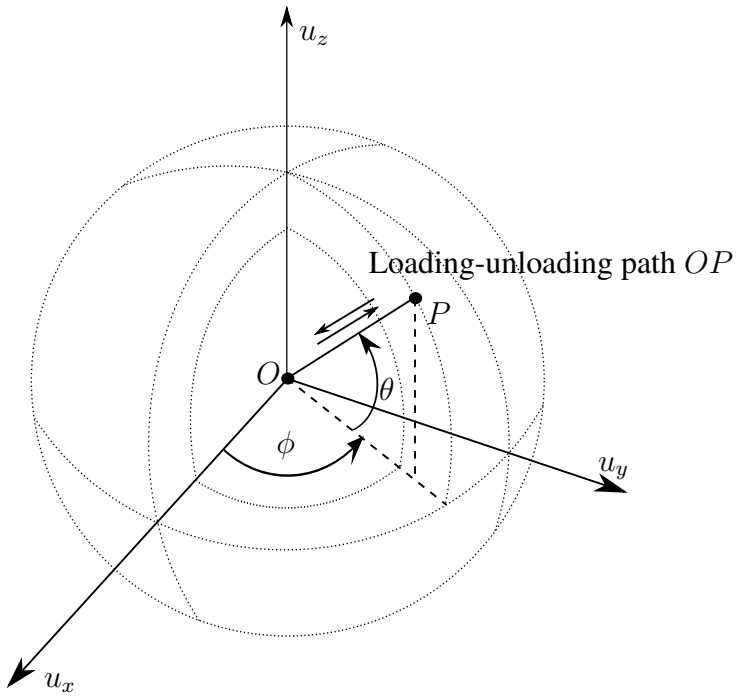


Figure 7.15. Loading-unloading paths for data collection in 3D.

where $\varepsilon^t_{acc,1}$, $\varepsilon^t_{acc,2}$ and $\varepsilon^t_{acc,3}$ are computed from the strain components ε^t_1, ε^t_2 and ε^t_3 respectively according to equation (7.26). Finally, the input and output data are obtained by combining the sequences from all of loading paths $OP_i, (i = 1, 2, ..., nl)$ as

$$\boldsymbol{I}_\varepsilon = \begin{bmatrix} \boldsymbol{\varepsilon}_{op1} & \boldsymbol{\varepsilon}_{op2} & ... & \boldsymbol{\varepsilon}_{opi} & ... \end{bmatrix}_{6\times M}, \quad \boldsymbol{O}_\sigma = \begin{bmatrix} \boldsymbol{\sigma}_{op1} & \boldsymbol{\sigma}_{op2} & ... & \boldsymbol{\sigma}_{opi} & ... \end{bmatrix}_{3\times M}, \tag{7.32}$$

where nl is the number of loading path and $M = np \times nl$.

7.4.3 Data collection from analytical model

Apart from collecting the training data from experiments, simulation data using the von Mises plasticity model can be collected as well to train the ML tool, in this way the performance of the ML based plasticity model can be verified by comparing to the analytical model. In this work, the strain-stress sequences are collected at a Gauss point of a finite element under specific loading paths described above. The strain is computed as the symmetric part of the displacement gradient for small deformations

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\boldsymbol{H} + \boldsymbol{H}^T), \tag{7.33}$$

where $\boldsymbol{H}$ is the displacement gradient with $\boldsymbol{H} = \mathrm{Grad}\boldsymbol{u}$. Using spectral decomposition, the strain can be formulated as

$$\boldsymbol{\varepsilon} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T, \tag{7.34}$$

where $\mathbf{\Lambda}$ is the principle strain and $\boldsymbol{Q}$ is the rotation matrix obtained from the eigendirections. The principle strain $\mathbf{\Lambda}$ along different loading paths will serve as input data to train the ML based plasticity model.
For small strain plasticity, the additive decomposition of strain into elastic and plastic parts is assumed

$$\mathbf{\Lambda} = \mathbf{\Lambda}^e + \mathbf{\Lambda}^p, \tag{7.35}$$

where $\mathbf{\Lambda}^e$ and $\mathbf{\Lambda}^p$ are the elastic strain and plastic strain, respectively. The plastically admissible stress is given by

$$\mathbf{\Sigma} = \rho \frac{\partial \psi}{\partial \mathbf{\Lambda}^e}, \tag{7.36}$$

where ρ is the mass density and ψ is called the specific free energy or alternatively the specific strain energy. The unit is energy per volume. In this work, the linear elasticity is assumed in the elastic part of deformation with the free energy density $\psi = \frac{\lambda \mathrm{tr}^2(\mathbf{\Lambda}^e)}{2} + \mu \mathrm{tr}(\mathbf{\Lambda}^{e,2})$, where λ and μ are the Lamé constants. The principle stress $\mathbf{\Sigma}$ will serve as the output data, corresponding to the principle strain as input. By assuming the von Mises yield criteria, the yield function is written as

$$f = \sqrt{\frac{3}{2}} \|\mathbf{\Sigma}^{dev}\| - \sigma_y(\alpha), \tag{7.37}$$

where $\mathbf{\Sigma}^{dev}$ is the deviatoric stress with $\mathbf{\Sigma}^{dev} = \mathbf{\Sigma} - \frac{1}{3} tr\mathbf{\Sigma} \cdot \mathbf{1}$ and α is the isotropic hardening variable. By using the associated plastic flow rule, the evolution equations for the principle plastic strain and the hardening variable are formulated as

$$\dot{\mathbf{\Lambda}}^p = \dot{\gamma} \frac{\partial f}{\partial \mathbf{\Sigma}^{dev}}, \quad \dot{\alpha} = \dot{\gamma} \frac{\partial f}{\partial A} \tag{7.38}$$

where γ is the plastic multiplier and A is the thermodynamic force conjugate with α. The plastic flow has to fulfill the Kuhn-Tucker conditions

$$f \leqslant 0, \quad \dot{\gamma} \geqslant 0, \quad \dot{\gamma} f = 0. \tag{7.39}$$

By applying the return mapping algorithm, the increment of plastic multiplier and the elastic strain will be calculated by solving the yield equation and the evolution equation. The principle Cauchy stress can thus be updated by the linear elastic law. Then the Cauchy stress is computed by transforming the principle Cauchy stress into the general space

$$\boldsymbol{\sigma} = \boldsymbol{Q} \cdot \mathbf{\Sigma} \cdot \boldsymbol{Q}^T. \tag{7.40}$$

Based on the above formulation in principle space, the principle strain $\mathbf{\Lambda}$ and principle stress $\mathbf{\Sigma}$ will serve as the input and output of the ML based plasticity model, respectively.
After the data collection, the ML technology is applied to learn the constitutive law behind the data. In this work, two ML approaches, RNN and PODFNN, are applied for the data-driven material modelling, which will be introduced in the following sections.

7.5 ML based plasticity using Recurrent Neural Network (RNN)

The Recurrent Neural Network (RNN) has been applied in many fields to solve the sequential problem, such as the recognition of speech and prediction of time-varying patterns in a dynamic system. Since RNN has the internal state variables that can record the history of the sequence data, the history dependent model can be reproduced without including additional history information in the inputs of the neural network. According to the descriptions in section 7.3.1, the strain-stress data along the loading paths of the plasticity model can be seen as the time sequence data. Thus, RNN is applied to learn the history dependent plastic behaviour from data in this work.

7.5.1 Network architecture

As shown in Fig. 7.16, the RNN is composed of an input layer, a hidden layer with the Long Short Term Memory (LSTM) neurons and an output layer. Similar to FNN, the input layer of RNN is fully connected with the hidden layer and the output layer is attached on the hidden layer as a regression layer. The difference is that the hidden layer of RNN contains specially designed neurons, the LSTM neurons, where the data generated during the last time step will be stored in the state variables and applied as input data in the next time step.

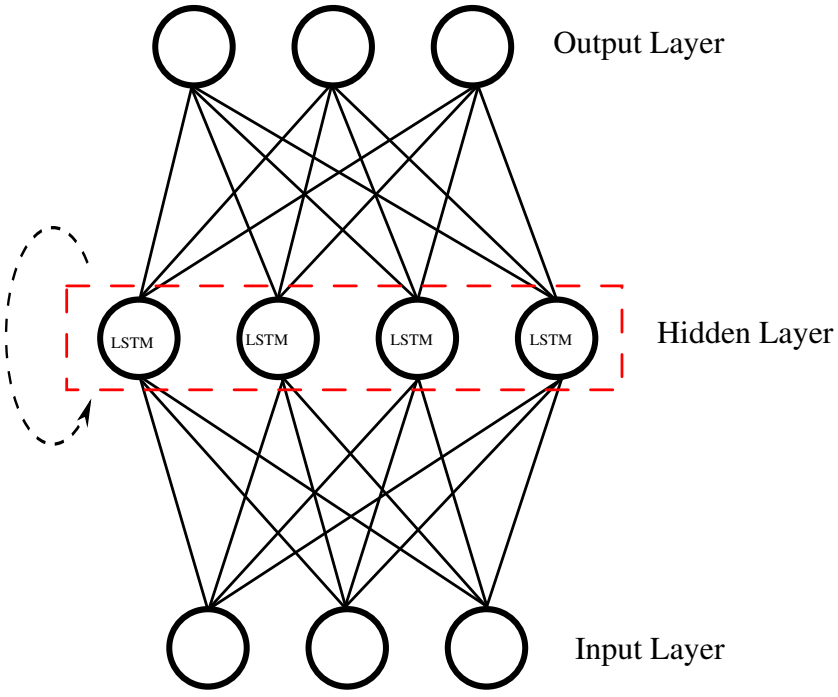


Figure 7.16. The recurrent neural network.

To get a better understanding of RNN, the data flow within one LSTM neuron is unfolded along with the time steps as shown in Fig. 7.17. The unfolded LSTM neuron can be seen

as a copied chain-like network along with the time series, where each LSTM neuron will receive the history information from the previous one and then pass the information to its successor within each prediction step. The information communicated between time steps is contained by the cell state $\boldsymbol{c}_t$ and the hidden state $\boldsymbol{h}_t$ in the network. At the same time, the network predicts an output $\boldsymbol{y}_t$ based on the current input $\boldsymbol{x}_t$ at each time step. Thus, the input and the output of the LSTM neuron can be seen as the sequence data, which allows the recurrent neural network to deal with the history dependent problem. If the strain sequence is applied as the input and the stress sequence is applied as the output, the history dependent plastic behaviour can be learned by the LSTM neurons within the recurrent neural network, which is the objective of this part.

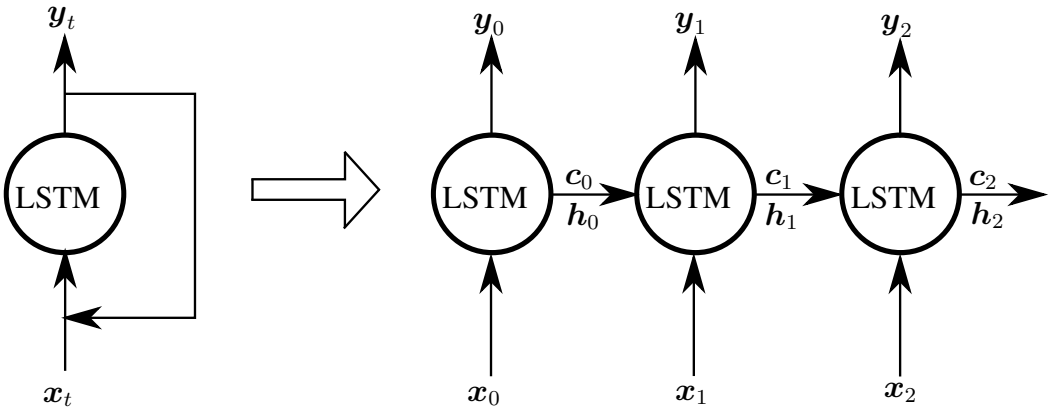


Figure 7.17. The unfolded recurrent neural network.

7.5.2 The RNN based plasticity model

According to the sequence data sets collected in the last section, the RNN based plasticity model will be formulated within the principle space, where the principle strain components act as the input and principle stress components act as the output. The loading history will be taken into account automatically by the internal state variables of the RNN.

Input layer

The input of the RNN based model is the principle strain, which is computed by the spectral decomposition of small strain in equation (7.34). The principle strain vector as the input is thus obtained as $\boldsymbol{\varepsilon} = \{\boldsymbol{\Lambda}_{11}, \boldsymbol{\Lambda}_{22}, \boldsymbol{\Lambda}_{33}\}$. Before feeding into the input layer of RNN, the principle strain has to be scaled into a specific range, which allows the faster convergence during the neural network training. The strain component $\varepsilon_{i,t}$ at time t is scaled by

$$\varepsilon^s_{i,t} = v_{min} + (v_{max} - v_{min})\frac{\varepsilon_{i,t} - \varepsilon_{i,min}}{\varepsilon_{i,max} - \varepsilon_{i,min}}, \quad i = 1, 2, 3. \tag{7.41}$$

where $\varepsilon^s_{i,t}$ is the scaled strain component, $\varepsilon_{i,max}$ and $\varepsilon_{i,min}$ are the maximum and minimum value of the input strain, v_{max} and v_{min} are the scaling limits. The range of the input data and

the scaling data will be identical for both training and testing of the neural network based model. After scaling, the input vector $\boldsymbol{x}_t$ of the RNN is written as

$$\boldsymbol{x}_t = \begin{bmatrix} \varepsilon^s_{1,t} & \varepsilon^s_{2,t} & \varepsilon^s_{3,t} \end{bmatrix}. \tag{7.42}$$

Since the neurons in the input layer are fully connected with the hidden layer, the input $\boldsymbol{x}_t$ will be feed into the hidden layer by multiplying the weighs.

Hidden layer

As the key part of RNN, the LSTM neurons play a very important role in dealing with the history dependent problem. The LSTM neuron has the memory on its prediction history and thus can learn long-term dependencies between data sets in a sequence. As shown in Fig. 7.18, the LSTM neuron is composed by four gates: the forget gate f, the input gates (g, i) and the output gate o. The forget gate controls the length of the network memory through the activation function. The history information is contained by the cell state c and the hidden state h. The current input x_t and the state variables (c_{t-1}, h_{t-1}) are feed into the neuron from the left hand side, then the data flow from the left hand side to the right hand side of the LSTM neuron by passing through all of the gates. Finally, the neuron will export an output y_t and the state variables (c_t, h_t).

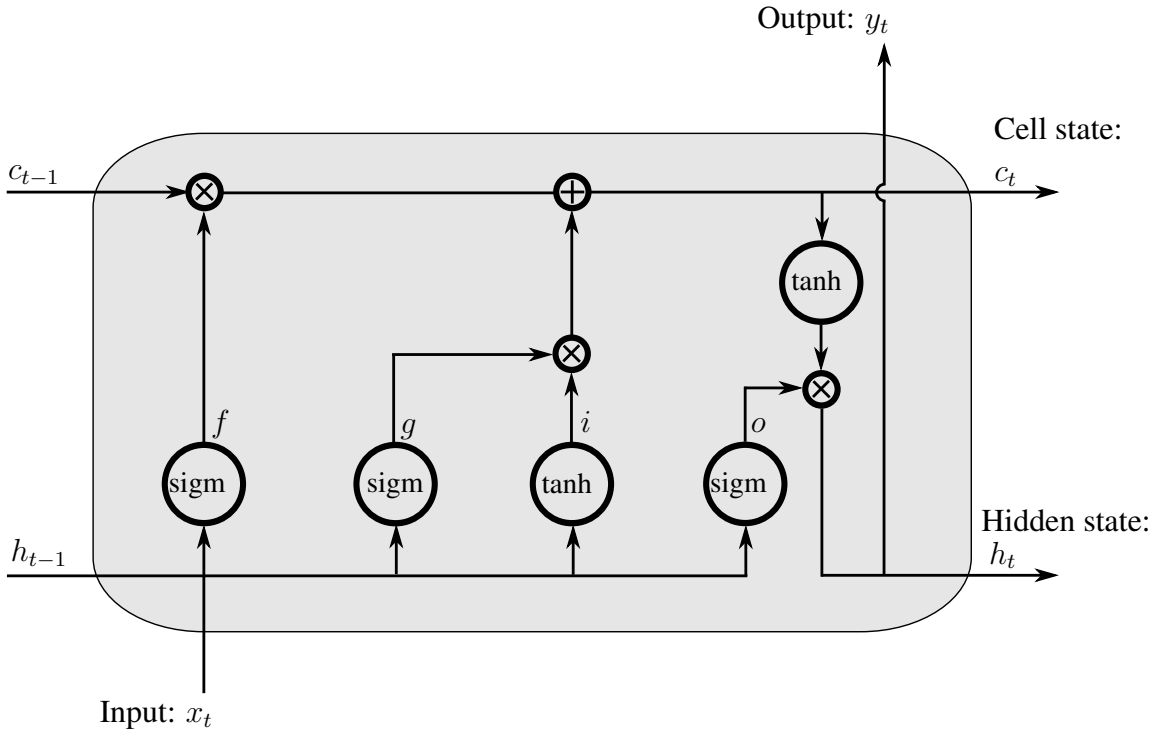


Figure 7.18. The architecture of LSTM neuron.

In the hidden layer of the RNN, multiple LSTM neurons are fully connected with the neurons of input layer. The variables of the LSTM neurons in hidden layer will be stored as vectors.

The gate vectors $\boldsymbol{f}_t$, $\boldsymbol{g}_t$, $\boldsymbol{i}_t$ and $\boldsymbol{o}_t$ at time t are computed from the input vector $\boldsymbol{x}_t$ and the hidden state vector $\boldsymbol{h}_{t-1}$ at time $t-1$. The gate vectors are thus formulated as

$$\boldsymbol{f}_t = f_s(\boldsymbol{W}_f \cdot \boldsymbol{x}_t + \boldsymbol{R}_f \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_f), \tag{7.43}$$
$$\boldsymbol{g}_t = f_t(\boldsymbol{W}_g \cdot \boldsymbol{x}_t + \boldsymbol{R}_g \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_g), \tag{7.44}$$
$$\boldsymbol{i}_t = f_s(\boldsymbol{W}_i \cdot \boldsymbol{x}_t + \boldsymbol{R}_i \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_i), \tag{7.45}$$
$$\boldsymbol{o}_t = f_s(\boldsymbol{W}_o \cdot \boldsymbol{x}_t + \boldsymbol{R}_o \cdot \boldsymbol{h}_{t-1} + \boldsymbol{b}_o), \tag{7.46}$$

where $\boldsymbol{W}_f$, $\boldsymbol{W}_g$, $\boldsymbol{W}_i$ and $\boldsymbol{W}_o$ are the input weight matrices corresponding to the four gates, $\boldsymbol{R}_f$, $\boldsymbol{R}_g$, $\boldsymbol{R}_i$ and $\boldsymbol{R}_o$ are the recurrent weight matrices for the four gates, $\boldsymbol{b}_f$, $\boldsymbol{b}_g$, $\boldsymbol{b}_i$ and $\boldsymbol{b}_o$ are the biases, $f_s(x)$ denotes the sigmoid activation function in equation (7.2) and $f_t(x)$ denotes the hyperbolic tangent activation function

$$f_t(x) = \frac{e^{2x}-1}{e^{2x}+1}. \tag{7.47}$$

In the implementation, the weight matrices and biases are stored in matrices as

$$\boldsymbol{W}_I = \begin{bmatrix} \boldsymbol{W}_i \\ \boldsymbol{W}_f \\ \boldsymbol{W}_g \\ \boldsymbol{W}_o \end{bmatrix}_{[4Nh \times Ni]}, \quad \boldsymbol{W}_R = \begin{bmatrix} \boldsymbol{R}_i \\ \boldsymbol{R}_f \\ \boldsymbol{R}_g \\ \boldsymbol{R}_o \end{bmatrix}_{[4Nh \times Nh]}, \quad \boldsymbol{B}_I = \begin{bmatrix} \boldsymbol{b}_i \\ \boldsymbol{b}_f \\ \boldsymbol{b}_g \\ \boldsymbol{b}_o \end{bmatrix}_{[4Nh \times 1]}, \tag{7.48}$$

in which $\boldsymbol{W}_I$ is the input weight matrix, $\boldsymbol{W}_R$ is the recurrent weight matrix, $\boldsymbol{B}_I$ is the bias matrix, Ni is the number of input and Nh is the number of hidden neuron in the LSTM layer. Once the gate values are computed, the cell state $\boldsymbol{c}_t$ and the hidden state $\boldsymbol{h}_t$ for the next step will be updated as

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t, \tag{7.49}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot f_t(\boldsymbol{c}_t), \tag{7.50}$$

where the symbol $\odot$ denotes the element wise multiplication of two vectors. The output of the hidden layer $\boldsymbol{y}_t$ is identical with the hidden state $\boldsymbol{h}_t$.

Output layer

As shown in Fig. 7.16, a fully connected regression layer is attached behind the hidden layer in RNN. The output of RNN can be computed from the output of the hidden layer as

$$\boldsymbol{\sigma}_t^s = \boldsymbol{W}_{out} \cdot \boldsymbol{h}_t + \boldsymbol{b}_{out}, \tag{7.51}$$

where $\boldsymbol{W}_{out}$ and $\boldsymbol{b}_{out}$ are the weight and bias of the output layer. Since the stress data have been scaled to the range of (q_{min}, q_{max}) during training process, the predictions of RNN $\boldsymbol{\sigma}_t^s$ are the scaled values of stress as well. The real value of stress components can be descaled as

$$\sigma_{i,t} = \sigma_{i,min} + (\sigma_{i,max} - \sigma_{i,min}) \frac{\sigma_{i,t}^s - q_{min}}{q_{max} - q_{min}}, \quad i = 1, 2, 3. \tag{7.52}$$

where $\sigma_{i,max}$ and $\sigma_{i,min}$ are the maximum and minimum value of the real stress components, respectively. By using the same transformation matrix $\boldsymbol{Q}$ in spectral decomposition of strain, the general stress can be compute as

$$\boldsymbol{\sigma}^t_{RNN} = \boldsymbol{Q}\boldsymbol{\Sigma}^t_{RNN}\boldsymbol{Q}^T, \tag{7.53}$$

where diagonal matrix $\boldsymbol{\Sigma}$ contains the principle stress components $\sigma_{i,t}$.

7.5.3 Training algorithm of RNN

After collecting the sequence data, the RNN will be trained by the use of proper training algorithm. Similar to the FNN training, the RNN training is an optimisation problem as well, where the objective is to minimize the global error. The mean squared error, defined by the difference between the neural network prediction and the target value in equation (7.3), is applied to evaluate the training performance and thus as the objective function. To update the weights of RNN, the back propagation algorithm will be employed to compute the error gradients. Furthermore, the Adaptive Momentum (Adam, see KINGMA & BA (2014)) algorithm is applied as the training algorithm for the RNNs.

Adam algorithm

Adam algorithm is also a gradient based method, which seeks to improve network training by using different learning rates for different weights. At the same time, the adaptive learning rate is set for each optimisation step. The Adam algorithm computes the moving average of the gradient and the moving average of the squared value of gradient as

$$m^n_i = \beta_1 \cdot m^{n-1}_i + (1-\beta_1) \cdot \frac{\partial E(\boldsymbol{w})}{\partial w^n_i}, \tag{7.54}$$

$$v^n_i = \beta_2 \cdot v^{n-1}_i + (1-\beta_2) \cdot \left[\frac{\partial E(\boldsymbol{w})}{\partial w^n_i}\right]^2, \tag{7.55}$$

where m^n_i and v^n_i are the moving averages that estimates the first moment (the mean) and the second moment (the uncentered variance) of the gradients respectively, β_1 and β_2 are the gradient decay factor and squared gradient decay factor respectively, w^n_i is the i-th weight at time step n. The network weights are updated by the moving averages as

$$w^{n+1}_i = w^n_i - \alpha \frac{m^n_i}{\sqrt{v^n_i} + \epsilon}, \tag{7.56}$$

where α is the learning rate, ϵ is a parameter which makes the denominator nonzero. The common values of β_1, β_2 and ϵ are chosen as 0.9, 0.999 and 10^{-8} respectively.

Adaptive learning rate schedule

The learning rate α is an important argument of the Adam optimizer. Before training, the learning rate is initialized with a specific value by experimenting with different values. Instead of using constant learning rate during the training process, the learning rate is gradually

reduced as the training progresses, which allows that the optimisation converges to the global minimum and circumvents the oscillation when the mean squared error becomes very small. The learning rate reduction can be conducted by using pre-defined learning rate schedules. In this work, the step decay schedule is applied where the learning rate is reduced by a factor of 0.2 every 2000 epochs.

7.5.4 The residual and tangent

After training, the weights $\boldsymbol{w}$ of RNN will be stored as the material parameters of the RNN based plasticity model. To apply the RNN based plasticity model in the finite element analysis, the residual vector and tangent matrix have to be derived. The RNN based plasticity model defining an incremental constitutive function which can be formulated as

$$\boldsymbol{\sigma}^t_{RNN} = \hat{\boldsymbol{\sigma}}(\boldsymbol{\varepsilon}_t, \boldsymbol{h}_{t-1}, \boldsymbol{c}_{t-1}, \boldsymbol{w}), \tag{7.57}$$

where $\boldsymbol{w}$ contains the weights and bias of the neural network, $\boldsymbol{h}_{t-1}$ and $\boldsymbol{c}_{t-1}$ are the state variables of the RNN, $\boldsymbol{\sigma}^t_{RNN}$is the stress predicted by the RNN based model. The residual for the static problem is given by

$$\boldsymbol{R}(\boldsymbol{u}) = \boldsymbol{f} - \int_\Omega \boldsymbol{B}^T \boldsymbol{\sigma}_{RNN} d\Omega, \tag{7.58}$$

$$\boldsymbol{f} = \int_\Omega N\rho\hat{\boldsymbol{b}} dv - \int_{\partial\Omega} N\hat{\boldsymbol{t}} da. \tag{7.59}$$

The stiffness is computed by taking the derivative of residual in terms of displacement

$$\boldsymbol{K}_T = \frac{\partial \boldsymbol{R}(\boldsymbol{u})}{\partial \boldsymbol{u}}. \tag{7.60}$$

To derive the tangent matrix for the RNN based material model by the chain rule, the automatic symbolic differentiation tool AceGen, see KORELC & WRIGGERS (2016), is applied.

7.5.5 Testing the RNN based plasticity model in FEM

Uniaxial tension and compression

To test the performance of the RNN based plastic model, the 1D uniaxial tension and compression test are conducted firstly. The von Mises plasticity with the linear isotropic hardening is applied as the target model. The material parameters of the plasticity model are set as: Young's modulus $E = 700N/mm^2$, yield stress $\sigma_y = 100MPa$ and isotropic hardening parameter $H_{iso} = 10MPa$. To prepare the training data, 11 sets of strain-stress sequence data are collected from the target model with strain increment being increased linearly from 0.02 to 0.03.

The RNN with an architecture of (1-10-1) is applied in the model, where the input layer with 1 neuron is connected with the hidden layer composed of 10 LSTM neurons. A fully connected layer with 1 neuron is attached as the output layer. The input of the RNN is the

total strain sequence and the output is the stress sequence. After 1000 epochs of training, the mean squared error is decreased to 10^{-4} and the training progress is terminated.
The RNN based material model is validated using a strain sequence with different strain increment (0.15) from the training data. The stress computed from the RNN based plasticity model is compared with the stress from the target plasticity model, as shown in Fig. 7.19. It can be seen that the predicted stress follows the exact solution well, which validates the effectiveness of RNN for dealing with history dependent problem. Furthermore, the result shows that the state variables inside the RNN fully captures the loading history under the cyclic loading condition.

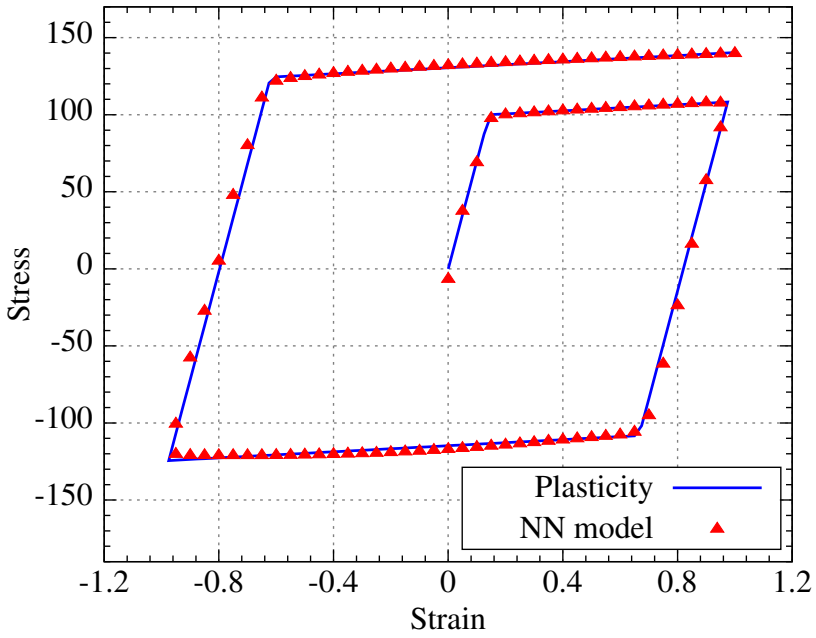


Figure 7.19. Uniaxial tension and compression.

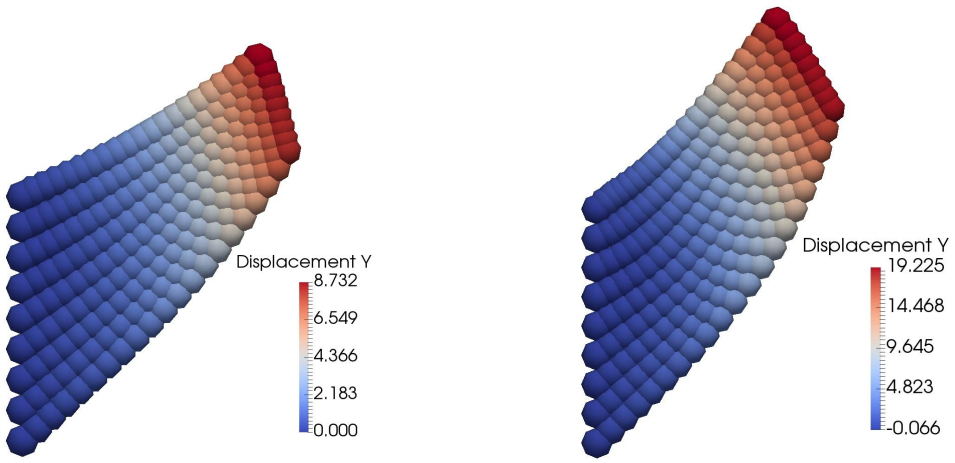
Application in FEM analysis

To evaluate the performance of the RNN based plasticity model, the benchmark test in 2D finite element is presented here. The strain-stress sequences as training data are generated by the biaxial tension and compression test introduced in section 7.4. The von Mises plasticity with an exponential isotropic hardening law $\sigma_y = y_0 + y_0(0.00002 + \gamma)^{0.3}$ is set as the target model, where γ is the isotropic hardening strain. The material parameters are set as: Young's modulus $E = 1N/mm^2$, Poison's ratio $\nu = 0.33$, and the initial yield stress $y_0 = 0.05MPa$. To collect the training data, 150 loading paths evenly distributed within the circle in Fig. 7.13 are selected to conduct the biaxial tests.
The network architecture (2-20-2) is applied for the RNN, where the input layer contains 2 neurons is connected with a hidden layer with 20 LSTM neurons. The output layer contains 2 neurons. The principle total strain sequences serve as the inputs whereas the principle

stress sequences are the outputs. The Adam algorithm is applied as an optimizer in training, where the initial learning rate is set to be 0.01 and then drops with a factor of 0.8 every 2000 epochs. The training progress is terminated when the mean squared error is decreased to 3×10^{-4}.

After network training, the weights and biases of the RNN are stored as the constant parameters for the RNN based plasticity model. The tangent matrix and the residual vector of the finite element formulation are derived using the automatic symbolic differentiation tool AceGen. The 2D RNN based plasticity model is tested by the Cook's membrane problem. The beam is clamped at the left end by fixing in x and y directions and loaded at the right end by a constant distributed vertical load $q_0 = 0.03MPa$ as depicted in Fig. 7.8. The geometric domain of the structure is discretized by 40 quadratic 9-node quadrilateral elements leading to 189 nodes.

The final deformation state of the beam using the proposed RNN based plasticity model is compared with the target plasticity model, which is depicted in Fig. 7.20. It can be observed that the vertical displacement of the structure computed by the RNN based model is larger than that with the reference plasticity model.



(a) with plasticity model (b) with RNN model

Figure 7.20. Final deformation state of the 2D Cook's membrane.

The load deflection curve of the upper node $(48, 60)$ at the right end of the cantilever beam is plotted in Fig. 7.21. The figure shows that the RNN based plasticity model is not accurate enough to capture the plastic behaviour. The reason can be attributed to the insufficient training data or the limitation of training algorithm, which are the topics of future work. Alternatively, another machine learning approach using the feed forward neural network is applied to learn the plasticity in this work, which will be introduced in the next section.

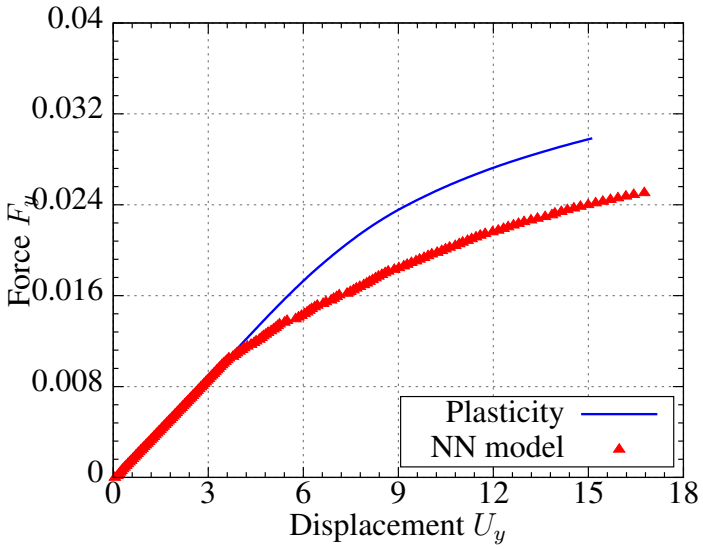


Figure 7.21. Load deflection curve of the 2D Cook's membrane.

7.6 ML based plasticity using Feed forward Neural Network (FNN)

In this section, a novel method called Proper Orthogonal Decomposition Feed forward Neural Network (PODFNN), which in combination with the introduced history variable is proposed for predicting the stress sequences in case of plasticity. By means of the Proper Orthogonal Decomposition (POD), the stress sequence is transformed into multiple independent coefficient sequences, where the stress at any time step can be recovered by a linear combination of the coefficients and the basis. The presented approach decomposes the strain-stress relationship into multiple independent neural networks with only one output respectively, which significantly decreases the complexity of the model. The effectiveness and generalisation of the ML based plasticity model are validated in 2D and 3D using several applications.

7.6.1 Decoupling the stress data by POD

The Proper Orthogonal Decomposition (POD) in combination with machine learning tools, such as Gaussian Processes (XIAO ET AL., 2010) and Long-Short-Term memory network (MOHAN & GAITONDE, 2018), as a reduced order model has been used to surrogate model generation of fluid dynamic systems. Here we introduce a novel combination framework, where POD and FNNs are combined for sequence data preprocessing and prediction. We call this approach Proper Orthogonal Decomposition Feed forward Neural Network (PODFNN). Using POD, the time series vector variables can be represented with a reduced number of modes neglecting higher order modes if the error is acceptable. Thus, by using the POD, the problem will be decoupled into a combination of several different modes.

As time series data, the stress sequence in training data can be rewritten as a snapshot matrix

$$\boldsymbol{O}_\sigma = \begin{bmatrix} \boldsymbol{\sigma}_{op1} & \boldsymbol{\sigma}_{op2} & \dots & \boldsymbol{\sigma}_{opi} & \dots \end{bmatrix} = \begin{bmatrix} \sigma_1^1 & \sigma_1^2 & \dots & \sigma_1^k & \dots \\ \sigma_2^1 & \sigma_2^2 & \dots & \sigma_2^k & \dots \\ \sigma_3^1 & \sigma_3^2 & \dots & \sigma_3^k & \dots \end{bmatrix}_{3\times M}, \tag{7.61}$$

where each column of the matrix is a snapshot and can be written as a vector $\boldsymbol{o}_\sigma^k = \begin{bmatrix} \sigma_1^k & \sigma_2^k & \sigma_3^k \end{bmatrix}^T$. Using POD, any snapshot $\boldsymbol{o}_\sigma^k$ can be represented by a linear combination of the basis

$$\boldsymbol{o}_\sigma^k = \bar{\boldsymbol{\sigma}} + \sum_{i=1}^{m} \alpha_i^k \boldsymbol{\varphi}_i, \tag{7.62}$$

where $\boldsymbol{\varphi}_i = \begin{bmatrix} \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^T$ is the i-th basis vector, α_i^k is the coefficient, m is the number of POD mode and $\bar{\boldsymbol{\sigma}} = \begin{bmatrix} \bar{\sigma}_1 & \bar{\sigma}_2 & \bar{\sigma}_3 \end{bmatrix}^T$ is the mean value of the snapshot matrix. Since $\bar{\boldsymbol{\sigma}}$, $\boldsymbol{\varphi}_i$ are constants, and the bases $\boldsymbol{\varphi}_i$ are independent with each other, the stress sequence can thus be decoupled into independent coefficient sequences.
The components of mean value vector $\bar{\boldsymbol{\sigma}}$ of the snapshot matrix are computed as

$$\bar{\sigma}_1 = \frac{1}{M}\sum_{i=1}^{M} \sigma_1^i, \quad \bar{\sigma}_2 = \frac{1}{M}\sum_{i=1}^{M} \sigma_2^i, \quad \bar{\sigma}_3 = \frac{1}{M}\sum_{i=1}^{M} \sigma_3^i. \tag{7.63}$$

To find the basis vectors and its coefficients, the deviation matrix is firstly computed as

$$\boldsymbol{O}_\sigma^{dev} = \begin{bmatrix} \sigma_1^1 - \bar{\sigma}_1 & \sigma_1^2 - \bar{\sigma}_1 & \dots & \sigma_1^k - \bar{\sigma}_1 & \dots \\ \sigma_2^1 - \bar{\sigma}_2 & \sigma_2^2 - \bar{\sigma}_2 & \dots & \sigma_2^k - \bar{\sigma}_2 & \dots \\ \sigma_3^1 - \bar{\sigma}_3 & \sigma_3^2 - \bar{\sigma}_3 & \dots & \sigma_3^k - \bar{\sigma}_3 & \dots \end{bmatrix}_{3\times M}. \tag{7.64}$$

Then, by applying Singular Value Decomposition (SVD) to the deviation matrix

$$\boldsymbol{O}_\sigma^{dev} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T, \tag{7.65}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are the unitary matrices, $\boldsymbol{S}$ is the diagonal matrix with non-negative real numbers on the diagonal, the basis vectors $\boldsymbol{\varphi}_i$ can be determined from the non-zero columns of matrix $\boldsymbol{U}$

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\varphi}_1 & \boldsymbol{\varphi}_2 & \dots & \boldsymbol{\varphi}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \dots & \boldsymbol{u}_m \end{bmatrix}_{3\times m}, \tag{7.66}$$

where $\boldsymbol{u}_m$ is the m-th non-zero column of matrix $\boldsymbol{U}$ and m is equal to the rank of $\boldsymbol{O}_\sigma^{dev}$.
The coefficients $\boldsymbol{\alpha}^k = \begin{bmatrix} \alpha_1^k & \alpha_2^k & \dots & \alpha_m^k \end{bmatrix}^T$ can be obtained by projecting the snapshot $\boldsymbol{o}_\sigma^k$ on the basis matrix $\boldsymbol{\Phi}$

$$\boldsymbol{\alpha}^k = \boldsymbol{\Phi}^T \boldsymbol{o}_\sigma^k. \tag{7.67}$$

Since the stress components are independent, the deviation matrix $\boldsymbol{O}_\sigma^{dev}$ has the full rank of $m = 3$. In this work, all of the 3 modes are applied in the POD representation of stress

sequences. The stress sequence in equation (7.32) can thus be represented by 3 independent coefficient sequences

$$\boldsymbol{O}_\sigma = \begin{bmatrix} \sigma_1^1 & \sigma_1^2 & ... & \sigma_1^k & ... \\ \sigma_2^1 & \sigma_2^2 & ... & \sigma_2^k & ... \\ \sigma_3^1 & \sigma_3^2 & ... & \sigma_3^k & ... \end{bmatrix}_{3\times M} \xrightarrow{POD} \begin{bmatrix} \alpha_1^1 & \alpha_1^2 & ... & \alpha_1^k & ... \end{bmatrix}_{1\times M} \begin{bmatrix} \alpha_2^1 & \alpha_2^2 & ... & \alpha_2^k & ... \end{bmatrix}_{1\times M} \begin{bmatrix} \alpha_3^1 & \alpha_3^2 & ... & \alpha_3^k & ... \end{bmatrix}_{1\times M} . \tag{7.68}$$

Therefore, the training data composed by the strain-stress sequences in equation (7.32) for plasticity model are transformed into training data composed by strain-coefficient sequences and can be written as

$$\begin{bmatrix} \boldsymbol{\varepsilon}_{op1} & \boldsymbol{\varepsilon}_{op2} & ... & \boldsymbol{\varepsilon}_{opi} & ... \end{bmatrix}_{6\times M} \Leftrightarrow \begin{bmatrix} \alpha_1^1 & \alpha_1^2 & ... & \alpha_1^k & ... \end{bmatrix}_{1\times M} \begin{bmatrix} \alpha_2^1 & \alpha_2^2 & ... & \alpha_2^k & ... \end{bmatrix}_{1\times M} \begin{bmatrix} \alpha_3^1 & \alpha_3^2 & ... & \alpha_3^k & ... \end{bmatrix}_{1\times M}, \tag{7.69}$$

where the three coefficient sequences are independent from each other.

7.6.2 Prediction of coefficients using FNN

Once the training data are prepared, FNNs are applied to learn the mapping between the strain sequence and the coefficient sequences in equation (7.69). Since the coefficients in the POD representation are independent, each coefficient can be predicted by one FNN, as shown in Fig. 7.22, where the original strain-stress mapping approximated by one complex neural network is decoupled into three independent strain-coefficient mappings approximated by simpler neural networks.

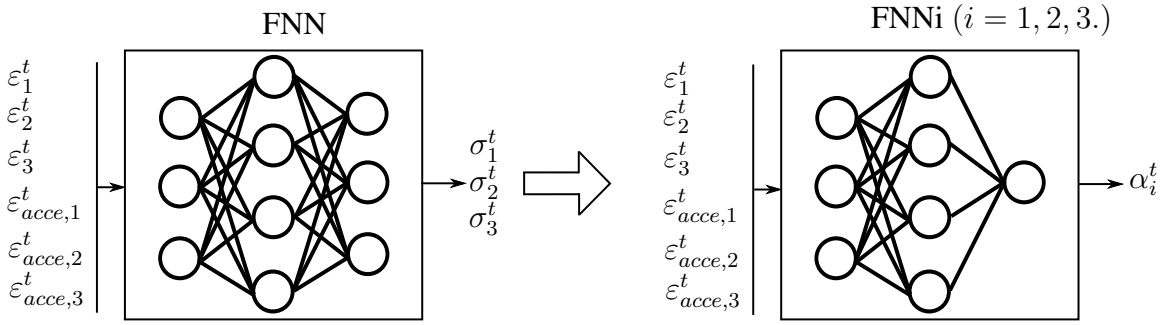


Figure 7.22. Decoupling the strain-stress mapping (left) into independent strain-coefficient mappings (right) by POD.

After training, the coefficient $\alpha_{NN,i}^t$ at time t will be predicted by the feed forward neural network FNN_i as

$$\alpha_{NN,i}^t = FNN_i(\boldsymbol{\varepsilon}^t, \boldsymbol{\varepsilon}_{acc}^t, \boldsymbol{w}, \boldsymbol{b}_s), \quad (i = 1, 2, 3), \tag{7.70}$$

where i indicates the number of the coefficient, $\boldsymbol{\varepsilon}^t = \begin{bmatrix} \varepsilon_1^t & \varepsilon_2^t & \varepsilon_3^t \end{bmatrix}^T$ is the current strain, $\boldsymbol{\varepsilon}_{acc}^t = \begin{bmatrix} \varepsilon_{acc,1}^t & \varepsilon_{acc,2}^t & \varepsilon_{acc,3}^t \end{bmatrix}^T$ is the accumulated absolute strain, $\boldsymbol{w}$ is the weight matrix and $\boldsymbol{b}_s$ is the bias of the neural network.

7.6.3 PODFNN based plasticity model

Once the coefficient $\boldsymbol{\alpha}^t_{NN}$ is predicted by FNNs as described in equation (7.70), the principle stress can be recovered from the POD representation as

$$\tilde{\boldsymbol{\sigma}}^t_{PODFNN} = \bar{\boldsymbol{\sigma}} + \sum_{i=1}^{3} \alpha^t_{NN,i} \boldsymbol{\varphi}_i. \tag{7.71}$$

Finally, the Cauchy stress is obtained by transforming the principle stress into the general space

$$\boldsymbol{\sigma}^t_{PODFNN} = \boldsymbol{Q} \tilde{\boldsymbol{\sigma}}^t_{PODFNN} \boldsymbol{Q}^T. \tag{7.72}$$

The formulation of the ML based plasticity model defines a constitutive function in the following format

$$\boldsymbol{\sigma}^t_{PODFNN} = PODFNN(\boldsymbol{\varepsilon}^t, \boldsymbol{\varepsilon}^t_{acc}, \boldsymbol{w}, \boldsymbol{b}_s), \tag{7.73}$$

where $\boldsymbol{\varepsilon}^t$ is the current total strain, $\boldsymbol{\varepsilon}^t_{acc}$ is the history variable. To apply the ML based plasticity model in the finite element analysis, the residual vector and tangent matrix have to be derived. The residual vector for the static problem is given by

$$\boldsymbol{R}(\boldsymbol{u}) = \boldsymbol{f} - \int_\Omega \boldsymbol{B}^T \boldsymbol{\sigma}^t_{PODNN} d\Omega, \tag{7.74}$$

$$\boldsymbol{f} = \int_\Omega N \rho \hat{\boldsymbol{b}} dv - \int_{\partial\Omega} N \hat{\boldsymbol{t}} da. \tag{7.75}$$

The tangent matrix is computed by taking the derivative of residual in terms of displacement

$$\boldsymbol{K}_T = \frac{\partial \boldsymbol{R}(\boldsymbol{u})}{\partial \boldsymbol{u}}. \tag{7.76}$$

To derive the tangent matrix for the PODFNN based material model by the chain rule, the automatic symbolic differentiation tool AceGen, see KORELC & WRIGGERS (2016), is applied again.

7.6.4 Testing the PODFNN based plasticity model in FEM

In the following, the performance of the developed PODFNN based plasticity model is evaluated in finite element applications.

Uniaxial tension and compression

To test the performance of the PODFNN based plasticity model, the 1D uniaxial tension and compression test are conducted firstly. The von Mises plasticity with the linear isotropic hardening is applied as the target model. The material parameters of the plasticity model are set as: Young's modulus $E = 700N/mm^2$, yield stress $\sigma_y = 100MPa$ and isotropic hardening parameter $H_{iso} = 10MPa$. To prepare the training data, 11 sets of strain-stress

sequence data are collected from the target model with strain increments being increased linearly from 0.02 to 0.03. The stress sequences are then transformed into the coefficient sequence by the POD.
The feed forward neural network with the architecture of (2-20-20-1) is applied to predict the coefficient, where the input layer containing 2 neurons is connected with two hidden layers containing 20 neurons each. The output layer contains one neuron. The input of the neural network is the total strain together with the accumulated absolute strain, and the output of the neural network is the coefficient transformed from the stress sequence. The Levenberg-Marquardt algorithm is applied as the optimizer in training. The weights are initialized by the Nguyen-Widrow method. After 4000 epochs, the mean squared error decreased to 0.0393 which costs the training time of $2m24s$.
The testing strain sequence is generated by setting the strain increment as 0.15 so that it is different from the training data. The stress computed from the PODFNN based plasticity model is compared with the stress from the target plasticity model, as shown in Fig. 7.23. It can be seen that the predicted stress follows the exact solution well, which shows the effectiveness of the proposed PODFNN approach for plasticity. This test also shows that the accumulated absolute total strain as a history variable captures the loading history for the cyclic loading condition.

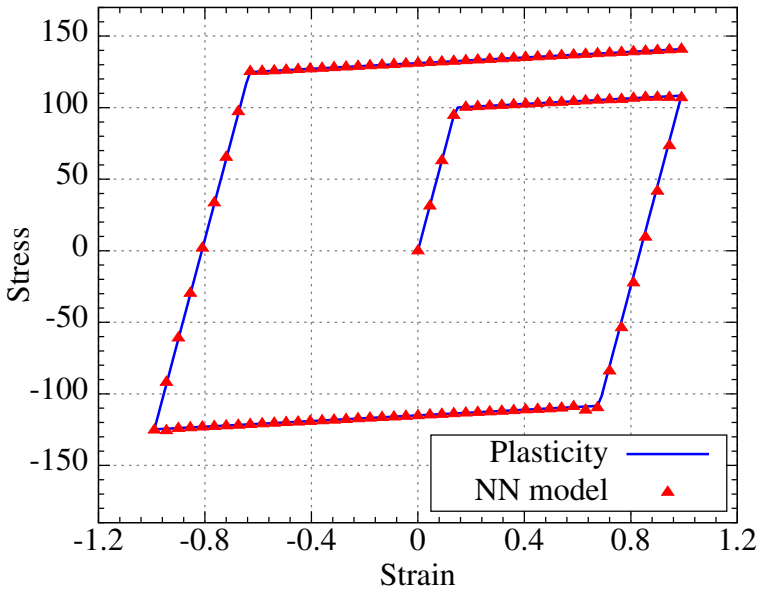


Figure 7.23. Uniaxial tension and compression.

Applications in 2D finite element analysis

To evaluate the PODFNN based plasticity model, benchmark tests in 2D are presented. The von Mises plasticity with an exponential isotropic hardening law $\sigma_y = y_0 + y_0(0.00002 + \gamma)^{0.3}$ is set as the target model. The material parameters are set as: Young's modulus $E =$

$1N/mm^2$, Poison's ratio $\nu = 0.33$, and the initial yield stress $y_0 = 0.05MPa$. To collect the training data, 122 loading-unloading paths evenly distributed within the circles ($r_1 = 0.1, r_2 = 0.075$) in Fig. 7.13 are selected to conduct the biaxial tests, where 61 values are assigned to the angle ϕ. Then the collected stress sequence data are transformed into the coefficient sequences using POD.

Since there are two coefficients referring to the two principle stress components in the 2D case, two neural networks will be required to predict the coefficients. In this part, the same network architecture (4-20-20-1) is applied for the two FNNs, where the input layer containing 4 neurons is connected with two hidden layers containing 20 neurons each. The output layer contains always one neuron. The total strain together with the accumulated absolute strain are applied as the input of the neural network. The output of the neural network is the coefficient transformed from the stress sequences. The Levenberg-Marquardt algorithm is applied as the optimizer as well. The training progress is terminated when the gradient of error is less than 10^{-7}, where the mean squared error is decreased to 7.96×10^{-9} for the first FNN and 6.35×10^{-9} for the second FNN.

After the training process, the weights and biases of the neural network are output as the constant model parameters, by which the Cauchy stress is recovered according to the POD formulation. The tangent matrix and the residual vector are derived using the symbolic differentiation tool AceGen again.

Firstly, the 2D PODFNN based plasticity model is tested by the Cook's membrane problem. The beam is clamped at the left end and loaded at the right end by a constant distributed vertical load $q_0 = 0.03Mpa$, as depicted in Fig. 7.8. In the unloading process, the direction of the vertical load is changed to be negative. The geometric domain of the structure is discretized by 40 quadratic 9-node quadrilateral elements leading to 189 nodes.

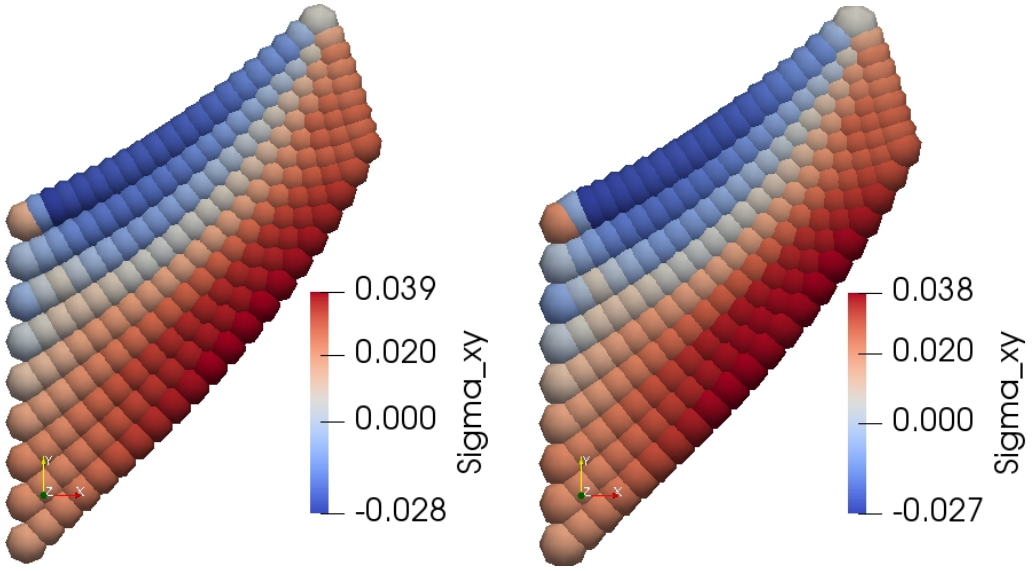
Before unloading, the stress field σ_{xy} in the final deformation state of the beam using the proposed ML based plasticity model is compared with the target model, which is depicted in Fig. 7.24. It can be observed that the stress distributions of the two models are very close.

The load displacement curve of the upper node $(48, 60)$ at the right end of the cantilever beam is plotted in Fig. 7.25. The displacements in Fig. 7.25 and in the remaining figures of this chapter are given in millimeter. The figure shows that the machine learning based plasticity model captures the loading and unloading behaviour very well.

The second example to test the PODFNN based plasticity model is a punch test as shown in Fig. 7.26, where the vertical displacement boundary condition $(u_0 = 0.07mm)$ is imposed on the top of the block and the bottom of the block is only fixed in the vertical direction. In the unloading process, the direction of the vertical displacement boundary is changed to positive. The block is discretized with 100 quadratic 9-node quadrilateral elements leading to 441 nodes.

Before unloading, the stress field σ_{xy} in the final deformation state of the block using the proposed ML based plasticity model is compared with the target model, which is depicted in Fig. 7.27. It can be observed that the stress distributions of the two models are very close.

The load displacement curve of the upper node $(0, 1)$ at the left end of the block is plotted in Fig. 7.28, where it can be seen that the PODFNN based model follows the plasticity model well both in loading and unloading.



(a) With plasticity model (b) With PODFNN model

Figure 7.24. Final deformation state of the 2D cook's membrane.

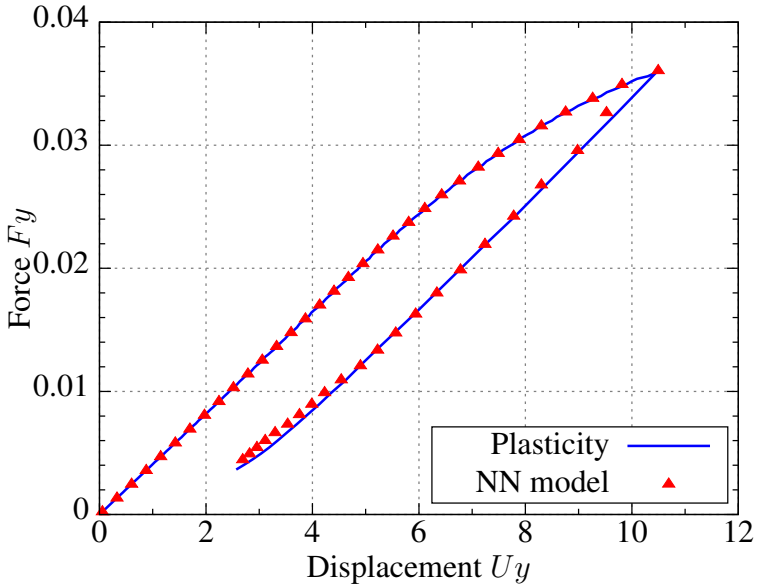


Figure 7.25. Load deflection curve of the 2D cook's membrane.

Applications in 3D finite element analysis

In this section, the PODFNN based plasticity model is extended to 3D applications. To generate the training data, one hexahedron finite element is applied to different loading situations as described in Fig. 7.14. The von Mises plasticity with an exponential isotropic hardening law $\sigma_y = y_0 + y_0(0.00002 + \gamma)^{0.1}$ is set as the target model. The material parameters are set

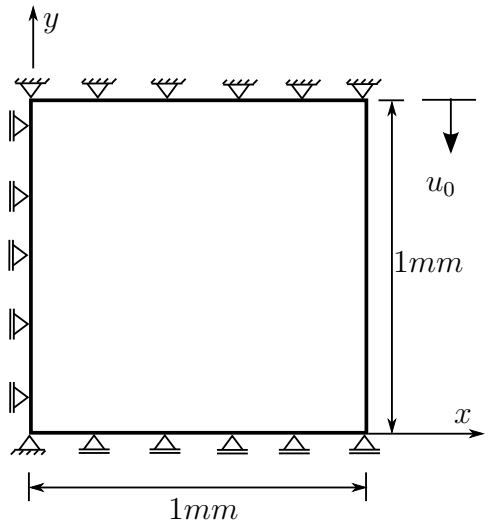
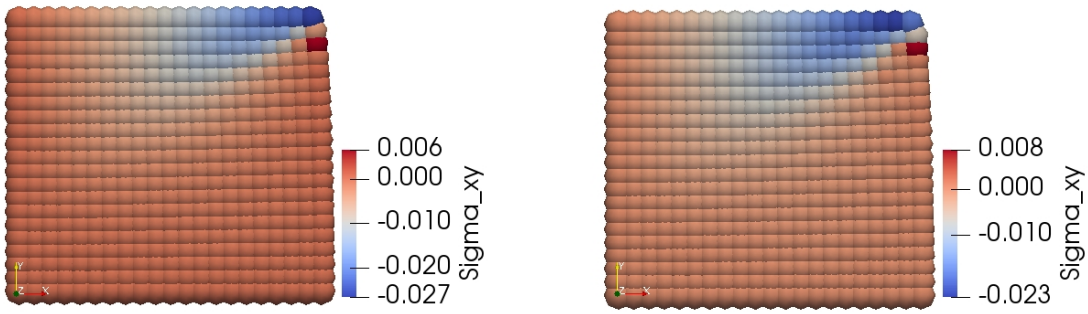


Figure 7.26. 2D punch problem.



(a) With plasticity model

(b) With PODFNN model

Figure 7.27. Final deformation state of the 2D block.

as: Young's modulus $E = 10N/mm^2$, Poison's ratio $\nu = 0.33$, and the initial yield stress $y_0 = 0.3MPa$. During the training data preparation, strain-stress sequences along 8100 loading paths are generated based on the sphere $(r = 0.02)$ in Fig. 7.15, where 90 values are assigned to the angles ϕ and θ, respectively. Since the huge amount of data have to be collected for unloading in 3D, only the loading data are collected here and the unloading is not considered in this part.

In the three-dimensional case, three FNNs are required to predict the coefficients, which are corresponding to the principle Cauchy stress components. The same network architecture (6-16-16-1) is employed for all FNNs, where three total strain components together with three accumulated absolute strains are applied as the input of the networks. The output of

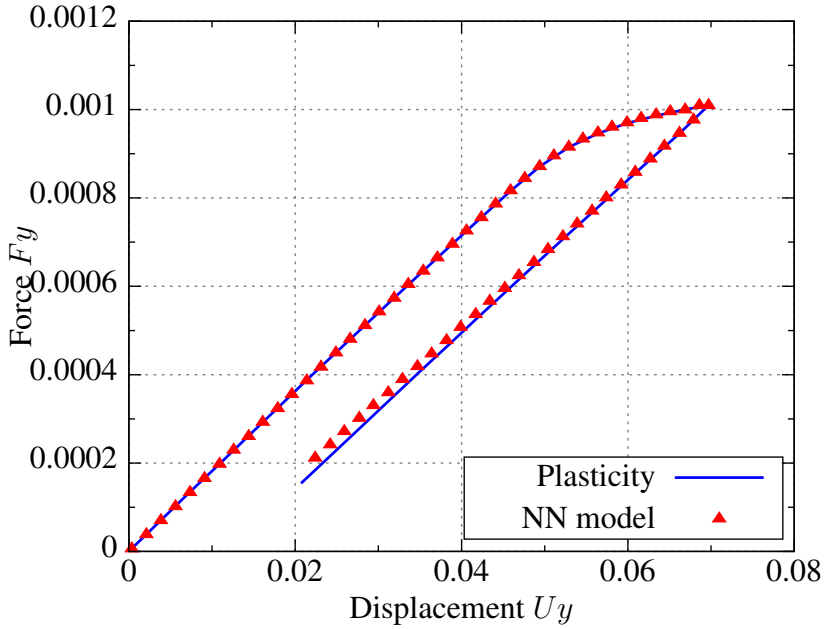


Figure 7.28. Load deflection curve of the 2D punch problem.

the neural network is the coefficient.

The weights are initialized by the Nguyen-Widrow method. During training, the Levenberg-Marquardt algorithm is applied as the optimizer, where the training progress is terminated when the gradient of the global error is less than 10^{-7}. The mean squared errors are decreased to 5.70×10^{-8}, 1.13×10^{-9} and 6.78×10^{-10} for the first, second and third FNN, respectively. The training process costs time of $1h20m21s$, $1h18m46s$ and $52m47s$ for the first, second and third FNN, respectively.

To evaluate the performance of the POD representation, the performances of training strain-stress model with one FNN(6-16-16-3) and training three strain-coefficient models with three FNNs(6-16-16-1) are compared. Without POD, only one FNN is required to approximate the mapping, where the output includes 3 stress components. With POD, three independent FNNs will be applied, where the output of FNN includes only one POD coefficient. The training performances within 2000 epochs are shown in Fig. 7.29. It can be seen that the average of the mean squared errors of the three FNNs is smaller than that without POD. Additionally, training a FNN with the architecture of (6-16-16-3) costs computation time of $4h22m43s$ whereas the average training time of the three FNNs (6-16-16-1) is $1h26m27s$. It can be observed that the POD approach leads to less training time and better training performance.

The first example to test the 3D PODFNN based plasticity model is the necking of a bar as shown in Fig. 7.30, where the left end of the bar is fixed and the displacement boundary $u_0 = 0.05mm$ is imposed at the right end along its axial direction. An artificial imperfection is set in the center of the bar to trigger the necking, where the radius at the center is chosen to be $R_c = 0.98R$. The bar is discretized with 200 quadratic 27-node elements leading to 2193 nodes.

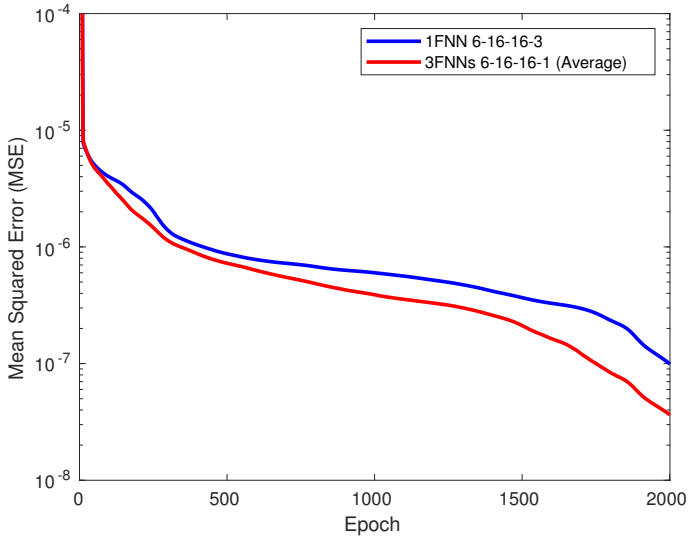


Figure 7.29. MSE of training FNN(6-16-16-3) and the average MSE of training 3 FNNs(6-16-16-1).

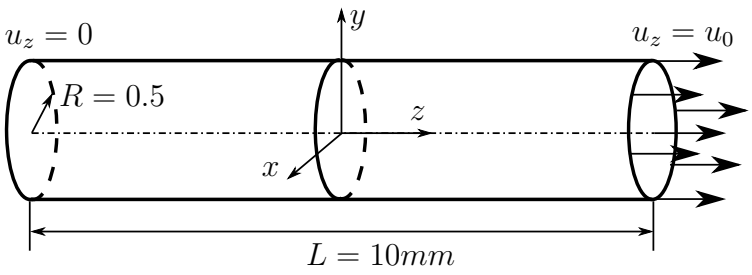


Figure 7.30. Geometry and boundary conditions of the bar.

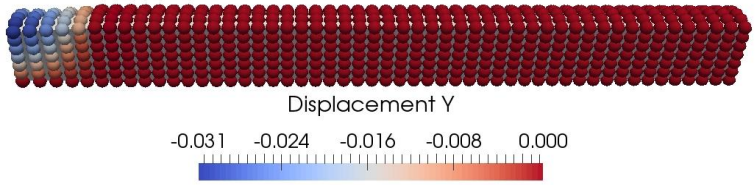
Fig. 7.31 shows the final deformation of the bar after tension, where only one quarter of the bar is computed due to the symmetry. It can be observed that the amounts of the necking computed by the two models are close to each other.

The load displacement curve of the bar under the uniaxial tension is plotted in Fig. 7.32, where the PODFNN based model follows the plasticity model quite well.

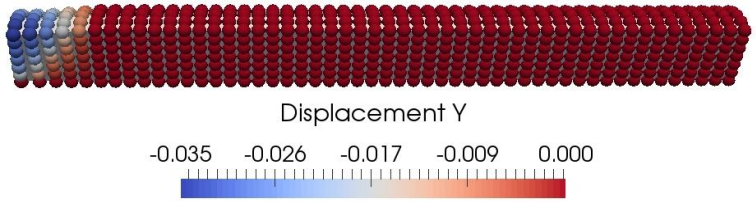
The second example is the punch test, where the vertical displacement boundary condition $(u_0 = 0.15mm)$ is imposed on the top of the block and the bottom of the block is only fixed in the vertical direction, as shown in Fig. 7.33. The block is discretized with 100 quadratic 27-node elements leading to 441 nodes.

Fig. 7.34 shows the final deformation of the block after compression. It can be observed that the displacements in the horizontal direction computed by the two models are close to each other.

The load displacement curve of the block under compression is plotted in Fig. 7.35. It can



(a) With plasticity model



(b) With PODFNN model

Figure 7.31. Final deformation state of cylindrical bar.

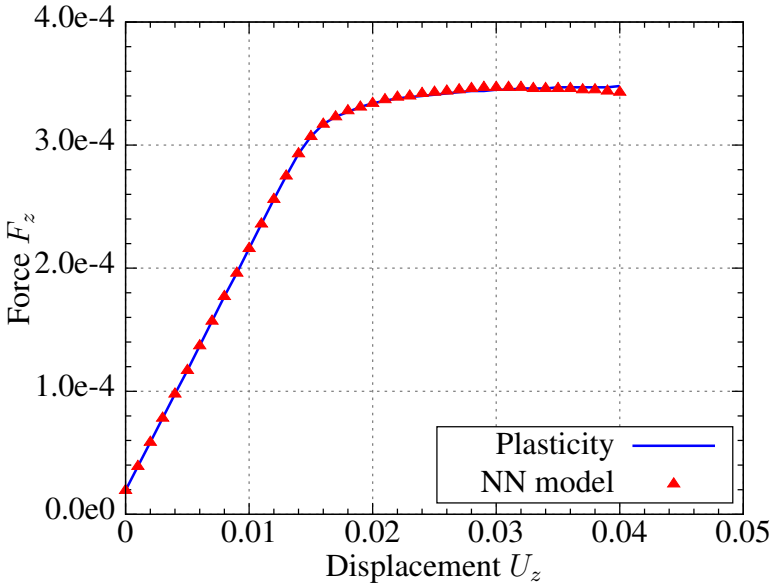


Figure 7.32. Load deflection curve of the bar.

be seen that the PODFNN based model follows the plasticity model quite well.

The last example for the 3D neural network based model is the Cook's membrane problem. The 3D beam is clamped at the left end and loaded at the right end by a constant distributed vertical load $q_0 = 0.03MPa$, as depicted in Fig. 7.36. By using the quadratic finite element with 27 nodes, the beam is discretized using 1080 elements leading to 10309 nodes.

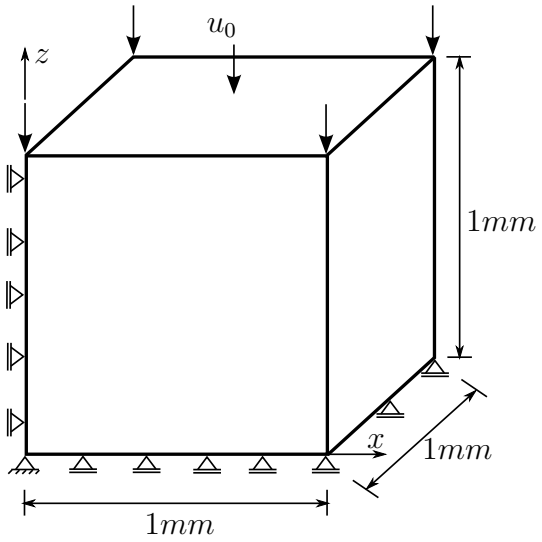
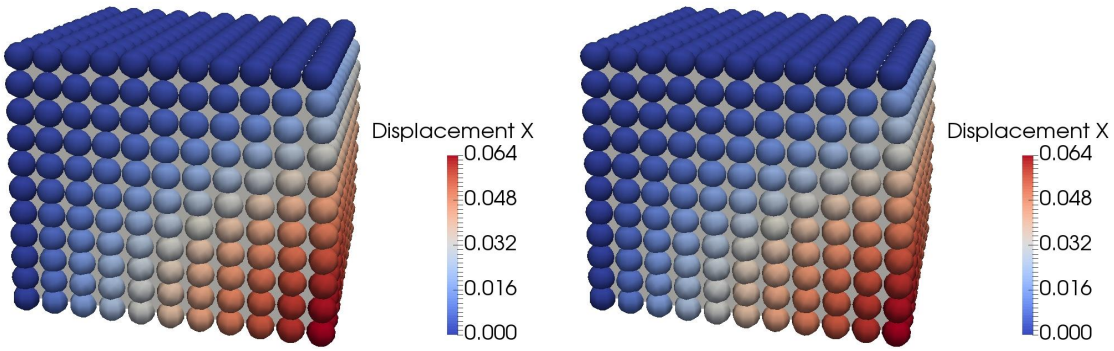


Figure 7.33. 3D punch problem.



(a) With plasticity model

(b) With PODFNN model

Figure 7.34. Final deformation state of the 3D block.

Fig. 7.37 shows the final deformation of the Cook's membrane. It can be observed that the displacements in the vertical direction computed by the two models are almost identical. The load displacement curve of the upper node (48,0,60) is plotted in Fig. 7.38. It can be seen that the PODFNN based model follows the plasticity model quite well.

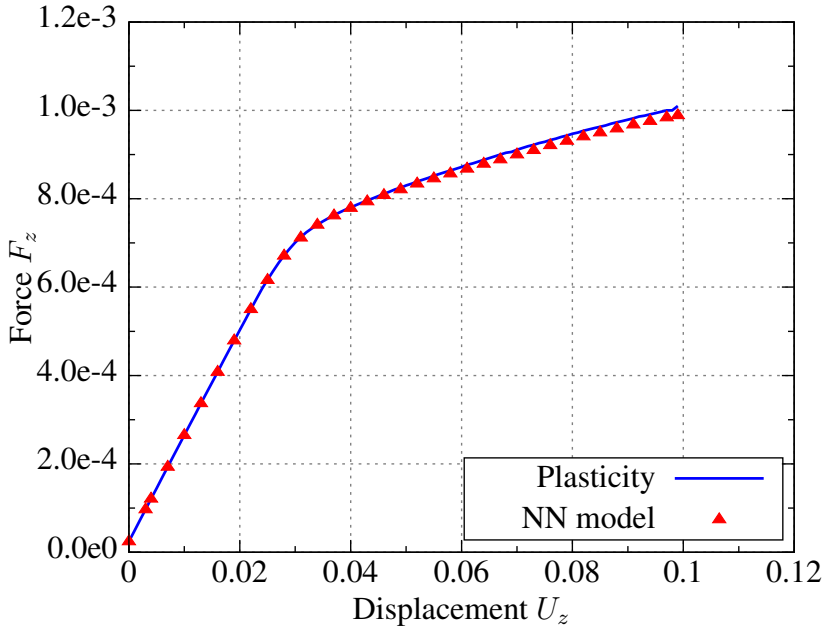


Figure 7.35. Load deflection curve of the 3D block.

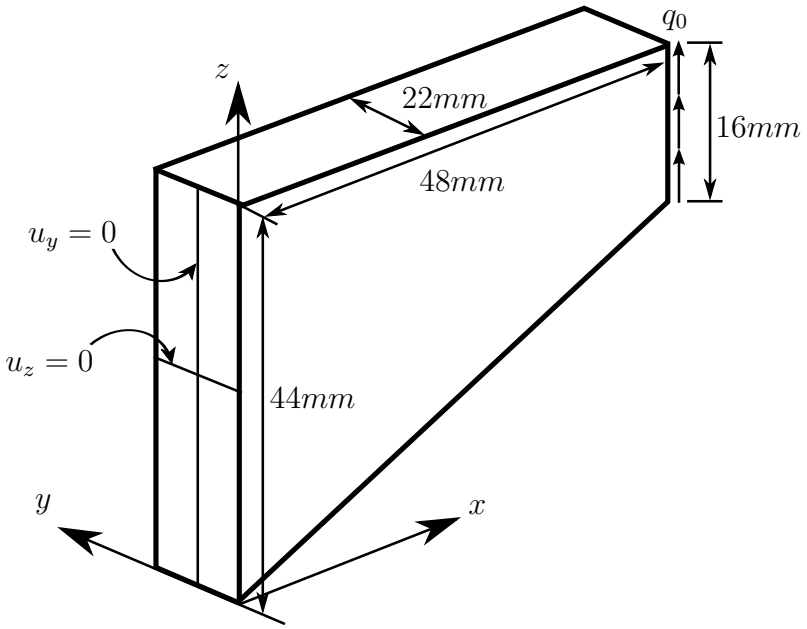
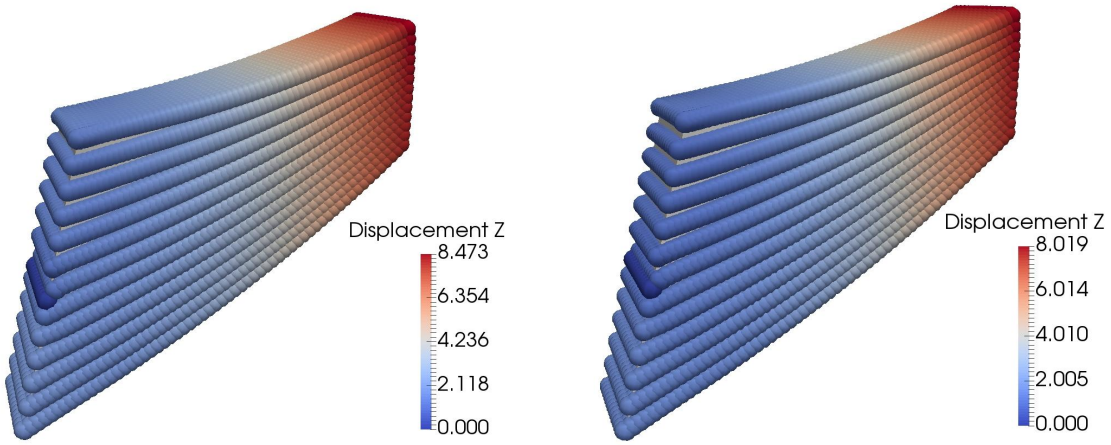


Figure 7.36. 3D cook's membrane problem.



(a) With plasticity model (b) With PODFNN model

Figure 7.37. Final deformation state of the 3D cook's membrane.

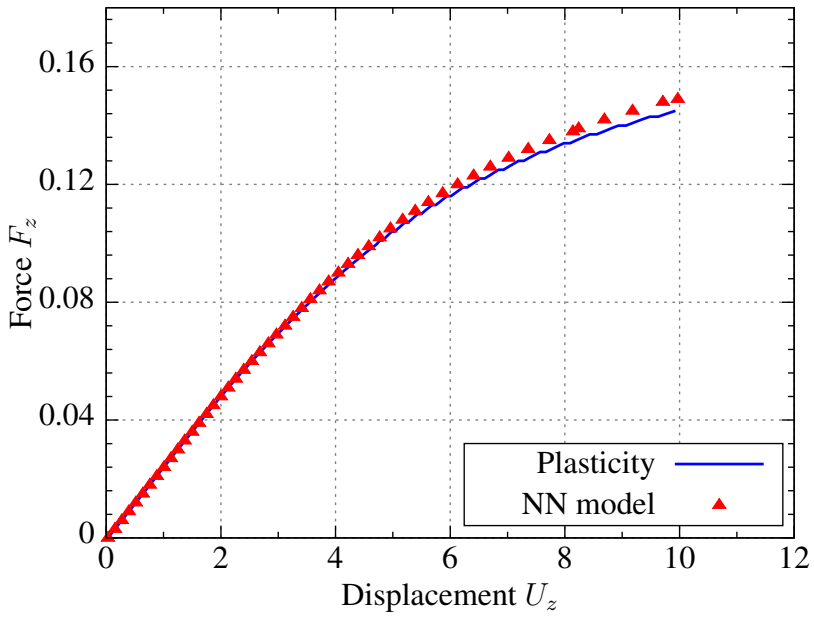


Figure 7.38. Load deflection curve of the 3D cook's membrane.

Chapter 8

Metal Cutting Simulation Using ML based Material Model

To provide a data-integrated framework for the simulation of the metal cutting process, the classical material model is replaced by the data-driven material model by the use of the Machine Learning (ML) technology. Since the chip formation has to be described with the finite plasticity, the ML based plasticity model is extended to the finite strain. In this chapter, the ML based finite plasticity is formulated by the use of the Proper Orthogonal Decomposition Feed forward Neural Network (PODFNN) approach introduced in Chapter 7. Finally, the chip formation of the metal cutting process is simulated by using the ML based finite plasticity model.

8.1 ML based finite plasticity

According to the finite plasticity formulation described in Chapter 5, the utilisation of the Henky strain leads to the formulation of the plasticity in the principle space. The principle stress is computed from the logarithmic strain which is obtained by the spectral decomposition of the left Cauchy Green tensor. The Cauchy stress is computed by transforming the principle stress into the general space. Thus, the neural network based finite plasticity model presented in Chapter 7 for the small strain situation can be naturally extended to the finite strain situation with the logarithmic strain as input and the principle stress as the output.
Within an updated Lagrangian formulation, the left Cauchy Green tensor at time t is computed by

$$\boldsymbol{b}^t = \Delta \boldsymbol{F}^t \boldsymbol{b}^{t-1} (\Delta \boldsymbol{F}^t)^T. \tag{8.1}$$

By applying the spectral decomposition to the left Cauchy Green tensor

$$\boldsymbol{b}^t = \boldsymbol{Q} \boldsymbol{\Lambda}^t \boldsymbol{Q}^T, \tag{8.2}$$

the logarithmic strain vector, also known as Henky strain, can be computed as

$$\boldsymbol{\varepsilon}_H^t = \frac{1}{2} \log(\boldsymbol{\Lambda}^t). \tag{8.3}$$

The components of the logarithmic strain will be applied as the input of the ML based model. To distinguish the loading history, the accumulated absolute strain computed from the logarithmic strain will be applied as the history variable as well. By the use of the Proper Orthogonal Decomposition (POD), the principle component of Cauchy stress is represented by the basis and the coefficients

$$\tilde{\boldsymbol{\sigma}}^t_{PODFNN} = \bar{\boldsymbol{\sigma}} + \sum_{i=1}^{3} \alpha^t_{NN,i} \boldsymbol{\varphi}_i. \tag{8.4}$$

where the coefficients $\alpha^t_{NN,i}, (i = 1, 2, 3)$ are predicted by three independent feed forward neural networks respectively.
Finally, the Cauchy stress is obtained by transforming the principle stress into the general space

$$\boldsymbol{\sigma}^t_{PODFNN} = \boldsymbol{Q}\tilde{\boldsymbol{\sigma}}^t_{PODFNN} Q^T. \tag{8.5}$$

Thus, the formulation of the PODFNN based finite plasticity model defines a constitutive function in the following format

$$\boldsymbol{\sigma}^t_{PODFNN} = PODFNN(\boldsymbol{b}^t, \boldsymbol{\varepsilon}^t_{acc}, \boldsymbol{w}, \boldsymbol{b}_s), \tag{8.6}$$

where $\boldsymbol{b}^t$ is the current left Cauchy Green tensor, $\boldsymbol{\varepsilon}^t_{acc}$ is the accumulated absolute logarithmic strain, $\boldsymbol{w}$ contains the weights of the neural networks and $\boldsymbol{b}_s$ is the bias.
The training data sets including the logarithmic strain as input and the POD coefficients as output are collected from the tension and compression tests in Fig. 7.14 and Fig. 7.15 in section 7.3.2, where the coefficients are obtained by POD decomposition of the stress sequence data.

8.2 Chip formation with PODFNN based material model

The PODFNN based finite plasticity model is applied to the metal cutting simulation in this part. The orthogonal cutting of a rectangular block is investigated, where the same geometry of workpiece as shown in Fig. 6.1 and the same geometry of cutting tool as shown in Fig. 6.2 are applied. For simplicity, the von Mises plasticity with a linear isotropic hardening law is set as the target model and thus applied to the metal cutting simulation. The material parameters are set as: Young's modulus $E = 114GPa$, Poison's ratio $\nu = 0.33$, the initial yield stress $y_0 = 782MPa$ and the isotropic hardening modulus $H_{iso} = 498MPa$.
To train the neural networks, the training data is collected according to the strategy introduced in section 7.4. Since the plain strain assumption is made in the simulation, the training data is collected within the x-y plane as same as the 2D case, where 200 loading paths evenly distributed within the circle in Fig. 7.13 are selected to conduct the biaxial tests. Then the collected stress sequence data is transformed into the coefficient sequences using POD.
Two feed forward neural networks are applied to predict the two POD coefficients, in which the same network architecture (4-20-20-1) is applied. The input layer containing 4 neurons is connected with two hidden layers containing 20 neurons each. The output layer contains

always one neuron. The logarithmic strain together with the accumulated absolute strain are applied as the input of the neural network. The output of the neural network is the coefficient transformed from the stress sequences. The Levenberg-Marquardt algorithm is applied as the optimizer as well. The training progress is terminated after the stop criteria of 2000 epochs is reached, where the mean squared error is decreased to 10^{-8}. After the training process, the weights and biases of the neural network are output as the constant model parameters, by which the Cauchy stress is recovered according to the POD formulation.
Apart from the plasticity model, the fracture model is required to capture the chip separation from the workpiece in metal cutting. Since the plastic variable is not available in the machine learning based model, a simple fracture criteria with total strain is applied in this part, i.e.

$$\varepsilon_{H,x}^2 + \varepsilon_{H,y}^2 > 0.5, \tag{8.7}$$

where $\varepsilon_{H,x}$ and $\varepsilon_{H,y}$ are the logarithmic strain components.
The chip formation process simulated in OTM using the proposed PODFNN based finite plasticity model is depicted in Fig. 8.1. It can be seen the ML based plasticity model captures the deformation of the chip well. When the fracture criteria is reached at the chip root, the first chip start to separate from the workpiece. However, the fracture in cutting zone is overestimated, which leads to the total separation of chip form the workpiece. Thus, a fracture model is required to match the machine learning based constitutive model, which suggests a future work.

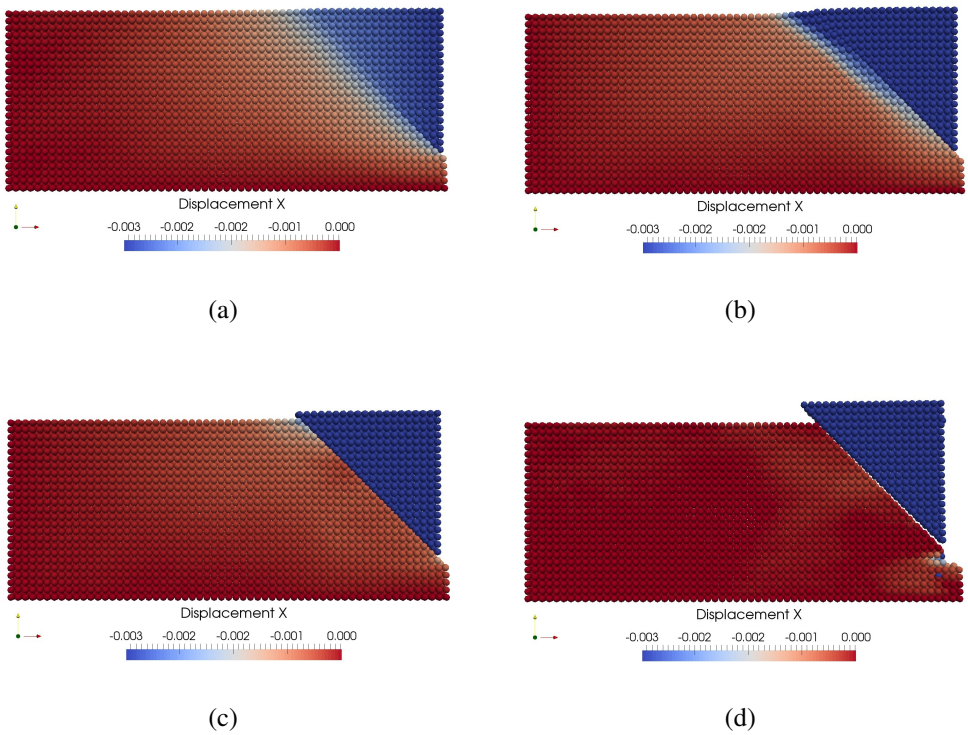


Figure 8.1. Metal cutting simulation using the machine learning based plasticity.

Chapter 9

Conclusion and Outlook

In this work, the chip formation of the metal cutting process is simulated by the use of the meshfree method and two classes of material models, the classical phenomenological model and the data-driven material model. By means of the stabilized OTM method, both the serrated morphology on the chip upper surface and the material separation at chip root are treated as the ductile fracture. This enables a realistic modelling of the serrated chip formation in the metal cutting process. As a phenomenological material model, the Johnson-Cook model together with a complete condition for stress triaxiality is applied to capture the chip formation process. Finally, a data-driven material modelling approach is proposed to replace the classical material model, where the machine learning technologies including the Feed forward Neural Network (FNN), the Recurrent Neural Network (RNN) and the Proper Orthogonal Decomposition (POD) are applied. Based on the data collection strategy for plasticity, the RNN based plasticity model and the PODFNN based plasticity model are developed.

By using the Johnson-Cook material model and OTM method, both the chip separation from the workpiece and the fracture on the chip upper surface are successfully captured by the improved fracture model together with a material point erosion approach. The stress triaxiality is proven to be crucial for the prediction of the ductile fracture locations in the chip formation process. In this study, it can be seen that the Johnson-Cook fracture model mostly used in modelling of the chip formation process is unable to accurately capture the fracture within the shear band. However, using the restriction of positive stress triaxiality in the fracture model, the unphysical over-fracture within the shear band can be circumvented. The calculated chip morphology allows more accurate measurements of the chip size, i.e. chip spacing, peak and valley.

Additionally, the simulation results show that the thermal softening is under-predicted by the Johnson-Cook flow stress model with the normal range of the Taylor-Quinney coefficient. The continuous chip is generated without a shear band in this situation. By setting a larger Taylor-Quinney coefficient, which is physically incorrect, the serrated chip with shear bands is generated with the higher temperature in the cutting zones. This study also indicates that the shear band formation can be attributed to the thermal softening effect.

In the simulation results, the influence of the cutting speed on the chip spacing and the shear band angle is also captured, which is known from experiments. The calculated chip spacing has a good convergence with the increase of the amount of material points. The chip size and

the cutting force variations calculated in this work are in line with the experimental results. The good agreements between the simulated chip formation process and the experimental two-stage chip formation process illustrate that the OTM framework combined with the improved Johnson-Cook fracture model is robust and accurate for chip formation modelling.
To replace the classical material model, in this work a data-driven material modelling approach based on machine learning technologies, including FNN and RNN, is proposed for both hyperelasticity and plasticity. The machine learning based model is first trained offline with the collected strain-stress data and then applied online in numerical simulations. For the hyperelasticity describing a one to one mapping relationship between strain and stress, the FNN shows proficient performances for capturing the strain-stress mapping without additional treatment of the input strain data.
However, history variables as additional inputs are required to distinguish the loading history in case of plasticity. Thus, the accumulated absolute strain is proposed to be the history variable, which captures the loading history well without the requirement for additional data collection. The strain-stress sequence data for the plasticity model are collected from different loading-unloading paths based on the concept of sequence for plasticity. The strategy of data collection from multi-axial tests serves as a guidance for conducting experiments.
In the RNN based plasticity model, the strain components act as the input and the stress components act as the output of the model, where no additional history variable is required because of its built-in memory function. In the PODFNN based plasticity model, the accumulated absolute strain is applied as the history variable in input. At the same time, the multi-dimensional stress sequence in output data is decoupled into independent one dimensional coefficient sequences. In this case, the neural network with multiple outputs to predict the stress components is replaced by multiple independent neural networks in which each possesses a one dimensional output. This leads to less training time and better training performance.
The automatic symbolic differentiation tool AceGen provides a very convenient way to derive the tangent matrix for the machine learning based material model, which leads to its efficient application in the finite element method. The generalisation and accuracy of the presented model as well as the data generation strategy have been verified by finite element applications both in 2D and 3D. Finally, the feasibility of applying the machine learning based model to the metal cutting simulation is demonstrated.
It has been shown that the RNN is able to capture the history dependent plastic behaviour. However, the accuracy of the RNN based model is not enough for applications. The PODFNN based model with the proposed history variable works well for capturing the plastic behaviour and is more accurate that the RNN based model. The thermodynamic consistence of the PODFNN based plasticity model has been verified by benchmark tests in finite element applications as well as the metal cutting simulation in OTM.
The machine learning based material model can be further improved, such as integrating the effects of damage and fracture. By collecting the strain-stress data from the whole range of deformation until fracture, the model can be trained to capture the deformation, softening as well as fracture behaviour at the same time. Thus, a more advanced material model can be developed to completely capture the serrated chip formation in metal cutting.

Bibliography

ALI U., MUHAMMAD W., BRAHME A., SKIBA O. ET AL. Application of artificial neural networks in micromechanics for polycrystalline metals. *International Journal of Plasticity*, 120 (2019): 205–219.

ARRAZOLA P., ÖZEL T., UMBRELLO D., DAVIES M. ET AL. Recent advances in modelling of metal machining processes. *CIRP Annals-Manufacturing Technology*, 62 (2013) (2): 695–718.

ARROYO M. & ORTIZ M. Local maximum-entropy approximation schemes: a seamless bridge between finite elements and meshfree methods. *International Journal for Numerical Methods in Engineering*, 65 (2006) (13): 2167–2202.

AYENSA-JIMÉNEZ J., DOWEIDAR M.H., SANZ-HERRERA J.A. & DOBLARÉ M. A new reliability-based data-driven approach for noisy experimental data with physical constraints. *Computer Methods in Applied Mechanics and Engineering*, 328 (2018): 752–774.

AYENSA-JIMÉNEZ J., DOWEIDAR M.H., SANZ-HERRERA J.A. & DOBLARÉ M. An unsupervised data completion method for physically-based data-driven models. *Computer Methods in Applied Mechanics and Engineering*, 344 (2019): 120 – 143.

BÄKER M., RÖSLER J. & SIEMERS C. A finite element model of high speed metal cutting with adiabatic shearing. *Computers & Structures*, 80 (2002) (5-6): 495–513.

CALAMAZ M., COUPARD D. & GIROT F. A new material model for 2d numerical simulation of serrated chip formation when machining titanium alloy ti–6al–4v. *International Journal of Machine Tools and Manufacture*, 48 (2008) (3-4): 275–288.

CAO T.S. Models for ductile damage and fracture prediction in cold bulk metal forming processes: a review. *International Journal of Material Forming*, 10 (2017) (2): 139–171.

CORONA E. & ORIENT G.E. An evaluation of the Johnson-Cook model to simulate puncture of 7075 aluminum plates. Technical report, Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States) (2014).

CORONA E. & REEDLUNN B. A review of macroscopic ductile failure criteria. *Sandia Report No. SAND2013-7989, Sandia National Laboratories, Livermore, CA*, (2013).

DUCOBU F., RIVIÈRE-LORPHÈVRE E. & FILIPPI E. Numerical contribution to the comprehension of saw-toothed ti6al4v chip formation in orthogonal cutting. *International Journal of Mechanical Sciences*, 81 (2014): 77–87.

EBERHARD P. & GAUGELE T. Simulation of cutting processes using mesh-free Lagrangian particle methods. *Computational Mechanics*, 51 (2013) (3): 261–278.

EGGERSMANN R., KIRCHDOERFER T., REESE S., STAINIER L. ET AL. Model-free data-driven inelasticity. *Computer Methods in Applied Mechanics and Engineering*, 350 (2019): 81 – 99.

GENTE A., HOFFMEISTER H.W. & EVANS C. Chip formation in machining ti6al4v at extremely high cutting speeds. *CIRP Annals-Manufacturing Technology*, 50 (2001) (1): 49–52.

GHABOUSSI J., PECKNOLD D.A., ZHANG M. & HAJ-ALI R.M. Autoprogressive training of neural network constitutive models. *International Journal for Numerical Methods in Engineering*, 42 (1998) (1): 105–126.

GHABOUSSI J. & SIDARTA D. New nested adaptive neural networks (nann) for constitutive modeling. *Computers and Geotechnics*, 22 (1998) (1): 29–52.

GOEL A., SHERAFATI A., NEGAHBAN M., AZIZINAMINI A. ET AL. A finite deformation nonlinear thermo-elastic model that mimics plasticity during monotonic loading. *International Journal of Solids and Structures*, 48 (2011) (20): 2977–2986.

GONZÁLEZ D., CHINESTA F. & CUETO E. Learning corrections for hyperelastic models from data. *Frontiers in Materials*, 6 (2019a): 14.

GONZÁLEZ D., CHINESTA F. & CUETO E. Thermodynamically consistent data-driven computational mechanics. *Continuum Mechanics and Thermodynamics*, 31 (2019b) (1): 239–253.

GROSS D. & SEELIG T. *Fracture mechanics: with an introduction to micromechanics*. Springer, 2017.

HAGAN M.T. Training feed forward networks with the marquardt algorithm. *IEEE Transaction on Neural Networks*, 5 (1994) (6): 989–993.

HARZALLAH M., POTTIER T., GILBLAS R., LANDON Y. ET AL. A coupled in-situ measurement of temperature and kinematic fields in ti-6al-4v serrated chip formation at microscale. *International Journal of Machine Tools and Manufacture*, 130 (2018): 20–35.

HASHASH Y., JUNG S. & GHABOUSSI J. Numerical implementation of a neural network based material model in finite element analysis. *International Journal for Numerical Methods in Engineering*, 59 (2004) (7): 989–1005.

HASSOUN M.H. ET AL. *Fundamentals of artificial neural networks*. MIT press, 1995.

HOLZAPFEL G.A. Nonlinear solid mechanics: a continuum approach for engineering science. *Meccanica*, 37 (2002) (4): 489–490.

HUANG D., FUHG J.N., WEISSENFELS C. & WRIGGERS P. A machine learning based plasticity model using proper orthogonal decomposition. *Computer Methods in Applied Mechanics and Engineering*, 365 (2020): 113 008.

HUANG D., WEISSENFELS C. & WRIGGERS P. Modelling of serrated chip formation processes using the stabilized optimal transportation meshfree method. *International Journal of Mechanical Sciences*, 155 (2019): 323–333.

HUGHES T.J. *The finite element method: linear static and dynamic finite element analysis*. Prentice Hall, 1987.

IBAÑEZ R., ABISSET-CHAVANNE E., AGUADO J.V., GONZALEZ D. ET AL. A manifold learning approach to data-driven computational elasticity and inelasticity. *Archives of Computational Methods in Engineering*, 25 (2018) (1): 47–57.

IBAÑEZ R., ABISSET-CHAVANNE E., GONZÁLEZ D., DUVAL J.L. ET AL. Hybrid constitutive modeling: data-driven learning of corrections to plasticity models. *International Journal of Material Forming*, 12 (2019) (4): 717–725.

IBAÑEZ R., BORZACCHIELLO D., AGUADO J.V., ABISSET-CHAVANNE E. ET AL. Data-driven non-linear elasticity: constitutive manifold construction and problem discretization. *Computational Mechanics*, 60 (2017) (5): 813–826.

JOHNSON G.R. & COOK W.H. A constitutive model and data for metals subjected to large strains, high strain rates and high temperatures. In *Proceedings of the 7th International Symposium on Ballistics*, volume 21, pages 541–547. The Netherlands, 1983.

JOHNSON G.R. & COOK W.H. Fracture characteristics of three metals subjected to various strains, strain rates, temperatures and pressures. *Engineering Fracture Mechanics*, 21 (1985) (1): 31–48.

KINGMA D.P. & BA J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, (2014).

KIRCHDOERFER T. & ORTIZ M. Data-driven computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 304 (2016): 81–101.

KIRCHDOERFER T. & ORTIZ M. Data-driven computing in dynamics. *International Journal for Numerical Methods in Engineering*, 113 (2018) (11): 1697–1710.

KOMANDURI R. & VON TURKOVICH B. New observations on the mechanism of chip formation when machining titanium alloys. *Wear*, 69 (1981) (2): 179–188.

KORELC J. & WRIGGERS P. *Automation of Finite Element Methods*. Springer, 2016.

LABERGERE C., RASSINEUX A. & SAANOUNI K. Numerical simulation of continuous damage and fracture in metal-forming processes with 2d mesh adaptive methodology. *Finite Elements in Analysis and Design*, 82 (2014): 46–61.

LE B., YVONNET J. & HE Q.C. Computational homogenization of nonlinear elastic materials using neural networks. *International Journal for Numerical Methods in Engineering*, 104 (2015) (12): 1061–1084.

LEE E. & LIU D. Finite strain elasto-plastic theory with applications to plane-wave analysis. *Journal of Applied Mechanics*, 38 (1967): 19–27.

LEE W.S. & LIN C.F. High-temperature deformation behaviour of ti6al4v alloy evaluated by high strain-rate compression tests. *Journal of Materials Processing Technology*, 75 (1998) (1-3): 127–136.

LEFIK M. & SCHREFLER B. Artificial neural network as an incremental non-linear constitutive model for a finite element code. *Computer Methods in Applied Mechanics and Engineering*, 192 (2003) (28-30): 3265–3283.

LEMAITRE J. *A course on damage mechanics*. Springer Science & Business Media, 2012.

LI B., HABBAL F. & ORTIZ M. Optimal transportation meshfree approximation schemes for fluid and plastic flows. *International Journal for Numerical Methods in Engineering*, 83 (2010) (12): 1541–1579.

LI B., KIDANE A., RAVICHANDRAN G. & ORTIZ M. Verification and validation of the optimal transportation meshfree (otm) simulation of terminal ballistics. *International Journal of Impact Engineering*, 42 (2012): 25–36.

LI B., PANDOLFI A. & ORTIZ M. Material-point erosion simulation of dynamic fragmentation of metals. *Mechanics of Materials*, 80 (2015): 288–297.

LI X., LIU Z., CUI S., LUO C. ET AL. Predicting the effective mechanical property of heterogeneous materials by image based modeling and deep learning. *Computer Methods in Applied Mechanics and Engineering*, 347 (2019): 735–753.

LIU Z., BESSA M. & LIU W.K. Self-consistent clustering analysis: an efficient multi-scale scheme for inelastic heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, 306 (2016): 319–341.

LIU Z., WU C. & KOISHI M. A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, 345 (2019): 1138–1168.

LU X., GIOVANIS D.G., YVONNET J., PAPADOPOULOS V. ET AL. A data-driven computational homogenization method based on neural networks for the nonlinear anisotropic electrical response of graphene/polymer nanocomposites. *Computational Mechanics*, 64 (2019) (2): 307–321.

MA W., CHEN X. & SHUANG F. The chip-flow behaviors and formation mechanisms in the orthogonal cutting process of ti6al4v alloy. *Journal of the Mechanics and Physics of Solids*, 98 (2017): 245–270.

MAHNKEN R., WOLFF M. & CHENG C. A multi-mechanism model for cutting simulations combining visco-plastic asymmetry and phase transformation. *International Journal of Solids and Structures*, 50 (2013) (20-21): 3045–3066.

MARUSICH T. & ORTIZ M. Modelling and simulation of high-speed machining. *International Journal for Numerical Methods in Engineering*, 38 (1995) (21): 3675–3694.

MERCHANT H.E.M. & ERNST H. Chip formation, friction and high quality machined surfaces. *Surface treatment of metals. Am Soc Met*, 29 (1941): 299–378.

MIGUÉLEZ M., SOLDANI X. & MOLINARI A. Analysis of adiabatic shear banding in orthogonal cutting of ti alloy. *International Journal of Mechanical Sciences*, 75 (2013): 212–222.

MIMOUNA A. & TCHELEPI H. Critical time-step for central difference integration schemes in discrete methods: Translational and rotational degrees of freedom. *Computer Methods in Applied Mechanics and Engineering*, (2019).

MOHAN A.T. & GAITONDE D.V. A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks. *arXiv:1804.09269*, (2018).

MOHR D., DUNAND M. & KIM K.H. Evaluation of associated and non-associated quadratic plasticity models for advanced high strength steel sheets under multi-axial loading. *International Journal of Plasticity*, 26 (2010) (7): 939–956.

NGUYEN D. & WIDROW B. Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In *1990 IJCNN International Joint Conference on Neural Networks*, pages 21–26. IEEE, 1990.

OISHI A. & YAGAWA G. Computational mechanics enhanced by deep learning. *Computer Methods in Applied Mechanics and Engineering*, 327 (2017): 327 – 351.

PABISEK E. Self-learning FEM/NMM approach to identification of equivalent material models for plane stress problem. *Computer Assisted Mechanics and Engineering Sciences*, 15 (2008) (1): 67.

PANDOLFI A., LI B. & ORTIZ M. Modeling fracture by material-point erosion. *International Journal of Fracture*, 184 (2013) (1-2): 3–16.

PANDOLFI A. & ORTIZ M. An eigenerosion approach to brittle fracture. *International Journal for Numerical Methods in Engineering*, 92 (2012) (8): 694–714.

PRIETO J.R., CARBONELL J.M., CANTE J., OLIVER J. ET AL. Generation of segmental chips in metal cutting modeled with the PFEM. *Computational Mechanics*, 61 (2018): 639–655.

RASMUSSEN C.E. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

RODRIGUEZ PRIETO J.M., JONSÉN P. & SVOBODA A. A particle finite element method for machining simulations. In *VII European Congress on Computational Methods in Applied Sciences and Engineering, Crete Island, Greece, 5–10 June 05/06/2016-10/06/2016*, volume 1, pages 539–553. National Technical University of Athens, 2016.

ROTH C.C. & MOHR D. Effect of strain rate on ductile fracture initiation in advanced high strength steel sheets: experiments and modeling. *International Journal of Plasticity*, 56 (2014): 19–44.

SABEL M., SATOR C. & MÜLLER R. A particle finite element method for machining simulations. *Computational Mechanics*, 54 (2014) (1): 123–131.

SCHMIDT B., FRATERNALI F. & ORTIZ M. Eigenfracture: an eigendeformation approach to variational fracture. *Multiscale Modeling & Simulation*, 7 (2009) (3): 1237–1266.

SCHULZ H. The history of high-speed machining. *Revista de Ciência e Tecnologia*, 7 (1999) (13): 9–18.

SHAKOOR M., KAFKA O.L., YU C. & LIU W.K. Data science for finite strain mechanical science of ductile materials. *Computational Mechanics*, 64 (2019) (1): 33–45.

SIMA M. & ÖZEL T. Modified material constitutive models for serrated chip formation simulations and experimental validation in machining of titanium alloy ti–6al–4v. *International Journal of Machine Tools and Manufacture*, 50 (2010) (11): 943–960.

SIMO J. Algorithms for static and dynamic multiplicative plasticity that preserve the classical return mapping schemes of the infinitesimal theory. *Computer Methods in Applied Mechanics and Engineering*, 99 (1992) (1): 61–112.

DE SOUZA NETO E.A., PERIC D. & OWEN D.R. *Computational methods for plasticity: theory and applications*. John Wiley & Sons, 2011.

STAINIER L., LEYGUE A. & ORTIZ M. Model-free data-driven methods in mechanics: material data identification and solvers. *Computational Mechanics*, (2019): 1–13.

SUTTER G. & LIST G. Very high speed cutting of ti–6al–4v titanium alloy–change in morphology and mechanism of chip formation. *International Journal of Machine Tools and Manufacture*, 66 (2013): 37–43.

TANG S., ZHANG G., YANG H., LI Y. ET AL. Map123: A data-driven approach to use 1d data for 3d nonlinear elastic materials modeling. *Computer Methods in Applied Mechanics and Engineering*, 357 (2019): 112 587.

TAYLOR G. & QUINNEY H. The latent energy remaining in a metal after cold working. *Proc. R. Soc. Lond. A*, 143 (1934) (849): 307–326.

VAZ M., OWEN D., KALHORI V., LUNDBLAD M. ET AL. Modelling and simulation of machining processes. *Archives of Computational Methods in Engineering*, 14 (2007) (2): 173–204.

VILLANI C. *Topics in optimal transportation*. Number 58. American Mathematical Soc., 2003.

VILLANI C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

WEISSENFELS C. & WRIGGERS P. Stabilization algorithm for the optimal transportation meshfree approximation scheme. *Computer Methods in Applied Mechanics and Engineering*, 329 (2018): 421–443.

WRIGGERS P. *Computational contact mechanics*, volume 2. Springer, 2006.

WRIGGERS P. *Nonlinear finite element methods*. Springer Science & Business Media, 2008.

XIAO M., BREITKOPF P., COELHO R.F., KNOPF-LENOIR C. ET AL. Model reduction by CPOD and Kriging. *Structural and Multidisciplinary Optimization*, 41 (2010) (4): 555–574.

YANG H., GUO X., TANG S. & LIU W.K. Derivation of heterogeneous material laws via data-driven principal component expansions. *Computational Mechanics*, (2019): 1–15.

YE G., XUE S., JIANG M., TONG X. ET AL. Modeling periodic adiabatic shear band evolution during high speed machining ti-6al-4v alloy. *International Journal of Plasticity*, 40 (2013): 39–55.

ZHANG Y., MABROUKI T., NELIAS D. & GONG Y. Chip formation in orthogonal cutting considering interface limiting shear stress and damage evolution based on fracture energy approach. *Finite Elements in Analysis and Design*, 47 (2011) (7): 850–863.

Curriculum vitae

Dengpeng Huang
born July 3, 1989 in Gansu, China

Professional Experience

09.2015 - 11.2019	Ph.D student at the Institute of Continuum Mechanics, Leibniz Universität Hannover
11.2019 - 12.2019	Visiting researcher at the Institute of Structure Analysis, Technische Universität Dresden
since 01.2020	Scientific staff at the Institute of Applied Dynamics, Friedrich-Alexander-Universität Erlangen-Nürnberg

Education

09.2008 - 06.2012	Shandong University, China Mechanical Engineering Degree: B. Sc.
09.2012 - 06.2015	Jilin University, China Mechanical Engineering Degree: M. Sc.