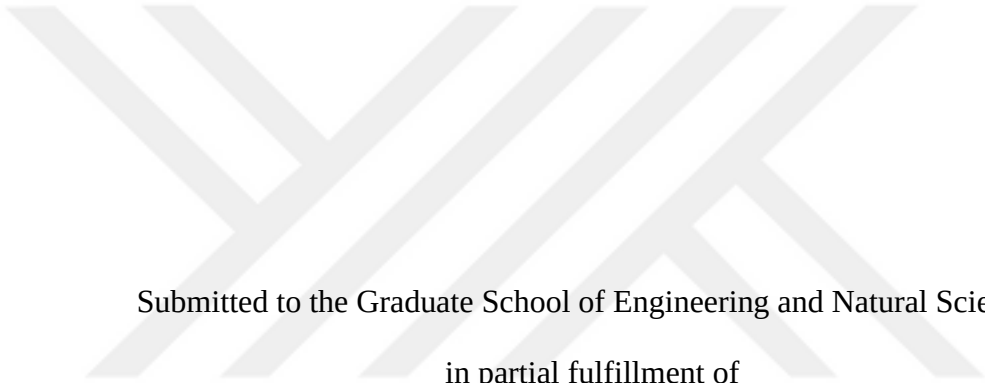


Developing a physics-based model within a crystal plasticity finite element framework to analyze texture evolution and deformation heterogeneity in dual-phase steel materials

By
Mohammad Hasan Joudivand Sarand



Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

SABANCI UNIVERSITY

December 2023

DEVELOPING A PHYSICS-BASED MODEL WITHIN A CRYSTAL
PLASTICITY FINITE ELEMENT FRAMEWORK TO ANALYZE
TEXTURE EVOLUTION AND DEFORMATION HETEROGENEITY
IN DUAL-PHASE STEEL MATERIALS

APPROVED BY:



.....

.....

.....

.....

.....

DATE OF APPROVAL:



© Mohammad Hasan Joudivand Sarand 2023

All Rights Reserved

DEVELOPING A PHYSICS-BASED MODEL WITHIN A CRYSTAL PLASTICITY FINITE ELEMENT FRAMEWORK TO ANALYZE TEXTURE EVOLUTION AND DEFORMATION HETEROGENEITY IN DUAL-PHASE STEEL MATERIALS

MOHAMMAD HASAN JOUDIVAND SARAND

Ph.D. Dissertation, December 2024

Supervisor: Prof. Dr. Burç Mısırlıoğlu

ABSTRACT

Keywords: Crystal Plasticity, Physics-Based Finite Element, Phenomenological Simulation, Dual-Phase Steel

In the first phase of the research a crystal plasticity model based on dislocation density was developed to understand the complex mechanical behavior of body-centered cubic (BCC) materials. The model incorporated non-Schmid effects using three-term projection operators. To evaluate the model, numerical results obtained under various crystal orientations, temperatures, and strain rates were compared with the corresponding experimental results reported in the literature. The capacity of the model to capture tension-compression asymmetry as one of the distinctive features of the BCC structure was also examined. Simulations were implemented at different scales. According to the results the model could accurately simulate the experimentally observed stress-strain curves and explain the tension-compression asymmetry in the flow stresses of α -iron.

In the second phase of the study, the scheme from the first phase was enhanced to feature dual phase properties. For this purpose, an isotropic (J2) plasticity subroutine was developed to represent martensite phase effects in the numerical model. It was subsequently combined with the density-based crystal plasticity finite element model (CPFEM), to simulate martensite and ferrite phases collective response in a dual phase

steel 600 (DP600). Data on the phase area fraction, crystallographic grain orientation, and microstructural texture of a sample of the DP alloy were obtained experimentally using the electron backscattered diffraction (EBSD) method and were applied to create a single-layer representative volume element (RVE) of the material.

Furthermore, the scheme is applied with synthetic models to demonstrate the evolution of simulation outputs, including dislocation density, suppression of deformation, load partitioning, and interruption of stress flow. The effect of martensite geometrical deployment with focus on banding issue on the strain localization and eventually overall hardening behavior of the dual phase steel is investigated. To evaluate the model's reliability in capturing the actual strain response, an experimental study conducted on a similar material from the literature is used. The resultant strain distribution and localization of our model are compared and analyzed against the reported experiment. Our results showed that the martensite phase plays an important role in the stress and strain distribution in the alloy by reducing plastic deformation. The misorientation evolution of the ferrite grains in a DP structure is extracted by post processing the result files, revealing the restrictive influence of the neighboring martensite regions on the ferrite phase and its impact on plasticity. The results indicate that the presence of the martensite phase leads to considerable heterogeneity in the stress distribution, an increase in misorientation polarization, and strain hardening. Ultimately, the hybrid simulation scheme that is developed in this study provides a clear insight into the effective microstructural factors in the plastic deformation behavior of DP steels, specifically the retexturing response and the effect of the martensite phase on ferrite plasticity under various loading conditions.

ÖZET

Araştırmanın ilk aşamasında, vücut merkezli kübik (VMK) malzemelerin karmaşık mekanik davranışını anlamak için bir kırılma yoğunluğuna dayalı kristal plastisite modeli geliştirildi. Model, üç terimli projeksiyon operatörlerini kullanarak non-Schmid etkilerini içeriyordu. Modelin değerlendirilmesi için çeşitli kristal yönelimleri, sıcaklıklar ve gerilme hızları altında elde edilen sayısal sonuçlar, literatürde rapor edilen karşılık gelen deneysel sonuçlarla karşılaştırıldı. Modelin VMK yapısının ayırt edici özelliklerinden biri olan gerilme-sıkışma asimetrisini yakalama yeteneği de incelendi. Simülasyonlar farklı ölçeklerde gerçekleştirildi. Sonuçlara göre model, deneysel olarak gözlemlenen gerilme-gerinme eğrilerini doğru bir şekilde simüle edebilir ve α -demirin akma gerilmelerindeki gerilme-sıkışma asimetrisini açıklayabilir.

Çalışmanın ikinci aşamasında, ilk aşamadaki düzenleme çift faz özelliklerini içerecek şekilde geliştirildi. Bu amaçla, sayısal modelde martensit faz etkilerini temsil etmek üzere izotropik (J2) plastisite bir alt programı geliştirildi. Daha sonra, bu alt program, martensit ve ferrit fazlarının toplu tepkisini simüle etmek için yoğunluk tabanlı kristal plastisite sonlu eleman modeli (CPFEM) ile birleştirildi ve çift fazlı çelik 600 (DP600) üzerinde uygulandı. DP alaşımının faz alan yüzdesi, kristalografik tahıl yönelimi ve malzemenin temsilci birim hacmini oluşturmak için elektron geri saçılma difraksiyonu (EBSD) yöntemi kullanılarak deneysel olarak elde edilen veriler, tek katmanlı temsilci hacim ögesi (RVE) oluşturmak için uygulandı.

Ayrıca, şema sentetik modellerle uygulanarak simülasyon çıktılarının evrimini, kırılma yoğunluğunu, deformasyonun bastırılmasını, yük bölümlenmesini ve stres akışının kesilmesini göstermek için kullanılır. Martensit geometrik konumunun odaklandığı bantlama sorunu üzerindeki etkisi, çift fazlı çelikte gerilme yerelleşmesi ve nihayetinde genel sertleşme davranışı üzerine incelenir. Modelin gerçek gerilme tepkisini yakalama güvenilirliğini değerlendirmek için literatürde benzer bir malzeme üzerinde gerçekleştirilen deneysel bir çalışma kullanılır. Sonuçta modelin gerilme dağılımı ve yerelleşmesi, rapor edilen deney ile karşılaştırılır ve analiz edilir. Sonuçlarımız, martensit fazının plastik deformasyonu azaltarak alaşımın stres ve gerinme dağılımında önemli bir rol oynadığını gösterdi. DP yapısındaki ferrit tanelerinin misorientasyon evrimi, sonuç dosyalarını işleme alarak çıkarılır ve komşu martensit bölgelerinin ferrit fazı üzerindeki

sınırlayıcı etkisini ve plastisite üzerindeki etkisini ortaya koyar. Sonuçlar, martensit fazının varlığının stres dağılımında önemli heterojenliklere, misorientasyon polarizasyonunda artışa ve sertleşmeye neden olduğunu göstermektedir. Sonuç olarak, bu çalışmada geliştirilen hibrit simülasyon şeması, DP çeliklerin plastik deformasyon davranışındaki etkili mikroyapısal faktörlere, özellikle retexturing tepkisine ve çeşitli yük koşulları altında martensit fazının ferrit plastisitesine etkisine net bir içgörü sağlar.





To my loving mother and my beloved father, my genuine source of inspiration, motivation, and unwavering support throughout my academic journey and personal life. I am forever grateful for the sacrifices they have made to ensure my success.

To my dearest brothers, my aunt, and my cousins, who have been my pillars of strength, my sounding boards, and my unwavering supporters throughout my life.

To my beloved uncle, who sadly passed away before he could see me achieve my academic dream.

ACKNOWLEDGMENT

I would like to take this opportunity to express my sincere gratitude to my supervisor, Prof. Dr. Burç Mısırlıoğlu, for his guidance, support, and mentorship throughout my PhD journey. His expertise, insight, and encouragement have been invaluable to me, and I am honored to have had the opportunity to work under his guidance. His constructive feedback, insightful advice, and unwavering encouragement have helped me to grow both as a researcher and as a person.

I would also like to express my deep appreciation and gratitude to my parents for their unwavering support throughout my PhD journey. Their love, encouragement, and understanding have been a constant source of strength and motivation, and I am deeply grateful for their unwavering support.

I would like to offer my appreciation to my former supervisor, Dr. Eralp Demir, for his support and mentorship in the beginning of my PhD journey.

I would like to extend a special thank you to Torkan Shafighfard, Sasan Karimi, Emel Manga, Sina Khalilvandi Behrouzgar, Can Bayraktar, Amin Balazadeh Koucheh, Kuray Dericiler, Atakan Koçanali, Peyman Khandar Shahabad, Vahid Ebrahimpour Ahmadi, Faraz Ganjdoust, Ali Barzgar, Omid Gheysar Gharamaleki and Farid Sayar Irani who have been true friends and confidants throughout this journey. Their encouragement, and friendship have been a constant source of strength, and I could not have done this without them.

My heartfelt thanks goes to my colleagues, who have become my second family during my PhD studies. I am grateful for their friendship, their humor, and their support, which have helped me to maintain a healthy work-life balance and to stay focused on my goals. You have made this journey not only manageable but also enjoyable.

Contents

ABSTRACT.....	ii
Notations, conventions and abbreviations	19
1 Introduction	20
1.1 Background and motivation	20
1.2 Mechanism of plasticity	21
1.3 Dual-phase steel	23
1.4 Microstructural modeling using crystal plasticity method.....	27
1.5 RVE.....	32
1.6 Research outline.....	33
2 Methodology and modeling implementation	35
2.1 Crystal mechanics	36
2.2 Dislocation-density based constitutive law	39
2.3 Dislocation kinetics.....	40
2.4 Dislocation evolution	42
2.5 Isotropic plasticity scheme.....	45
2.6 Ferrite and martensite mechanical properties	45
2.7 Numerical modeling in 1 st phase.....	49
2.8 Application of the material microstructure as model input data.....	51
2.9 Numerical modeling in 2 nd phase.....	53
3 Results and discussions	67
3.1 Crystal plasticity analysis.....	70
3.1.1 Single crystal orientation	71
3.1.2 Temperature	74

3.1.3	Strain-rate	78
3.1.4	Tension–compression asymmetry	82
3.2	Dual-phase RVE analysis.....	85
3.2.1	Mechanical response.....	85
3.2.2	Loading conditions effect	100
3.2.3	Microstructure orientation and phase configuration effect.....	108
3.2.4	Grain morphology and loading direction combined.....	120
3.2.5	Strain distribution assessment.....	135
4	Conclusion.....	139
5	References	146
6	Appendices	159
6.1	HYPELA2-CP Fortran subroutine	159
6.2	HYPELA2-CPJ2 Fortran subroutine	179
6.3	Material point SC MATLAB script	202
6.4	MATLAB code of EBSD data processing using MTEX toolbox.....	223
6.5	MATLAB script for phase detection from Kikuchi quality images	228
6.6	MATLAB script for plotting misorientation pole figure using MSC MARC output files.....	231
6.7	Dislocation density post processing MATLAB script	242
6.8	MATLAB script for selecting martensite banded area in the EBSD	246

List of Figures

Figure 1-1 application of DP steel in a car body	21
Figure 1-2 Optical microscopy pictures of (a) a DP-steel with larger grain size and (b) an ultra-fine DP-steel, as utilized in reference [38], where ferrite is bright and contrasts martensite which appears dark.	24
Figure 1-3 A schematic illustration for comparing the strength and ductility of various steel grades [39].	25
Figure 1-4 Tasan et al. [42] presented engineering stress-strain data for different steel grades.	26
Figure 2-1 A diagram illustrating the Schmid and non-Schmid projections for the BCC crystal structure is presented. An example of slip system $\alpha = 2$ or B4 in the Schmid-Boas notation is provided, where vectors belonging to the same plane are highlighted with the same color as the plane.	39
Figure 2-2 Separate the mechanical responses of the constituent phases, ferrite and martensite, as well as the dual-phase material.	49
Figure 2-3 Modeling schemes at different size scales: (a) MP model, (b) SE model, and (c) ETG.....	49
Figure 2-4 The specified and fixed displacements in the boundary conditions for the MP model (a), SE model (b), and ETG model (c) are represented by red and black arrows, respectively.	51
Figure 2-5 EBSD orientation map for the DP600 material on the Rolling direction (RD)-TD plane, IPF RD. Totally 1157156 data points recorded in a $100\mu m \times 100\mu m$ square. Data points are logged in the form of hexagons. For each of the grains corresponding Euler angles (ϕ_1 , Φ and ϕ_2) are recorded that will be used in generating the RVE for the simulations.....	52
Figure 2-6 Experimental uniaxial tensile test result for DP600 steel sample obtained at a strain rate of 0.00025 s^{-1} . The specimen is tested under 0° inclination. There are 4 tests performed on 4 samples and the average values of these tests are given above.....	53

Figure 2-7 grain size distribution of α phase, (a) major axis length, (b) minor axis length and (c) area fraction distribution.....	55
Figure 2-8 crystallographic state of the phases as per EBSD measurement results (a) grain shape aspect ratio ferrite, (b) grain shape aspect ratio martensite., (c) grain orientation spread ferrite, (d) grain orientation spread martensite., (e) major axis orientation ferrite, (f) major axis orientation distribution martensite	56
Figure 2-9 IPF plots in three pole directions [0 0 1], [0 1 1] and [-1 1 1], (a) for original EBSD, (b) for the created RVE models.....	57
Figure 2-10 square meshed (converted) selected area	58
Figure 2-11 Nodes ID in the CPFEM model for assigning the boundary conditions.	60
Figure 2-12 Schematic of the procedure for identifying the martensite phase in grains and its utilization in each of the discretization schemes using the image processing method outlined in the text.....	62
Figure 2-13 Image quality map in RD-ND plane (a) only ferrite phase distribution with filtered martensite regions (white regions represent martensite phase), (b) only martensite phase distribution with filtered ferrite regions (white background represent ferrite matrix)	64
Figure 2-14 Euler angles difference between adjacent ferrite elements in three different discretization size	64
Figure 2-15 martensite grains superimposed on the grain distribution in 51x51 discretization plot, black squares represent martensite, areas with similar color tones represent a single grain of ferrite	65
Figure 2-16 The procedure involves processing a screenshot extracted from Figure 2 in Pelligra et al. publication [153] to generate a 25×25 model. In this figure, black cells represent the martensite phase, while white cells depict the ferrite phase. During this process, the martensite-to-ferrite ratio is maintained at 40.5%.....	66
Figure 2-17 distances in the MCF calculation method	68

Figure 3-1 The stress-strain relationship for various crystal orientations under tensile loading is dependent on the model scale, as shown in (a) MP, (b) SE, and (c) ETG. The experimental data used in this analysis is from [5] 73

Figure 3-2 The maximum distribution of (a) resolved shear stress (RSS) ($\tau\alpha$) and (b) the cumulative RSS, including non-Schmid projections ($\tau\alpha + \sigma\alpha_{ns}$) for tensile loading, are displayed. The values for the non-Schmid constants are also provided. 73

Figure 3-3 The impact of crystal orientation on α -iron, through the calibration of hardening interaction coefficients, is examined using experimental data obtained from [6] 74

Figure 3-4 The stress-strain graphs of MP, SE, and ETG models were analyzed in relation to temperature, and were compared to experimental data [155]. 76

Figure 3-5 The relationship between flow stress, temperature, and crystal orientation is depicted as follows: (a) the computation of flow stress for the [1 1 16] tensile direction, (b) the results of flow stress for three distinct tensile directions of [1 1 16], [1 3 7], [4 9 9] and at four different temperatures of 143 K, 195 K, 250 K, and 295 K. χ refers to the deviation from the single slip orientation, [1 3 7]. 76

Figure 3-6 The influence of temperature on the stress-strain diagrams is demonstrated for the MP, SE, and ETG models in (a), (b), and (c) respectively, alongside a comparison to the experimental data [156] 78

Figure 3-7 The engineering stress-strain diagrams for various strain rates of the MP, SE, and ETG models are presented in (a), (b), and (c), respectively, along with a comparison to the reference [4] 80

Figure 3-8 Stress-strain curves for strain rates using different model scales of (a) MP, (b) SE, (c) ETG models and comparison to the reference [157]. 81

Figure 3-9 The tension-compression asymmetry of α -iron single crystals was simulated at various temperatures using different model scales, namely MP, SE, and ETG, as well as experimental results from [156]. The purpose of this was to compare the simulated asymmetry across different models and temperatures. 84

Figure 3-10 overlaid stress-strain graphs of the experimental and numerical tensile tests. The results report the plastic deformation behavior for different grain (element) size in numerical simulations, $0.49\ \mu\text{m} \times 0.49\ \mu\text{m}$, $0.373\ \mu\text{m} \times 0.373\ \mu\text{m}$ and $0.247\ \mu\text{m} \times 0.25$

μm . In the legend of figure, SP, DP and Num. terms mean single phase, dual phase and numerical, respectively. 85

Figure 3-11 The distribution of effective strain in the tensile scenario with a discretization size of $0.49\mu\text{m} \times 0.49\mu\text{m}$ is illustrated as follows: (a) for the HYPELA2-CPJ2 model (representing the dual phase, DP), and (b) for the HYPELA2-CP model (representing the single phase, SP). 87

Figure 3-12 mobile dislocation density distribution in $0.49\mu\text{m} \times 0.49\mu\text{m}$ RVE model, (a) single phase simulation result (HYPELA2-CP), (b) dual phase distribution (HYPELA2-CPJ2), martensite regions distribution can be distinguished with small white areas encompassed with wide black borders.....89

Figure 3-13. Martensite region distribution is superimposed on the deformed RVE model (for stress and strain distribution) in tensile simulation for the $0.49\mu\text{m} \times 0.49\mu\text{m}$ grain size model. The figure show that how regions with extreme strain/stress values are stretched/compressed between the martensite regions. In the figure Dis. Stands for distribution. 90

3-14 inhomogeneity of plastic deformation distribution histograms and CVs, (a) Single Phase case, (b) Dual Phase case.....91

Figure 3-15 A synthetic model with all ferrites in a cubic orientation is used. Elements number 23, 48, 64, 66, and 89 are randomly selected and assigned to the martensite phase in order to establish a 5% martensite area fraction. 92

Figure 3-16 MDD contour plots in a 10×10 synthetic model, (a) single phase simulation case, (b) dual phase simulation case. 94

Figure 3-17 strain distribution in the synthetic unidirectional oriented model, (a) single phase simulation result, (b) dual phase simulation result.....95

Figure 3-18 simulated 14×14 synthetic model, numbers inside the ferrite elements represent their element number in the simulation. 96

Figure 3-19 strain and stress distribution in 14×14 synthetic model, (a) strain distribution in single-phase ferrite, (b) strain distribution in dual-phase ferrite-martensite, M in the figure shows martensite regions. 97

Figure 3-20 selected martensite banded area, (a) Martensite distribution with the selected area boxed, (b) processed area to extract 51x51 RVE. Martensite area fraction is 9.4%.	99
Figure 3-21 comparison of the mechanical response of the main RVE (with 4.4% martensite) and banded RVE in tensile (with 9.4% martensite).	99
Figure 3-22 strain distribution in the selected martensite banded area, gray cells represent martensite lathes overlaid on the DP RVE.	99
Figure 3-23 Stress-strain results in $0.49 \times 0.49 \mu m$ grain size. Num. Ten.: Numerical result for tensile simulation, Num. Comp.: Numerical result for compression, Num. Shear: Numerical results for shear simulation, Exp. Ten: Experimental data from tensile testing, Exp. Comp.: Experimental data from achieved Figure 2 in Ghaei et al. study [154], Exp. Shear.: Experimental data taken from Figure 3 reported in Ghaei et al. study [154].	101
Figure 3-24 ϵ_{11} distribution in the DP steel sample with $0.49 \mu m \times 0.49 \mu m$ grain sized HYPELA-CPJ2 (DP) model under compression.	102
Figure 3-25 Misorientation pole figures of the $0.49 \mu m \times 0.49 \mu m$ HYPELA-CPJ2 model.	104
Figure 3-26 10 x 10 RVE sample where the element numbers are designated. Overall sample size is 25 x 25 μm . The orientation of the grains is defined randomly.	105
Figure 3-27 resultant distributions of the simulated 10×10 RVE, (a) strain distribution in compression, (b) strain distribution in tensile, (c) pure non-Schmid stress term contour plot for compression, (d) pure non-Schmid stress term contour plot for tensile.	106
Figure 3-28 average degree of misorientation in the simulated 10×10 RVE under compression and tensile cases.	107
Figure 3-29 (a) average mobile dislocation density versus strain in compression and tensile cases, (b) MDD distribution contour plot in tensile, (c) MDD distribution contour plot in compression.	108
Figure 3-30 simulation cases diagram, totally 8 different simulation combinations are performed.	110

Figure 3-31 domains shown with distinctive colors, (a) domains to define one martensite element randomly for each of them in fully scattered style, (b) domains employed to define four martensite elements randomly for each of them in concentrated martensite dispersion style. 111

Figure 3-32 martensite distribution in each of the styles of the simulations, (a) scattered style D1, (b) scattered style D2, (c) concentrated style D3, (d) concentrated style D4, in all of the style the martensite volume fraction is maintained at 11.1%. 112

Figure 3-33 ODF functions distribution of (a) O1 case and (b) O2 case. 113

Figure 3-34 strain distribution in 12×12 RVE, (a) scattered D1O1 (distribution 1 orientation 1), (b) scattered D1O2, (c) scattered D2O1, (d) scattered D2O2, (e) concentrated D3O1, (f) concentrated D3O2, (g) concentrated D4O1, (h) concentrated D4O2. 115

Figure 3-35 MDD distribution contour plots of the simulation combinations, (a) scattered D1O1, (b) scattered D1O2, (c) scattered D2O1, (d) scattered D2O2, (e) concentrated D3O1, (f) concentrated D3O2, (g) concentrated D4O1, (h) concentrated D4O2. 117

Figure 3-36 synthetic fully scattered martensite distribution, (a) D5 case, (b) D6 case, (c) D7 case, and (d) D8 case. 118

Figure 3-37 strain versus $MCF \times \kappa$ 119

Figure 3-38 ODF of the applied RVEs, (a) $0.49\mu\text{m} \times 0.49\mu\text{m}$, (b) $0.37\mu\text{m} \times 0.37\mu\text{m}$, and (c) $0.247\mu\text{m} \times 0.25\mu\text{m}$ 122

Figure 3-39 Three discretization level of the numerical models used in the simulations, (a) $0.49\mu\text{m} \times 0.49\mu\text{m}$ grain size, (b) $0.37\mu\text{m} \times 0.37\mu\text{m}$ grain size, (c) $0.247\mu\text{m} \times 0.25\mu\text{m}$ grain size discretization. The white squares represent the coordinates for the martensite phase ($\sim 4.4\%$ in all grids). 124

Figure 3-40. Stress-strain graphs for the three discretization levels in HYPEL-CPJ2, (a) tensile case, (b) compression case, experimental result is taken from Figure 2 reported by Ghaei et al. [154]. 125

Figure 3-41 strain and stress distribution plots in the simulated RVEs, (a) strain in $0.49\mu\text{m} \times 0.49\mu\text{m}$ case, (b) strain in $0.37\mu\text{m} \times 0.37\mu\text{m}$ case, (c) strain in $0.247\mu\text{m} \times 0.25\mu\text{m}$ case. 126

Figure 3-42 localized and concentrated responses are marked, white marks encircle the maximums and black ones encircle the minimums in strain distributions.....	126
Figure 3-43 Misorientation pole figures of the simulated RVEs, (a) $0.49\mu\text{m} \times 0.49\mu\text{m}$ case, (b) $0.37\mu\text{m} \times 0.37\mu\text{m}$ case (c) $0.247\mu\text{m} \times 0.25\mu\text{m}$ case, and (d) superimposed.	128
Figure 3-44 large-sized RVE morphology and martensite distribution illustration, (a) $3.57\mu\text{m} \times 3.57\mu\text{m}$ (7×7) case morphology, (b) 7×7 martensite distribution with 4.1% volume fraction (c) $2.50\mu\text{m} \times 2.50\mu\text{m}$ (10×10) case morphology, (d) 10×10 martensite distribution with 4% volume fraction, (e) $1.78\mu\text{m} \times 1.78\mu\text{m}$ (14×14) case morphology, and (f) 14×14 martensite distribution with 4.6% volume fraction.....	130
Figure 3-45 ODFs of the larg-sized RVEs, (a) 7×7 case, (b) 10×10 case, and (c) 14×14 case.	131
Figure 3-46 resultant strain distribution of the simulations with the developed large-sized RVEs, (a) tensile 7×7 , (b) tensile 10×10 , (c) tensile 14×14 , (d) compression 7×7 , (e) compression 10×10 , and (f) compression 14×14	132
Figure 3-47 deformation hotspots distribution evolution with the decrease of elements size, maximum points are shown in using white ellipses and minimum zones are marked by black ones (a) tensile case, and (b) compression case.	133
Figure 3-48 MDD distribution contour plots of the large size RVEs, (a) thensile 7×7 , (b) tensile 10×10 , (c) tensile 14×14 , (d) compression 7×7 , (e) compression 10×10 , (f) compression 14×14	134
Figure 3-49 double axis scatter plot average MDD, average degree of misorientation and RVE size.....	135
Figure 3-50 maximum MDD evolution in tenisle and compression cases.	135
Figure 3-51 κ/MCF versus CV for main realistic models with different element sizes	137
Figure 3-52 Contrasting the experimental strain distribution from reference [153] with the numerical results: (a) displays the experimental distribution of average strain, with a highlighted purple box marking the selected area, while (b) exhibits the corresponding total strain distribution within the deformed model.	139

Figure 3-53 superimposed martensite elements on the strain distribution achieved from the simulation to compare with the distribution provided in [153]......139

Figure 3-54 Processed phase distribution of [163] experimental investigation.....140

Figure 3-55 strain localization and inhomogeneity experimental validation with respect to Basu et al. [163] study, in both experimental distribution results from the literature the red box shows the area which our simulation model covers (a) experiemental reference strain distribution in 10% elongation, (b) our numerical result in undeformed state under 10% elongation, (c) experiemental reference strain distribution in 15% elongation, (d) our numerical result in undeformed state under 15% elongation. 141



List of Tables

Table 2-1 The slip directions and slip plane normals in the crystal reference frame for BCC materials are represented by $\mathbf{s}^a$ and $\mathbf{n}^a$, respectively. These correspond to the 111 family of directions and {110} family of planes. The non-Schmid stresses that cause tension-compression asymmetry are calculated using $\mathbf{n1}\alpha$	37
Table 2-2 Details and description of constants that are calibrated in the simulations for different experimental findings	46
Table 2-3 description and values of constants	47
Table 2-4 Material parameters utilized for the martensite lath model with isotropic characteristics.....	48
Table 2-5 Numerical simulation setting and microstructural features of the DP600 RVEs	58
Table 2-6 Description of the boundary conditions used in the simulations for various stress loadings.	60
Table 3-1 The reference [6] employs experimental test cases to investigate the impact of crystal orientation on α -iron.	71
Table 3-2 The reference [70] utilized experimental test cases to examine the temperature dependency of α -iron [155].....	74
Table 3-3 The reference [156] employed experimental test cases to investigate the temperature dependence of α -iron	77
Table 3-4 The reference [4] utilized numerical test cases to examine the strain rate dependency of α -iron.	79
Table 3-5 Numerical test cases used for strain rate dependence of α -iron in the reference [157].	80
Table 3-6 Numerical test cases used for strain rate dependence of α -iron in the reference [156].	82
Table 3-7 Description and values of constants applied for mechanical response calibration.....	82

Table 3-8 O1 and O2 von Mises-Fisher concentration parameter κ	113
Table 3-9 calculated MCF for the simulation combinations	118
Table 3-10 resultant κ/MCF in each of the simulation combination	119
Table 3-11 MCF, κ and CV factors for each of the element size in the real simulation cases.	136



Notations, conventions and abbreviations

Vector and tensor quantities are denoted using bold letters and symbols, while scalar variables and parameters are represented by regular, non-bold letters and symbols. We follow Einstein's summation convention for repeated indices, and we specify the range of free (or dummy) indices before (or after) the relevant equation.

- $\dot{(\)}$ time derivative of a variable
- $(\)^{-1}$ inverse of a variable
- $\cdot$ inner product operator
- $\times$ cross product operator
- $:$ double contraction operator
- $\otimes$ tensor product operator
- $(\)^T$ transpose of a tensor
- $tr(\)$ trace of a tensor

1 Introduction

1.1 Background and motivation

To improve the level of mechanical strength in a formed sheet metal, high performance materials including dual-phase steels are increasingly employed by automotive industry. The significance of this subject becomes even more pronounced when it comes to the weight of a vehicle since it has an impact on fuel consumption and subsequently air pollution issues. On the other hand, aerodynamic behavior considerations necessitate the external design of a vehicle to adhere some specific geometrical standards. This leads to a significant complexity during forming operations. Accordingly, there is a compromise between mechanical strength and forming capacity of such steel grades. Forming operations are usually expensive to experiment and also modify. Therefore, numerical simulations can be conducted to predict behavior of a DP steel under various forming conditions and consequently to select the most efficient experimental settings. For this reason, this study is aimed to broaden the understanding in this field in order to:

- Apply effective forming parameters
- Achieve a well-engineered texture
- Avoid or reduce unintended mechanical response
 - Spring back
 - Earing
 - Fretting
 - Fracture and cracking
 - Wrinkling



Figure 1-1 application of DP steel in a car body

Our study's results can be applied in microscopic and macroscopic analysis of a dual-phase steel to determine effective parameters in mechanical response and forming behavior of the material. Consequently, the most efficient strength to formability ratio of a DP steel with considering its geometry and texture can be defined by a cost effective and reliable approach. Accordingly, it is necessary to delve deeper into the topics related our research including:

- Mechanism of plasticity
- Dual phase material
- Crystal plasticity finite element method

1.2 Mechanism of plasticity

Plasticity refers to the ability of a material to undergo permanent deformation when subjected to an applied stress or load. In crystalline metals plasticity mainly takes place due to dislocation glide. A dislocation glides when its corresponding resolved shear stress (RSS) reach a critical value. RSS refers to the component of stress that acts in a single direction within a slip system, causing the material to deform by slip in the plane of dislocation. Schmid's law asserts that the critical plane-direction combinations for active slip systems are those possessing the highest RSS. Several commonly employed metals,

whether they are single-phase or multi-phase (such as steel, silicon, tantalum, and others), exhibit a BCC crystal structure. A comprehensive comprehension of the deformation in single crystals is essential to effectively simulate the mechanical properties of polycrystalline materials possessing intricate microstructures. Accordingly, developing physics-based models that incorporate strain rate, inherent temperature, and crystal orientation dependencies of bcc materials is highly important for precisely predicting the mechanical behavior of both single and polycrystalline materials, as emphasized in recent research studies [1-3].

Plastic deformation in BCC structured metals is governed by the glide of screw dislocations [4]. In materials with a BCC structure, deformation occurs along certain crystallographic $\langle 111 \rangle$ directions, which belong to specific closed-packed planes. This phenomenon is commonly known as ‘pencil glide’ where the slip occurs on different glide planes. Based on experimental observations, it was found that the crystallographic slip planes predominantly include $\{110\}$, $\{112\}$, and $\{123\}$ within a broad temperature range. This results in a total of 48 active slip systems [5]. The most commonly observed active slip planes for bcc materials at room temperature were $\{110\}$ and $\{112\}$ [6-15].

In BCC structured metals, deformation is driven by lattice friction instead of obstacle resistance which is the case for FCC metals [16]. The plastic strain rate in BCC materials is primarily governed by the movement of screw dislocations. This is because edge dislocations tend to move at a relatively higher speed compared to screw dislocations, which move in a manner characterized by double kinks. Accordingly, Peierls stress threshold was introduced to the developed physics based models as a barrier with thermal dependence in activation in earlier research studies [17]. The flow stress in BCC metals is determined by the behavior of screw dislocations, and as a result, pertinent kinetics associated with screw dislocations were employed to ascertain the yield point under various stages of hardening [18].

As atomistic simulations indicate, in screw dislocations, the core is supposed to transform prior to the slip of dislocation [19]. This transformation can be influenced by shear stress components in the slip directions generated in planes other than the main slip plane, as well as by shear stress components in the direction perpendicular to the Burgers vector. Accordingly, the impact of these non-glide stress components originating in the secondary slip planes on the dislocation cores and consequently the dislocation motion causes the violation of the Schmid law [20] or in other terms non-Schmid effect [21]. This

observation is also substantiated by mechanical tests performed on single crystals of bcc materials, which revealed notable deviations from the Schmid law [22].

BCC materials plastic deformation behavior can be considerably influenced by two main parameters, temperature and the rate of deformation [23, 24]. Tests conducted at low-temperature values under both tension and compression loading modes on high-purity single crystals of niobium [10, 25, 26] and molybdenum [27, 28] indicated significant temperature dependence in their deformation response. This phenomenon was linked to the distribution of screw dislocation core [19]. It addresses the movement of dislocations with smaller RSS (violation of the Schmid law) under the influence of stress components applied to secondary conjugated planes [29]. As the atomistic simulations revealed the glide of screw dislocations in BCC crystals were not restricted to their primary slip planes. Instead, they dispersed into multiple planes perpendicular to the $\langle 111 \rangle$ slip direction, with the most prominent among them belonging to the $\{110\}$ family of planes [19, 21, 30, 31]. As a result, the mechanical response of a bcc crystal relied significantly on both its crystallographic orientation concerning the applied load and the nature of the load, whether it was compressive or tensile.

To achieve a deeper understanding how deformation occurs and to correlate the exerted stress to the plastic deformation BCC crystal and its geometry have been explored in the literature [14, 29, 32-34]. Comparison of BCC structure behavior in different crystal orientations for various materials also clearly demonstrated that the behavior of screw dislocations is influenced by the normal stress components perpendicular to the slip direction in addition to shear stresses [35]. Three parameter laws were formulated for $\{110\}$ crystallographic family of slip planes effectively. Therefore, crystal plasticity models were modified by including the three non-Schmid terms. This adjustment was made to more accurately account for the tension-compression asymmetry and the crystal orientation-dependent behavior observed in bcc metals [36].

1.3 Dual-phase steel

As its name implies, dual-phase steels are composite materials comprising two distinct phases, with BCC ferrite typically serving as the matrix where the second phase develops. BCC ferrite is known as the stable form of pure iron at room temperature. At

extremely high temperatures (912°C), ferrite undergoes a complete transformation into FCC structure which is another phase named austenite [37]. In the context of the research discussed here, the focus is on ferrite as the matrix of primary interest.

The type of second phase that emerges within the ferrite matrix can vary depending on both the heat treatment applied and the amount of carbon present:

- Martensite forms when austenite is rapidly cooled (quenched) to a temperature near room temperature. The microstructure of martensite can vary depending on the carbon content. However, typically, it takes on a shape reminiscent of sharp needles closely packed together, giving the appearance of blocks or islands within the ferrite matrix.
- Pearlite: got its name because when observed under a microscope, it closely resembles mother of pearl. This structure forms when austenite is gradually cooled below 727°C .
- Bainite: is another microstructure that results from reducing the temperature during the transformation of austenite, with its formation typically commencing at around 540°C .

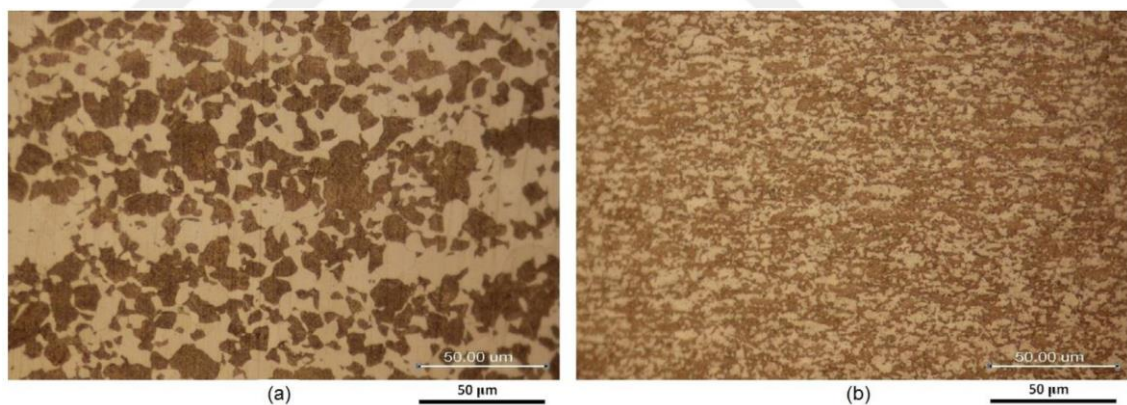


Figure 1-2 Optical microscopy pictures of (a) a DP-steel with larger grain size and (b) an ultra-fine DP-steel, as utilized in reference [38], where ferrite is bright and contrasts martensite which appears dark.

Of the three microstructures mentioned earlier, the one that has gained significant attention is the dual-phase steel composed of ferrite and martensite. Ferrite is a relatively soft phase, but its strength is enhanced by the presence of extremely hard and robust martensite blocks. Consequently, dual-phase steels composed of ferrite and martensite showcase an excellent balance between ductility and high strength. In Figure 1-3, a comparison is made between three steel types regarding their strength and total

elongation. It's clear that dual-phase steel demonstrates a significantly superior balance of strength and ductility when compared to traditional types of steel.

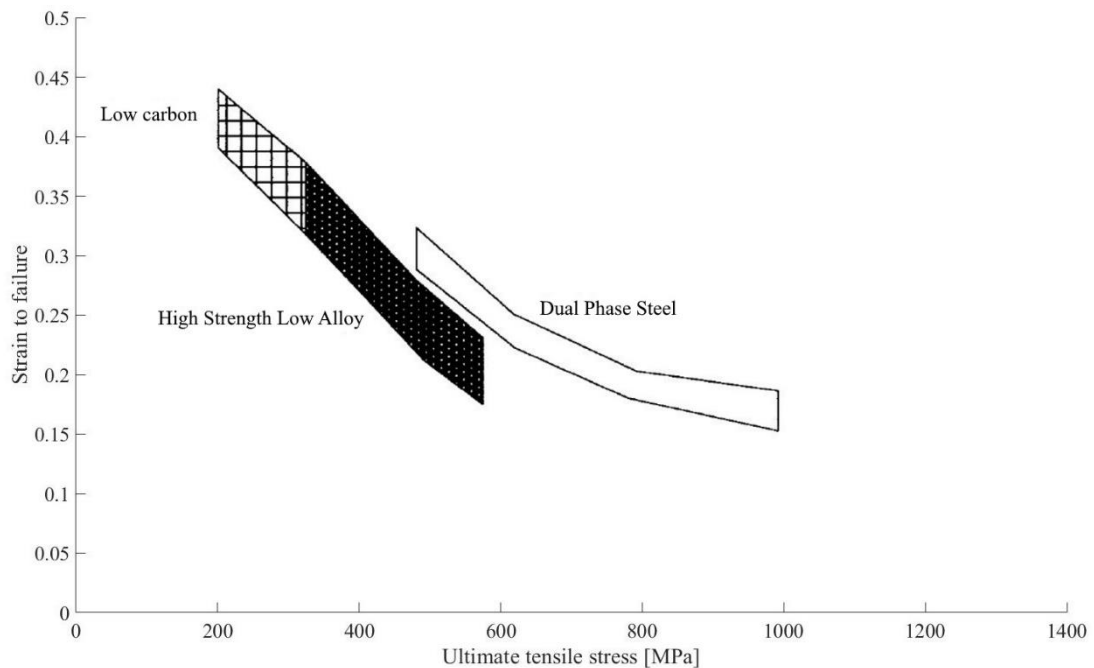


Figure 1-3 A schematic illustration for comparing the strength and ductility of various steel grades [39].

The automotive industry is increasingly adopting DP steels, particularly in response to the current automotive industry's emphasis on enhancing fuel efficiency while maintaining the strength and safety of vehicles. DP steel is a material capable of offering this unique combination of properties. Below are some of the characteristics that have rendered DP steel an appealing material option for automobile manufacturers [39]:

- High strain hardening in DP steels leads to their excellent capacity of formability (see Figure 1-4).
- In DP steels, Lüder elongation and yield point elongation are absent in their plastic deformation response which could result in aesthetic defects.
- DP steels exhibit bake hardening characteristics, a treatment process that boosts their strength. Consequently, when welded structures go through paint baking processes, their strength is improved.
- The outstanding formability of DP steels coexists with their strength without any compromise.

Dual-phase steels find application in manufacturing wheel components and side impact bars, as suggested by Vasilakos et al. [40]. Increasing demands for enhanced roof crush strength to improve passenger safety during rollovers necessitate materials capable of withstanding impacts without adding excess weight to the vehicle. DP steels also find use in cold-heading processes to manufacture high-strength fasteners, as noted by Wanjara et al. [41]. Cold-heading procedures involve heat treatment, which can lead to challenges like quench cracking and distortion. To avoid the heat treatment step, one solution is to employ highly malleable materials with the right strain hardening attributes. Microstructures that incorporate a second phase, such as martensite within a ferrite matrix, exhibit this desirable characteristic.

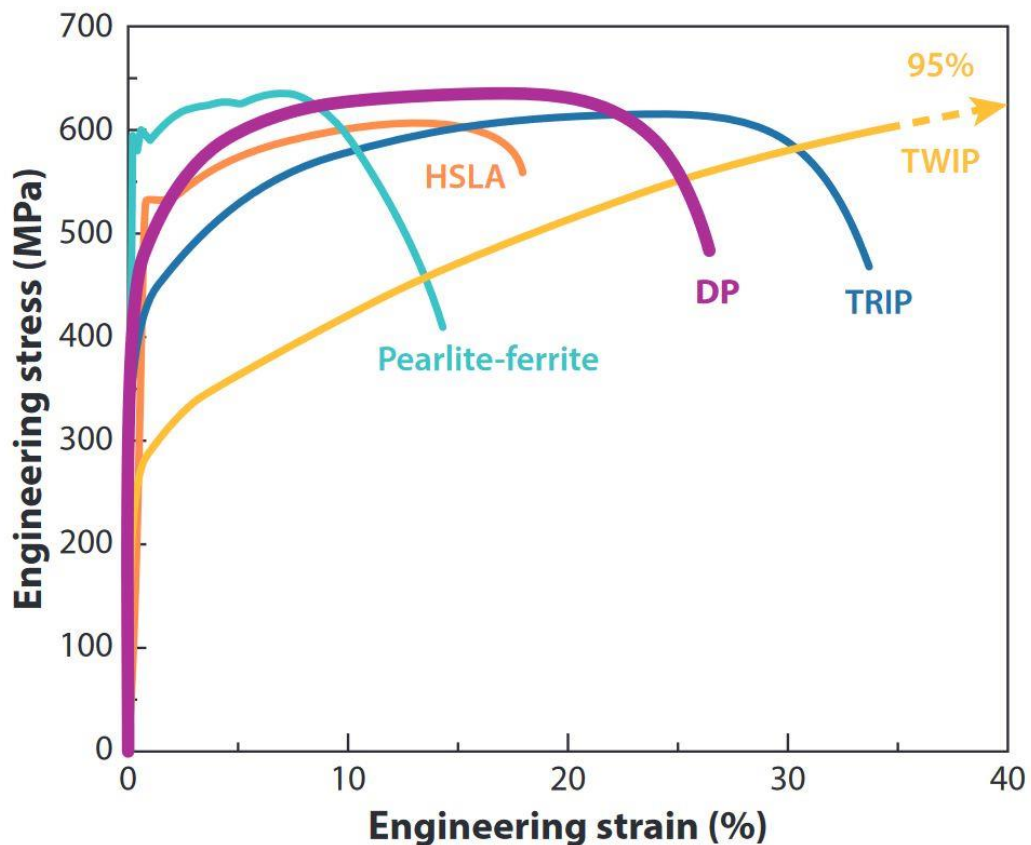


Figure 1-4 Tasan et al. [42] presented engineering stress-strain data for different steel grades.

The mentioned applications, which represent just a portion of DP steel's potential uses, have motivated researchers to delve deeper into the study of DP steels and their properties. Numerous studies have been conducted to explore the connection between the mechanical properties of DP steels and their physical and chemical attributes. In the remainder of the section, we provide an overview of such research endeavors.

1.4 Microstructural modeling using crystal plasticity method

Research on DP steels has indicated that the martensite and ferrite phases exhibit uneven behavior during plastic deformation due to their distinct strength (load-displacement relationship) and work hardening characteristics, as observed in nanoindentation experiments (Ghassemi-Armaki et al. [43]) and micropillar tests (Chen et al. [44]; Tian et al. [45]), as well as through the graph superposition method (Habibi et al. [46]). Considering the intricate interactions occurring within and between the component phases within a DP structure, the use of numerical modeling approaches becomes essential and is widely employed to better understand the micromechanical behavior of DP steels. Researchers have developed various crystal plasticity models that differ fundamentally and can be categorized into two groups. The first group includes upper-bound models, such as mean free self-consistent models (Riyad et al. [47]; Riyad et al. [48]; Zecevic and Knezevic [49]) and Taylor-based models (Gudmundson and Dahlberg [50]; Knezevic and Kalidindi [51]). The second group comprises full-field microstructural models, including Green's function (Eghesad and Knezevic [52, 53]; Li et al. [54]) and spatially resolved crystal plasticity finite element (CPFE) models (Ardeljan and Knezevic [55]; Barrett et al. [56]). The CPFEM presents an encouraging foundation for developing a comprehensive plasticity theory. This theory aims to blend well-established knowledge of deformation physics (Arsenlis et al. [57]) with the computational capabilities of continuum mechanics (Zienkiewicz and Taylor [58]), with the ultimate objective of devising advanced, physics-based design methods for engineering applications (Roters et al. [59]). Models based on CPFE theory offer greater flexibility in incorporating the physics of deformation, thanks to their consideration of crystallography in deformation mechanisms. One notable advantage of CPFE models lies in their ability to tackle intricate crystal mechanical challenges, particularly when confronted with complex internal and external boundary conditions (Sachtleber et al. [60]). These models excel at capturing the development of microscopic structures and the inherent anisotropy in elasto-plastic behavior (Daroju et al. [61]). Additionally, the CPFEM numerical approach can also be employed for analyzing material parameters derived from micropillar tests (Bittencourt [62]).

Models based on dislocation density have been employed to provide a more accurate representation of the temperature-dependent aspects of metal plasticity physics [63-70].

The dislocation density-based model enabled a more accurate depiction of dislocation kinetics, including phenomena like the Orowan law, Peierls barrier and thermally activated slip. Furthermore, an improved hardening response was achieved by employing a dislocation interaction matrix that accounts for the primary dislocation interactions among the slip systems [70-73]. Hence, there has been a growing trend toward using physics-based relationships to describe the response of polycrystals or the deformation behavior of a single crystal under complex loading conditions, such as during indentation.

Crystal plasticity models were employed to forecast the yield behavior of bcc single crystals, taking into account non-Schmid effects. As a result, Qin and Bassani [74, 75] proposed a phenomenological approach to include the impact of non-glide stresses in the flow rule. This was achieved by introducing an effective yield stress term as the non-Schmid contribution. Likewise, Steinmann et al. [76] were among the first to introduce the utilization of two extra non-Schmid components as projections onto the slip plane's normal and the co-slip direction. Yalcinkaya et al. [77, 78] employed physics-based thermal activation models for material point (MP) simulations, which encompassed asymmetry related to twinning and anti-twinning within the slip plane families {110} and {112}. Constitutive models in crystal plasticity were utilized to establish a connection between the crystal orientation dependency of the critical RSS. These models specifically considered the non-Schmid terms associated with the {110} slip plane family, accounting for both positive and negative slip directions through the use of three distinct non-Schmid terms [18, 79]. Lately, simulations were conducted on polycrystals to explore the combined influence of non-Schmid terms and crystallographic texture. These simulations encompassed both three-parameter and five-parameter non-Schmid effective terms [80].

The assessment of plastic behavior of DP steels depends on two key factors: the accuracy of the available single crystal constants for martensite and ferrite and the composition of the microstructure, as outlined by Cantara et al. [81]. Microstructure models are primarily constructed using parameters obtained through experimental methods like scanning and transmission electron microscopy, as demonstrated by Sedighiani et al. [82]. Constructing the RVE for DP alloys through the incorporation of experimental data concerning the texture, distribution of component phases, and crystallographic orientations facilitates the precise utilization of a CPFEM analysis. To reliably replicate the experimental behavior of the constituent ferrite and martensite phases in DP steels, it is essential to employ a combination of simulation schemes,

considering their significantly distinct mechanical properties. Simulations utilizing the CPFEM in conjunction with isotropic J2 schemes to represent ferrite and martensite, respectively, demonstrate that the mechanical strength of DP steel improves with the growth of the martensite volume fraction, as shown in the studies by Cai et al. [83] and Habibi et al. [46]. It has been observed that with an increase in the volume of martensite, the behavior of DP steels becomes less influenced by the specific shape of the martensite regions, as indicated by Yalçinkaya et al. [84]. Phase duality numerical techniques have been employed to examine the impact of morphology on failure characteristics (Yalçinkaya et al. [85]) as well as formability (Chatterjee et al. [86]; Chen et al. [44]; Qayyum et al. [87]). These investigations aim to shed light on the anisotropic and localized deformation behavior of DP steels.

Various microscopic hybrid plasticity models have been developed to capture the deformation behavior associated with phase duality by incorporating both kinematic hardening and isotropic laws. In this regard, various methods have been employed to capture the dual characteristics present in the microstructural properties of DP steel. Among these approaches, three distinct combinations of plasticity models have been employed for the ferrite and martensite phases within the dual-phased finite element method (FEM) model. In one method, it is assumed that both ferrite and martensite experience isotropic deformation, but they possess distinct plasticity parameters, as demonstrated in studies by Govik et al. [88], Paul and Kumar [89], Ramazani et al. [90], and Sun et al. [91, 92]. An alternative method involves considering isotropic deformation for martensite while utilizing the crystal plasticity (CP) scheme to describe the behavior of ferrite, as outlined in the study by Kadkhodapour et al. [93]. Alternatively, the CP approach, as demonstrated by Chen et al. [44], Kim et al. [94], Tasan et al. [95], and Woo et al. [96], is employed for modeling both ferrite and martensite. To reduce the computational expenses, scientists have adopted a variety of techniques throughout various phases of the numerical modeling procedure. These approaches span from employing selective sampling techniques to predict single crystal characteristics (as shown by Biswas et al. [97]; Bonatti et al. [98], Castillo and Kalidindi [99], Sedighiani et al. [82], Venkatraman et al. [100]) to simplifying the model's underlying physics (demonstrated by Hollenweger and Kochmann [101], Sahoo et al. [102], Zhang et al. [103]).

One can utilize a two-dimensional microstructure that is obtainable through experimental extraction from real samples. Subsequently, this extracted input can be incorporated into the simulation to accurately depict the way stress and strain are distributed in the dual-phase scenario. An exemplary study in this regard is the work conducted by Kadkhodapour et al. [104]. They employed an isotropic-CP dual-phased model to examine the processes behind void initiation and expansion during the failure of DP steels. Their findings indicated that the primary reason for the material failure was the non-uniform deformation behavior of martensite and ferrite phases. Ramazani and colleagues [90] conducted a detailed examination of crack formation and propagation in various DP steels. They achieved this by employing a RVE model designed to simulate their test materials numerically. Kim and co-authors [105] conducted an analysis of the effects of phase morphology and grain orientation on interfacial de-cohesion in dual-phased steel material by combining the CPFEM and RVE methods. The study's findings suggested that while 2D models offer faster simulations, achieving a strong alignment between experimental tests and numerical predictions, especially in accurately forecasting the flow curve, is better accomplished through 3D modeling of dual-phase steel using the RVE method rather than 2D modeling. Eghtesad and Knezevic [106] utilized a numerical platform to simulate the elasto-plastic monotonic and cyclic deformation responses of various types of DP steel. This platform incorporated a dislocation density law to capture the hardening behavior. In a separate study focusing on the same material, the authors found that martensite exhibited nearly isotropic behavior, whereas ferrite displayed a high level of anisotropy in its mechanical response. This was documented in a paper by Cantara et al. [81]. To comprehend the impact of phase duality and mechanical anisotropy between the tough martensite regions and the more yielding ferrite matrix, Zhang et al. [107] employed a crystal plasticity simulation method in various tensile directions when studying a DP steel. Distinct crystal plasticity constitutive parameters were applied to the individual phases by the research group. Knezevic and colleagues [108] utilized a physically grounded crystal plasticity model to simulate the behavior of DP steel under cyclic tension-compression loading conditions. This study found that the precision in capturing the behavior of the martensite phase is crucial for achieving accuracy in the mechanical response.

While there have been numerical investigations into the behavior of dual-phase steels under various loading conditions, such as forward-reversal cyclic loading (as

demonstrated by Daroju et al. [61], Zecevic et al. [49]), tension either uniaxial or biaxial (as explored by Hussein et al. [109], Jafari et al. [110]), and compression (as studied by Jia et al. [111]), there remains a significant deficiency in conducting a comprehensive and integrated examination of the effects of these different loading conditions.

In-situ mechanical testing setups offer the opportunity to directly collect data on strain distribution, enabling microstructural imaging during the deformation process and thus facilitating microscopic-digital image correlation (μ DIC) analysis. This approach has been demonstrated in studies by Ghadbeigi et al. [112], Han et al. [113], and Joo et al. [114]. Through the utilization of this approach, research has revealed that the detrimental effects of microstructural banding are greatly influenced by factors such as the band's continuity, the mechanical properties of the phase comprising the band, and its morphology, as demonstrated by Tasan et al. [115]. Additionally, studies have indicated that the distribution of strain displays notable variations, which become more pronounced as the overall deformation level rises, as reported by Kapp et al. [116]. These areas of concentrated deformation, often referred to as "hot spots," tend to appear within the ferrite channels located between the larger martensite regions. They also occur at grain boundaries that run perpendicular to the applied load. To enhance the experimental testing methods, a combination of microlithography-based patterning and microstructure characterization through EBSD is applied. This approach helps to clarify the role of various microstructural parameters in strain heterogeneity, as demonstrated in studies by Han et al. [113] and Marteau et al. [117]. The results presented in recent research clearly demonstrate that the μ DIC method is a potent tool for capturing the intricate patterns of strain distribution that develop in DP steels.

Simulation studies on single-phase steels have sought to establish connections between grain orientation, loading conditions, stress/strain distribution, and misorientation evolution. These investigations have been conducted using crystallographic reconstruction methods like DAMASK and viscoplastic self-consistent (VPSC), as demonstrated by Galán-López and Hidalgo [118], Sinha et al. [119], and Zhang et al. [120]. However, it's worth noting that similar research on materials with a DP structure remains unexplored.

1.5 RVE

The majority of engineering materials exhibit non-uniform distribution of components on a microscopic scale. RVEs serve as a valuable means to simulate this heterogeneity present in materials [121]. It is a fundamental cell that is presumed to repeat uniformly within the material and is of sufficient size to replicate the material's overall mechanical properties. Therefore, the critical aspect in modeling RVE is the determination of its size, along with the chosen approach for averaging the non-uniformity of constituents (or properties of the constituents). In the past, conventional techniques, as demonstrated by Mori and Tanaka [122], were employed to create RVEs for DP steel. However, with the advent of miniature sample testing and more efficient data collection methods, it is now possible to generate RVEs based on actual micrographs obtained using SEM or TEM, as illustrated in the work by Barlat et al. [123]. Numerous scholars have played a role in advancing the methodology for creating comprehensive RVEs capable of efficiently producing macroscopic data.

In their review, Mishnaevsky et al. [124] undertook one of the initial endeavors to explore ideas applicable for simulating ductile failure through FEM. In a separate publication, Mishnaevsky and colleagues [125] aimed to enhance the characteristics of tool steel, which is a two-phase material. They attempted to simulate fracture in both actual microstructure and idealized forms, utilizing Multiphase Finite Element (MPFE) and Element Elimination Technique (EET). Sun and colleagues [91] constructed an RVE for DP steel by employing High Energy X-Ray Diffraction (HEXRD) during an in-situ experiment with the loaded sample. The micrograph obtained from this in-situ experiment was initially divided into its individual phases, namely ferrite and martensite. Properties were attributed to the phases by matching the experimental data with Ludwik's equation. To simulate the 2D RVE for uniaxial testing, plane stress and plane strain elements were utilized.

The relaxed grain cluster approach is a version of the grain interaction model [126] that accounts for finite-strain effects. The model discussed by Roters et al. [127] was implemented in the context of cup drawing of DP steels. While there has been increasing interest in utilizing multiscale finite element (FE²) techniques for DP steels, the process of creating more manageable representative volume elements (RVEs) remains challenging. Schröder and colleagues [128] conducted a study on creating RVEs and

simulations, and noted that a form of weak coupling, inspired by FE² methods, can also be employed. With this method, using the outcomes of a corresponding large-scale simulation the loading conditions for the small-scale model are determined. As an illustration, Uthaisangsuk and colleagues [129] utilized a hole expansion simulation to determine the boundary conditions for an RVE model, in order to study stretch flangeability, a crucial factor in evaluating specific intricate sheet metal forming processes. The same idea was applied by Prawoto and team [130], who included damage modeling in their implementation, to simulate impact loading.

Since the lattice structure and parameters of ferrite and martensite are alike, it is not possible to differentiate between them using traditional methods including EBSD. Furthermore, within DP Steel, the boundary between martensite and ferrite exhibits an elevated dislocation density because of the transformation process. As the dislocation count rises, the quality of micrographs produced by EBSD deteriorates, as noted by De Cooman et al. [131]. When it comes to creating a RVE using actual microstructural data, the EBSD technique can be employed. However, to assign the phases accurately, supplementary methods such as image quality processing techniques need to be employed.

1.6 Research outline

This research project can be divided into two main phases:

- Phase one: developing a physics-based crystal plasticity scheme to simulate BCC structured material deformation
- Phase two: creating a dual-phase calculation scheme and introducing DP steel material's phase effect to the simulation

Within the scope of the research study, in the first phase, an innovative physics-based model was developed, and the mechanical behavior of metal materials was modeled using the crystal plasticity finite element method. A distinct crystal plasticity model based on physics of the deformation was created to address the Schmid and non-Schmid characteristics of BCC materials. The developed CPFEM was not previously available in the literature. It was conducted to compute mechanical and microstructural properties of parts after metal forming operations. The physics-based constitutive laws were

established in a unique and with a much simpler form than the models available in the literature. The single state variable that governs the yield behavior of BCC single crystal was the dislocation density, which was utilized to calculate other properties like mobile dislocation density, dislocation evolution rates, slip rates, among others. The model incorporated non-Schmid effects by including three non-Schmid parameters that were derived from simulations based on dislocations, which only took into account the {110} family of crystallographic planes [18]. The developed physics-based model was employed to replicate the existing experimental and numerical research outcomes for α -iron BCC single crystals at three distinct modeling levels: the material point (MP) with one quadrature point, single finite element (SE) containing one element and eight quadrature points, and exact test geometry (ETG) comprising multiple finite elements. The proposed model was utilized to thoroughly study the impact of crystallographic orientation, strain rate, and temperature dependence on the stress-strain response (including yield, hardening behavior, and tension-compression asymmetry) of BCC single crystals.

In the second phase of the research, the crystallographic texture for a polycrystal material was experimentally obtained by using EBSD method and the measurements were applied to create a numerically representative model replicating the material. The dual phase steel behavior was numerically captured through a dual scheme featuring the martensite and ferrite phases effects. The scheme featured an isotropic J2 plasticity section, and a rate-dependent crystal plasticity section, which was already developed in the first phase of the study, to simulate martensite and ferrite phases respectively. Calculations were effectuated by writing two different FORTRAN subroutines and applying them in the MSC MARC package. Experimental EBSD data was processed using MTEX toolbox in MATLAB to achieve models with different grain refinement dimensions with keeping identical phase fraction ratios. An image quality distribution, which discriminates phases in the EBSD measurement as a bi-modal distribution was used to differentiate ferrite and martensite regions; both of which were normally indexed as BCC regions. Phases were numerically assigned to grains in the models to replicate a dual-phase material in the scheme. Simulation was performed for the physically meaningful quantities of stress and strain distribution and texture, and eventually the results were compared to achieve a computationally efficient simulation setup.

For post processing the analysis results, a method was developed to track evolution in the orientation of the crystals during numerical plastic deformation process. The proposed approach was applied to compute texture evolution in a polycrystal under different loading cases. Furthermore, the modeling approach was applied to analyze the effect of martensite phase in deformation behavior of the dual phase steel under different loading modes.

2 Methodology and modeling implementation

Section 2.1 to 2.2 details the mechanics of crystal plasticity that are employed to create the basic calculation scheme, which covers Schmid and non-Schmid features specific to BCC materials. In Section 2.3 and 2.4, physically based constitutive relations are introduced that establish a connection between mechanical properties, crystal orientation, temperature, and strain-rate. Section 2.4 specifically provides the formulations that are applied to associate the evolution of dislocations to the plastic deformation behavior of the model. Isotropic calculation scheme is formulated in Section 2.5 to explain how martensite effect is included. Mechanical properties of the constituent phases are detailed in Section 2.6. Section 2.7 provides an explanation of the numerical model created using MATLAB and the MSC MARC FEM package to execute the subroutine. In section 2.8 and 2.9, a technique is described for integrating phases, crystal orientations, and various plasticity calculation methods into the numerical simulations. Section 2.9 renders an explanation of the microstructure of DP600 steel, where the grain orientations and phase fraction are determined through EBSD experiments. Furthermore, within the same section, the specifications of the model utilized to simulate the dual-phase scheme are outlined. The model results and discussion are presented in Section 0, where the findings are compared to those of previous experiments in the literature for single crystal α -iron and for DP600 steel. The conclusions are presented in Section 0.

2.1 Crystal mechanics

As explained in [132], equation (1) formulates the decomposition of the total deformation gradient¹, $\mathbf{F}$, into its elastic component, $\mathbf{F}_e$, and plastic component, $\mathbf{F}_p$, through a multiplication process. The crystal plasticity problem involves determining the proportion of deformation that is elastic or plastic for a given total deformation amount:

$$\mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_p \quad (1)$$

Equation (2) employs the Green-Lagrange strain method to determine the elastic strains, $\mathbf{E}_e$, for large deformations:

$$\mathbf{E}_e = \frac{\mathbf{F}_e^T \cdot \mathbf{F}_e - \mathbf{I}}{2} \quad (2)$$

The elasticity is utilized to compute the vectorized version of the stress responsible for dislocation glide in the intermediate configuration (known as the 2nd Piola-Kirchhoff stress), $\boldsymbol{\sigma}$. The elasticity matrix $\mathbf{C}$ and the vectorized form of the elastic strain $\mathbf{e}_e$ are represented in equation (3):

$$\boldsymbol{\sigma} = \mathbf{C} \cdot \mathbf{e}_e \quad (3)$$

Equation (4) defines $\mathbf{C}$ as the elasticity matrix in the crystal frame, which is constructed utilizing three temperature-dependent elastic constants for cubic metals, namely C_{11} , C_{12} , and C_{44} :

$$\mathbf{C} = \begin{bmatrix} C_{11} & C_{12} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{11} & C_{12} & 0 & 0 & 0 \\ C_{12} & C_{12} & C_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{44} \end{bmatrix} \quad (4)$$

Equation (5) is used to derive the resolved shear stress (RSS), τ^α , by projecting the 2nd Piola-Kirchhoff stress $\boldsymbol{\sigma}$ onto the slip system α through the Schmid tensor, $\mathbf{S}^\alpha$.

¹ Bold uppercase and lowercase letters are utilized to represent second-order tensors and vectors, respectively, where A equals A_{ij} and b equals b_i . Scalars are indicated using regular font letters. The inner product is denoted by the symbol $\cdot$, such that $A : A = A_{ij}A_{ij}$. The identity matrix is denoted by $\mathbf{I}$ and has dimensions of three by three. The transpose of a matrix is denoted with a superscript T .

$$\tau^\alpha = \mathbf{F}_e^T \cdot \mathbf{F}_e \cdot \boldsymbol{\sigma} : \mathbf{S}^\alpha \quad (5)$$

Equation (5) describes the Schmid tensor, $\mathbf{S}^\alpha$, as the product of the slip direction ($\mathbf{s}^\alpha$) and the slip plane normal ($\mathbf{n}^\alpha$) for slip system α :

$$\mathbf{S}^\alpha = \mathbf{s}^\alpha \otimes \mathbf{n}^\alpha \quad (6)$$

The slip plane normal vectors and slip directions for BCC materials are provided in Table 2-1, which has been extracted from reference [66]. It is assumed that only one family of slip planes $\{110\}$ is active, and the slip occurs separately along positive and negative directions, resulting in a total of 24 potential slip systems.

Equation (7) defines the governing plastic flow rule based on kinematics:

$$\dot{\mathbf{F}}_p = \mathbf{L}_p \cdot \mathbf{F}_p \quad (7)$$

Equation (8) describes the process of obtaining the plastic velocity gradient, $\mathbf{L}_p$, by adding up the plastic slip rates $\dot{\gamma}^\alpha$ for each slip system:

$$\mathbf{L}_p = \sum_{\alpha=1}^{24} \tilde{\mathbf{S}}^\alpha \dot{\gamma}^\alpha \quad (8)$$

Equation (9) is utilized to compute the Cauchy stress, which represents the stress at the deformed configuration $\tilde{\boldsymbol{\sigma}}$:

$$\tilde{\boldsymbol{\sigma}} = \frac{\mathbf{F}_e \cdot \boldsymbol{\sigma} \cdot \mathbf{F}_e^T}{\det(\mathbf{F}_e)} \quad (9)$$

Table 2-1 The slip directions and slip plane normals in the crystal reference frame for BCC materials are represented by $\mathbf{s}^\alpha$ and $\mathbf{n}^\alpha$, respectively. These correspond to the $\langle 111 \rangle$ family of directions and $\{110\}$ family of planes. The non-Schmid stresses that cause tension-compression asymmetry are calculated using $\mathbf{n}_1^\alpha$.

Slip sys.	$\mathbf{s}^\alpha$	$\mathbf{n}^\alpha$	$\mathbf{n}_1^\alpha$	Slip sys.	$\mathbf{s}^\alpha$	$\mathbf{n}^\alpha$	$\mathbf{n}_1^\alpha$
1 (B2)	[111]	(01 $\bar{1}$)	($\bar{1}$ 10)	13	$[\bar{1}\bar{1}\bar{1}]$	(01 $\bar{1}$)	(10 $\bar{1}$)
2 (B4)	[111]	($\bar{1}$ 01)	(0 $\bar{1}$ 1)	14	$[\bar{1}\bar{1}\bar{1}]$	($\bar{1}$ 01)	($\bar{1}$ 10)
3 (B5)	[111]	(1 $\bar{1}$ 0)	(10 $\bar{1}$)	15	$[\bar{1}\bar{1}\bar{1}]$	(1 $\bar{1}$ 0)	(0 $\bar{1}$ 1)

4 (A3)	$[\bar{1}11]$	$(\bar{1}0\bar{1})$	$(\bar{1}\bar{1}0)$	16	$[1\bar{1}\bar{1}]$	$(\bar{1}0\bar{1})$	$(01\bar{1})$
5 (A2)	$[\bar{1}11]$	$(0\bar{1}1)$	(101)	17	$[1\bar{1}\bar{1}]$	$(0\bar{1}1)$	$(\bar{1}\bar{1}0)$
6 (A6)	$[\bar{1}11]$	(110)	$(01\bar{1})$	18	$[1\bar{1}\bar{1}]$	(110)	(101)
7 (C1)	$[\bar{1}\bar{1}1]$	$(0\bar{1}\bar{1})$	$(1\bar{1}0)$	19	$[11\bar{1}]$	$(0\bar{1}\bar{1})$	$(\bar{1}0\bar{1})$
8 (C3)	$[\bar{1}\bar{1}1]$	(101)	(011)	20	$[11\bar{1}]$	(101)	$(1\bar{1}0)$
9 (C5)	$[\bar{1}\bar{1}1]$	$(\bar{1}10)$	$(\bar{1}0\bar{1})$	21	$[11\bar{1}]$	$(\bar{1}10)$	(011)
10 (D4)	$[1\bar{1}1]$	$(10\bar{1})$	(110)	22	$[\bar{1}1\bar{1}]$	$(10\bar{1})$	$(0\bar{1}\bar{1})$
11 (D1)	$[1\bar{1}1]$	(011)	$(\bar{1}01)$	23	$[\bar{1}1\bar{1}]$	(011)	(110)
12 (D6)	$[1\bar{1}1]$	$(\bar{1}\bar{1}0)$	$(0\bar{1}\bar{1})$	24	$[\bar{1}1\bar{1}]$	$(\bar{1}\bar{1}0)$	$(\bar{1}01)$

Equation (10) contains three terms that together represent the non-Schmid effects. These terms are represented by constants a_1 , a_2 and a_3 :

$$\sigma_{ns}^\alpha = [a_1 \mathbf{s}^\alpha \otimes \mathbf{n}_1^\alpha + a_2 (\mathbf{n}^\alpha \times \mathbf{s}^\alpha) \otimes \mathbf{n}^\alpha + a_3 (\mathbf{n}_1^\alpha \times \mathbf{s}^\alpha) \otimes \mathbf{n}_1^\alpha] : \boldsymbol{\sigma} \quad (10)$$

Reference [133] displays an example case, slip system $\alpha = 2$ or B4, depicting the directions and planes of the three non-Schmid terms in Figure 2-1. The non-Schmid terms are detailed as follows:

- $\sigma_{ns1}^\alpha = \mathbf{s}^\alpha \otimes \mathbf{n}_1^\alpha : \boldsymbol{\sigma}$ – This term refers to a stress component that is aligned with the slip direction and lies on a plane that is at an angle of 60 degrees to the glide plane. This stress component represents the non-Schmid term that causes twinning/anti-twinning asymmetry.
- $\sigma_{ns2}^\alpha = (\mathbf{n}^\alpha \times \mathbf{s}^\alpha) \otimes \mathbf{n}^\alpha : \boldsymbol{\sigma}$ – This term refers to a stress component that is perpendicular to the slip direction and lies on the slip plane. This stress component represents the non-Schmid term that causes tension-compression asymmetry.
- $\sigma_{ns3}^\alpha = (\mathbf{n}_1^\alpha \times \mathbf{s}^\alpha) \otimes \mathbf{n}_1^\alpha : \boldsymbol{\sigma}$ – This term refers to a stress component that is perpendicular to the slip direction and lies on a plane that is at an angle of 60

degrees to the glide plane. This stress component represents the non-Schmid term that causes tension-compression asymmetry.

The resultant shear stress on the α slip system, τ_{eff}^α , can be calculated by combining the non-Schmid stress component, σ_{ns}^α , with the RSS (Schmid term) and then removing the impact of long-range stresses as shown in equation (11):

$$\tau_{eff}^\alpha = \tau^\alpha + \sigma_{ns}^\alpha - \tau_\mu^\alpha \quad (11)$$

The method proposed by Dawson et al [134], known as the two-level semi-implicit time integration method, is utilized. This method is based on using a plastic initial guess, which in comparison to numerical methods that rely on stress guess is more stable and smooth converging [135]. A thorough explanation of the numerical approach was presented in [136].

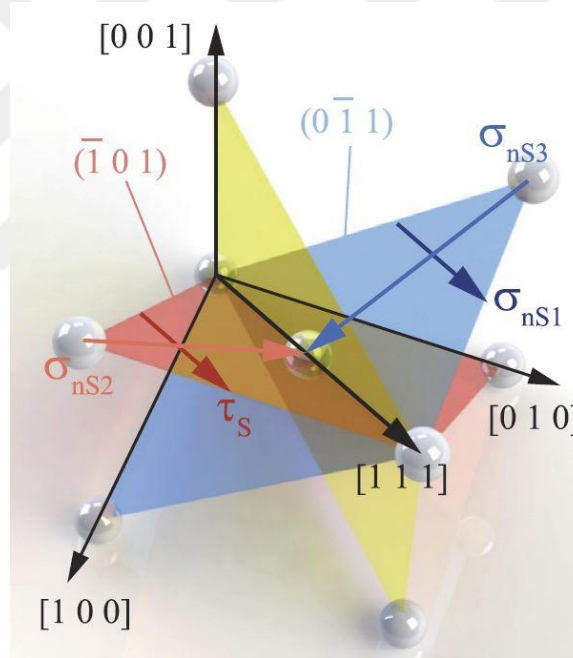


Figure 2-1 A diagram illustrating the Schmid and non-Schmid projections for the BCC crystal structure is presented. An example of slip system $\alpha = 2$ or B4 in the Schmid-Boas notation is provided, where vectors belonging to the same plane are highlighted with the same color as the plane.

2.2 Dislocation-density based constitutive law

This section describes the constitutive laws that rely on the density of dislocations in single crystals. Besides the stress per slip system, the state variable is the density of single

dislocations at the slip system α , denoted as ρ^α . All other constitutive quantities are determined by utilizing this single state variable per slip system.

The study defines forest dislocations as dislocations that penetrate the slip plane and have a density of ρ_f^α . Equation (12) used to calculate the forest dislocation density involves the projection of slip plane normals, which is a distinctive aspect of this research.

$$\rho_f^\alpha = \sum_{\beta=1}^{24} (1 - \mathbf{n}^\alpha \cdot \mathbf{n}^\beta) \rho^\beta \quad (12)$$

2.3 Dislocation kinetics

To determine the shear rates on the slip systems, $\dot{\gamma}^\alpha$, the Orowan equation in rate form is employed. Equation (13) includes the Burgers vector b , mobile dislocation density ρ_m^α , and average glide velocity v^α [137]:

$$\dot{\gamma}^\alpha = \rho_m^\alpha b v^\alpha \quad (13)$$

Equation (14) used to calculate the average glide velocity, v^α , involves computing the geometric average of the velocities of viscous drag, v_{dr}^α , and thermal activation, v_{th}^α [69]:

$$\frac{1}{v^\alpha} = \frac{1}{v_{dr}^\alpha} + \frac{1}{v_{th}^\alpha} \quad (14)$$

For BCC metals, the thermally activated velocity depends on the work done by the applied effective stress, τ_{eff}^α , to overcome the Peierls barrier, which is a short-range obstacle in an existing long-range athermal stress field, τ_μ^α . Equation (15) uses the following terms: Q_{Pei} , K_B , T , v_0 , d^α , and V^α . These terms correspond to Peierls activation energy, Boltzmann constant, temperature, Debye frequency, jump distance, and activation volume, respectively. The effective stress and shear stress must always be positive in direction because negative slip systems are taken into account [16]:

$$v_{th}^\alpha = \begin{cases} v_0 d^\alpha \exp\left(-\frac{Q_{Pei}}{K_B T}\right) \exp\left(\frac{\tau_{eff}^\alpha V^\alpha}{K_B T}\right), & \text{if } (\tau_{eff}^\alpha > 0) \text{ and } (\tau^\alpha > 0) \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

The temperature and rate-independent component of the flow stress, known as athermal flow stress, is responsible for the long-range portion of the flow stress denoted as τ_μ^α . The temperature sensitivity of τ_μ^α arises solely from the temperature sensitivity of the shear modulus, G [138]. The internal stress fields that operate over long distances are referred to as long-range stress fields, which are unaffected by thermal fluctuations to a great extent. Equation (16) calculates the athermal component of the flow stress, τ_μ , where c_1 represents the geometric factor and ρ^α denotes the primary dislocation density:

$$\tau_\mu^\alpha = c_1 G b \sqrt{\rho^\alpha} \quad (16)$$

The distance a dislocation line or jump can move without encountering obstacles, referred to as the obstacle-free distance, can be expressed as d^α . This distance is inversely proportional to the square root of the forest dislocation density, as demonstrated in equation (17), where the scaling constant is represented by c_2 [137, 138]:

$$d^\alpha = \frac{c_2}{\sqrt{\rho_f^\alpha}} \quad (17)$$

The activation volume, denoted as V^α , is directly proportional to the cubic value of the Burgers vector b , which is represented by b^3 . Equation (18) that expresses this relationship also includes a constant, c_3 , that remains unchanged [5]:

$$V^\alpha = c_3 b^3 \quad (18)$$

The equation (14) calculates the velocity at which drag occurs. This velocity is determined by a linear relationship with the effective stress, which is expressed in equation (19) [69]:

$$v_{dr}^\alpha = \frac{b \tau_{eff}^\alpha}{B_0} \quad (19)$$

The value of B_0 , which represents the constant that governs drag, is determined by the equation (20). This equation involves three variables: q , c_s , and b , which respectively refer to the number of atoms per unit cell (which is equal to two for BCC materials), the speed of shear waves, and the Burgers vector. The speed at which shear waves propagate, denoted as c_s , can be determined by dividing the shear modulus G by the mass density ρ . In other words, the equation to calculate the shear wave speed is $c_s = \sqrt{G/\rho}$:

$$B_0 = \frac{3K_B T q}{20c_s b^2} \quad (20)$$

Equation (21) states that the mobile dislocation density [16], ρ_m^α , is a fraction of the total dislocation density, which is represented by $r(\tau^\alpha, \rho_f^\alpha)$:

$$\rho_m^\alpha = r(\tau^\alpha, \rho_f^\alpha) \rho^\alpha \quad (21)$$

To calculate the mobile dislocation density, a simplified version of the log-normal distribution is used. This distribution is based on the average distance between dislocations. To determine the critical bow-out radius, the effective stress that is applied is taken into account. Any dislocation segments that are longer than this critical bow-out radius are assumed to be mobile, as indicated by equation (22) and source [72]:

$$r(\tau^\alpha, \rho_f^\alpha) = c_7 (2\pi)^{3/2} \left(\frac{\tau^\alpha}{G} \right)^2 \frac{1}{\rho_f^\alpha} \quad (22)$$

2.4 Dislocation evolution

The dislocations are restricted after moving a distance of the average-free path, which is inversely proportional to the square root of the density of dislocations [138]. Equation (23) is used to calculate the increase in dislocation density, $\dot{\rho}_{mul}^\alpha$, caused by the multiplication of dislocations:

$$\dot{\rho}_{mul}^\alpha = c_4 \frac{\sqrt{\rho^\alpha}}{b} |\dot{\gamma}^\alpha| \quad (23)$$

Another way in which mobile dislocations can be stored is through the formation of dipoles, which is referred to as $\dot{\rho}_{dip}^\alpha$. Previous studies have shown that the rate of storage is inversely related to the applied stress [71]. The Poisson ratio is represented by the symbol ν in equation (24).

$$\dot{\rho}_{dip}^\alpha = c_8 \frac{\sqrt{3} G b \rho_m^\alpha}{16\pi(1-\nu)\tau_{eff}^\alpha} |\dot{\gamma}^\alpha| \quad (24)$$

At high temperatures, dislocation climb can occur, and it is referred to as a reaction for the annihilation of dislocations, denoted by $\dot{\rho}_{climb}^\alpha$. According to Frost and Ashby's

research [139], it is assumed that the rate of dislocation climb depends on both the climb stress σ^α and the effective diffusion rate D_{eff} , as indicated in equation (25).

$$\dot{\rho}_{climb}^\alpha = -c_5 \frac{D_{eff} b^3 \sigma^\alpha}{2K_B T} (\rho^\alpha)^2 |\dot{\gamma}^\alpha| \quad (25)$$

Equation (26) shows that the effective climb rate D_{eff} is determined by the coefficients of volume diffusion D_v and core diffusion $a_c D_c$, which are related to the diffusion of atoms through the volume and the core of the material, respectively.

$$D_{eff} = D_v + \rho^\alpha a_c D_c \quad (26)$$

Equation set (27) describes the relationship between the climb activation energies (Q_v and Q_c) and the reference rates of volume and core diffusion D_{v0} and $a_c D_{c0}$, which are used to determine the diffusion rate terms in the effective climb rate equation.

$$\begin{aligned} D_v &= D_{v0} \exp\left(-\frac{Q_v}{K_B T}\right) \\ D_c &= a_c D_{c0} \exp\left(-\frac{Q_c}{K_B T}\right) \end{aligned} \quad (27)$$

In only a small number of dislocation density models, such as those referenced in [2] and [140], is the force along the slip plane normals considered the driving force for dislocation climb. This means that the stress driving the climb σ^α , as shown in equation (28), is perpendicular to the resolved shear stress (RSS), rather than being aligned with it:

$$\sigma^\alpha = \mathbf{n}^\alpha \otimes \mathbf{n}^\alpha : \boldsymbol{\sigma} \quad (28)$$

Cross-slip is the primary mechanism for dislocation annihilation in BCC materials and is denoted as ρ_{cs}^α . This model utilizes the thermally activated cross-slip model proposed by Nes [141], which is represented by equation (29):

$$\dot{\rho}_{cs}^\alpha = -\frac{2\sqrt{\rho^\alpha} v_0}{c_6^2 b} \exp\left(-\frac{Q_{cs}}{K_B T}\right) \exp\left(\frac{g}{K_B T}\right) |\dot{\gamma}^\alpha| \quad (29)$$

Equation (31), χ , is utilized to calculate the cross-slip energy barrier, Q_{cs} , and the activation energy for cross-slip, g . This equation involves the stacking fault energy of the material (represented by A and B), along with two other material constants:

$$g = \frac{\tau_{eff}^\alpha b}{\chi} [\ln(BGb/\chi)]^{-0.5} \quad (30)$$

$$Q_{cs} = AGb^3 \frac{Gb}{\chi} [\ln(BGb/\chi)]^{0.5} \quad (31)$$

Equation (32) is used to calculate the total dislocation density by adding up all the reactions:

$$\dot{\rho}^\alpha = \frac{1}{2} \chi_\beta^\alpha (\dot{\rho}_{mul}^\beta + \dot{\rho}_{dip}^\beta + \dot{\rho}_{climb}^\beta + \dot{\rho}_{cs}^\beta) \quad (32)$$

Equation (32) uses χ_β^α to represent the dislocation interaction matrix, which considers the variation in hardening interactions between slip systems caused by different mechanisms for lock formation [142]. Equation (33) specifies the form of the interaction matrix for the 24 slip systems as a whole:

$$\chi_\beta^\alpha = \begin{bmatrix} X_\beta^\alpha & X_\beta^\alpha \\ X_\beta^\alpha & X_\beta^\alpha \end{bmatrix} \quad (33)$$

Equation (34) contains dislocation interaction coefficients g_{d0} , G_0 , g_1 , g_2 , g_3 and g_4 , which correspond to various types of dislocation interactions including self, coplanar, glissile, Hirth lock, and Lomer-Cottrell lock. The coefficients for these interactions are given as 0.0, 1.0, 1.5, 3.2, 2.4, and 1.8, respectively:

$$X_\beta^\alpha = \begin{bmatrix} g_{d0} & G_0 & G_0 & g_3 & g_1 & g_3 & g_3 & g_3 & g_3 & g_3 & g_3 & g_3 \\ G_0 & g_{d0} & G_0 & g_3 & g_1 & g_3 & g_3 & g_3 & g_3 & g_1 & g_1 & g_1 \\ G_0 & G_0 & g_{d0} & g_3 & g_1 & g_3 & g_1 & g_1 & g_1 & g_3 & g_3 & g_3 \\ g_3 & g_3 & g_3 & g_{d0} & G_0 & G_0 & g_1 & g_1 & g_1 & g_3 & g_3 & g_3 \\ g_1 & g_1 & g_1 & G_0 & g_{d0} & G_0 & g_3 & g_3 & g_3 & g_3 & g_3 & g_3 \\ g_3 & g_3 & g_3 & G_0 & G_0 & g_{d0} & g_3 & g_3 & g_3 & g_1 & g_1 & g_1 \\ g_3 & g_3 & g_1 & g_1 & g_3 & g_3 & g_{d0} & G_0 & G_0 & g_2 & g_2 & g_2 \\ g_3 & g_3 & g_1 & g_1 & g_3 & g_3 & G_0 & g_{d0} & G_0 & g_3 & g_3 & g_3 \\ g_3 & g_3 & g_1 & g_1 & g_3 & g_3 & G_0 & G_0 & g_{d0} & g_3 & g_3 & g_3 \\ g_3 & g_1 & g_3 & g_3 & g_3 & g_1 & g_2 & g_3 & g_3 & g_{d0} & G_0 & G_0 \\ g_3 & g_1 & g_3 & g_3 & g_3 & g_1 & g_2 & g_3 & g_3 & G_0 & g_{d0} & G_0 \\ g_3 & g_1 & g_3 & g_3 & g_3 & g_1 & g_2 & g_3 & g_3 & G_0 & G_0 & g_{d0} \end{bmatrix} \quad (34)$$

2.5 Isotropic plasticity scheme

The rate-dependent law, developed by Maresca et al. [143], is employed to calculate the equivalent plastic strain rate pertaining to the isotropic portion of the model. For this purpose, the equivalent reference strain rate $\dot{\epsilon}_0$ in the model is calculated using the following equation:

$$\dot{\epsilon} = \dot{\epsilon}_0 \left(\frac{\sigma}{\sigma_c} \right)^{\frac{1}{m}} \quad (35)$$

Where $\dot{\epsilon}_0$, m , σ and σ_c correspond to reference strain rate (1/s), strain rate sensitivity, equivalent stress (Pa), and yield stress (Pa) parameters.

The equation that accounts for the influence of the rate of change on the yield stress, σ_y , over time is as follows:

$$\dot{\sigma}_c = h \dot{\epsilon} \quad (36)$$

Hardening modulus h_b in J2 isotropic plasticity scheme is applied using Equation (37) starting with an initial yield stress of σ_0 at time $t=0$.

$$h_b = h_0 \left(1 - \frac{\sigma_c}{S_m} \right)^{a_m} \quad (37)$$

Where h_0 , S_m , and a_m are material-specific constants. S_m , and a_m specifically determine the yield stress saturation.

2.6 Ferrite and martensite mechanical properties

The crystal plasticity theory, which was introduced in Section 2.1 to 2.4, outlines how ferrite's hardening characteristics are influenced by the rate of change. As a result, it is necessary to acquire the material parameters specific to ferrite that are linked to this theory.

Ferrite is presumed to exhibit cubic anisotropy, as indicated by the non-zero components of the elasticity tensor outlined in [144]:

$$C_{11} = 239260 - \frac{24520}{\exp\left(\frac{392.3}{T}\right) - 1} \text{ MPa} \quad (38)$$

$$C_{12} = 135780 - \frac{6550}{\exp\left(\frac{469.65}{T}\right) - 1} \text{ MPa} \quad (39)$$

$$C_{44} = 120720 - \frac{3500}{\exp\left(\frac{162.9}{T}\right) - 1} \text{ MPa} \quad (40)$$

Equations (38) to (40) demonstrate how the cubic elastic constants, C_{11} , C_{12} , and C_{44} , for alpha-iron vary with temperature T . Using the elastic constants obtained from equations (41) and (42), the shear modulus G and Poisson's ratio ν are computed.

$$G = \sqrt{(C_{11} - C_{12})C_{44}/2} \quad (41)$$

$$\nu = \frac{C_{12}}{C_{11} + C_{12}} \quad (42)$$

Table 2-2 presents the material constants related to single crystal alpha-iron, while Table 2-3 presents the model constants utilized in the calibration process along with their corresponding descriptions.

Table 2-2 Details and description of constants that are calibrated in the simulations for different experimental findings

Constant	Value	Unit	Description
b	2.86×10^{-10}	m	Burgers vector [145]
K_B	1.38×10^{-23}	JK^{-1}	Boltzmann constant [138]
χ	80×10^{-3}	Jm^{-2}	Stacking fault energy [146]
ρ	7890	kgm^{-3}	Mass density
Q_v	197×10^3	$J \text{ mole}^{-1}$	Volume (lattice) diffusion energy [139]
Q_c	117×10^3	$J \text{ mole}^{-1}$	Core diffusion energy [139]
D_{v0}	2×10^{-5}	m^2s^{-1}	Volume (lattice) diffusion constant [139]

$a_c D_{c0}$	1.0×10^{-24}	$m^4 s^{-1}$	Core diffusion constant [139]
A	5×10^{-4}	—	Cross-slip constant-1
B	0.07	—	Cross-slip constant-2
v_0	10^{10}	s^{-1}	Attack frequency of dislocations [71]
q	2	—	Number of atoms per unit cell
G_0	1	—	Co-directional interaction coefficient [142]
g_1	1.5	—	Co-planar interaction [142]
g_2	3.2	—	Lomer–Cottrell interaction [142]
g_3	2.4	—	Glissile junction [142]
g_4	1.8	—	Hirth-lock interaction [142]

Table 2-3 description and values of constants

Constant	Unit	Description
c_1	-	Geometric constant for the long-range flow stress
c_2	-	Jump distance constant
c_3	-	Adjusting parameter for activation volume
c_4	-	Mean-free-path hardening constant
c_5	-	Climb distance factor
c_6	-	Cross-slip dislocation width constant
c_7	-	Scaling constant for mobile dislocation density
c_8	-	Scaling constant for dipole storage

a_1	-	Non-Schmid parameter (1st term)
a_2	-	Non-Schmid parameter (2nd term)
a_3	-	Non-Schmid parameter (3rd term)
ρ^0	m^{-2}	Initial dislocation density
Q_{Pei}	J	Peierls energy barrier

The isotropic elastic constants of the martensite laths are taken from studies of Maresca et al. [143, 147, 148] based on indentation experiment, to align the stress-strain response of martensite lathes with that of the result reported in Bower et al. [44]. The parameters applied for isotropic plasticity model of martensite are listed in Table 2-4.

Table 2-4 Material parameters utilized for the martensite lath model with isotropic characteristics.

Constant	Unit	Value	Description
G	GPa	76.00	Shear modulus
K	GPa	166.66	Bulk modulus
σ_c	GPa	1.14	Initial isotropic yield stress
S_m	GPa	3.00	Yield stress saturation
h_0	GPa	20.0	Initial hardening rate
$\dot{\epsilon}_0$	-	0.01	Reference strain rate
m	-	0.10	Strain rate sensitivity
m	-	1.50	Hardening exponent

The mechanical responses of each individual phase, as well as the dual-phase material, with applying the provided mechanical properties of the constituent phases in the simulation cases, are illustrated in Figure 2-2.

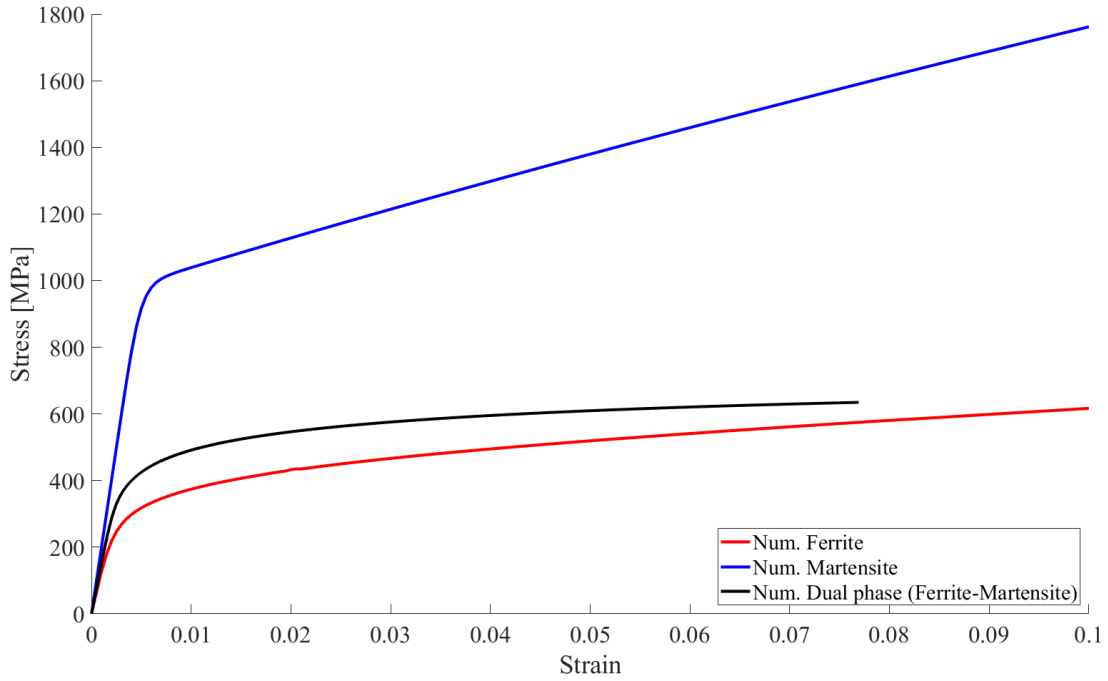


Figure 2-2 Separate the mechanical responses of the constituent phases, ferrite and martensite, as well as the dual-phase material.

2.7 Numerical modeling in 1st phase

Figure 2-3 illustrates the use of a physical model to conduct simulations at three different levels: MP, SE, and ETG models.

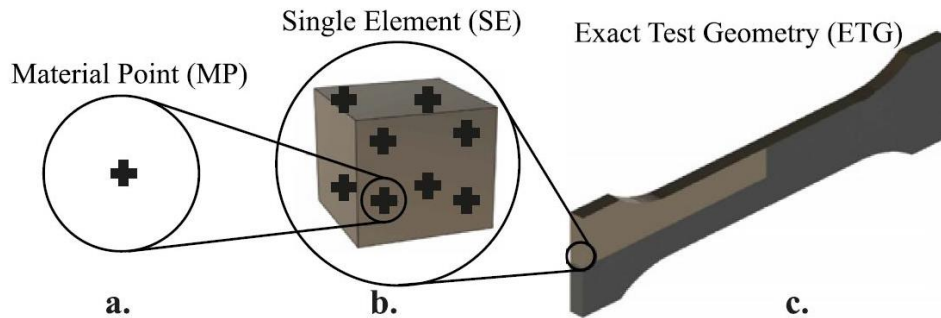


Figure 2-3 Modeling schemes at different size scales: (a) MP model, (b) SE model, and (c) ETG.

Simulations for uniaxial axisymmetric deformation gradient equation (43) are conducted using the MP model generated in MATLAB ®.

$$\mathbf{F} = \begin{bmatrix} \exp(\dot{\epsilon}_0 t) & 0 & 0 \\ 0 & \exp(-0.5\dot{\epsilon}_0 t) & 0 \\ 0 & 0 & \exp(-0.5\dot{\epsilon}_0 t) \end{bmatrix} \quad (43)$$

Equation (44) is utilized to determine the initial crystal orientation, $\mathbf{g}$, by assuming a transverse direction $[uvw]$ perpendicular to the axial loading (tension or compression) direction (hkl) . The values for (hkl) and $[uvw]$ are computed as well.

$$\mathbf{g} = \begin{bmatrix} h & p & u \\ k & q & v \\ l & r & w \end{bmatrix} \quad (44)$$

To establish general material rules, the constitutive relations for dislocation density based on crystal are integrated into a commercial finite element software (MSCMARC R) via its user-subroutine (HYPELA2). The finite element model is then utilized to perform simulations for two distinct models, SE and ETG, with the latter representing one-quarter of an ETG sample in accordance with EN ISO 6892-1 standard. The ETG model comprises 232 finite elements.

The SE and ETG finite element models share similar characteristics, including the use of hexahedral linear elements with eight nodes per element. In order to achieve a uniaxial strain state throughout the sample (as shown in Figure 2-4), specific boundary conditions are implemented. To determine the overall engineering stress, the summation of all reaction forces along the axial direction (x-direction) at the gauge cross-section of the sample is divided by the initial undeformed area. Similarly, strains are calculated by normalizing the given displacements by gauge length.

In order to ensure that the results of our simulations are consistent with both the experimental and numerical references, calibrations were conducted. First, the impact of each multiplier on the stress-strain curves was assessed through trial runs. Several trial runs were then conducted using different values as the calibrators while keeping the parameters within reasonable and specific ranges to prevent drastic deviations among case studies. Finally, the sets of values that provided the best matches between the simulation and reference results were chosen and reported.

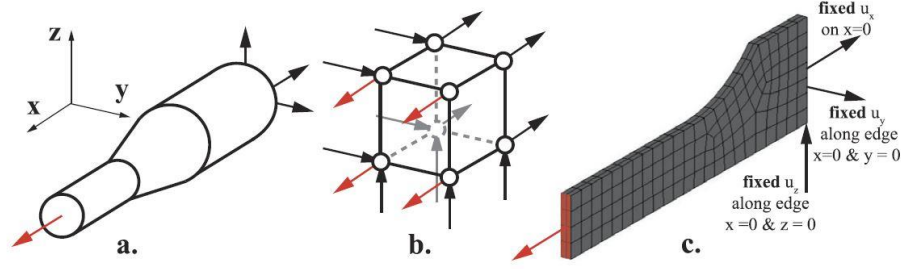


Figure 2-4 The specified and fixed displacements in the boundary conditions for the MP model (a), SE model (b), and ETG model (c) are represented by red and black arrows, respectively.

2.8 Application of the material microstructure as model input data

The electron backscatter diffraction (EBSD) technique is utilized to experimentally analyze the phase distribution, grain size, and crystallographic texture of DP600 steel. With this technique, it is possible to accurately and quantitatively confirm the existence of austenite, ferrite, bainite, and martensite phases. The Zeiss Merlin scanning electron microscope, which is equipped with an EDAX/TSL EBSD system and a Digiview EBSD camera, is used for orientation microscopy. This technique is utilized to determine the relative texture of the co-existing phases. During the EBSD scan, the acceleration voltage was adjusted to 15 kV and the working distance was set to 14 mm. A hexagonal grid with a step size of 100 nm was used to generate maps. The EBSD camera was operated at 100 frames per second with a 10 ms exposure time in 4x4 binning mode (160x 120 pixels). To create the Kikuchi patterns, 2 frames were captured and averaged for each point during the EBSD scan. A square region with dimensions of $100\mu\text{m} \times 100\mu\text{m}$ was examined during the analysis. The EBSD map of the DP600 steel, which displays the analyzed region, is shown in Figure 2-5.

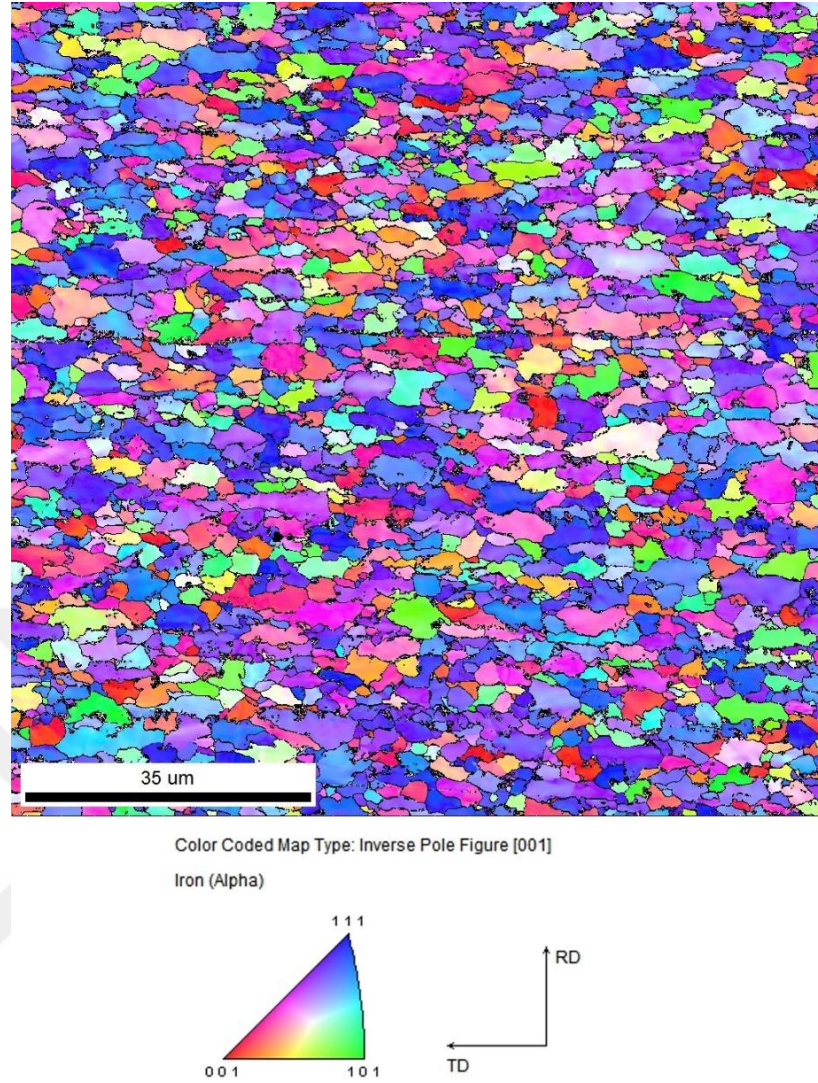


Figure 2-5 EBSD orientation map for the DP600 material on the Rolling direction (RD)-TD plane, IPF RD. Totally 1157156 data points recorded in a $100\mu\text{m} \times 100\mu\text{m}$ square. Data points are logged in the form of hexagons. For each of the grains corresponding Euler angles (ϕ_1 , Φ and ϕ_2) are recorded that will be used in generating the RVE for the simulations.

Experimental testing was conducted on the DP600 alloy to obtain its flow curve along the rolling direction (RD) using a uniaxial tensile test. The specimen was secured on both ends and stretched at a steady strain rate of 0.00025 s^{-1} , which is nearly equivalent to a quasi-static state. Figure 2-6 displays the results of the tensile test curve acquired from this investigation. The information presented in the diagram is utilized to adjust the parameters of the crystal plasticity hardening and deformation behavior model. These parameters include cutting stress, jump distance, activation volume, mean free path, climb distance, cross-slip width, as well as scaling of dipole formation constants, which are empirical parameters used in the simulations.

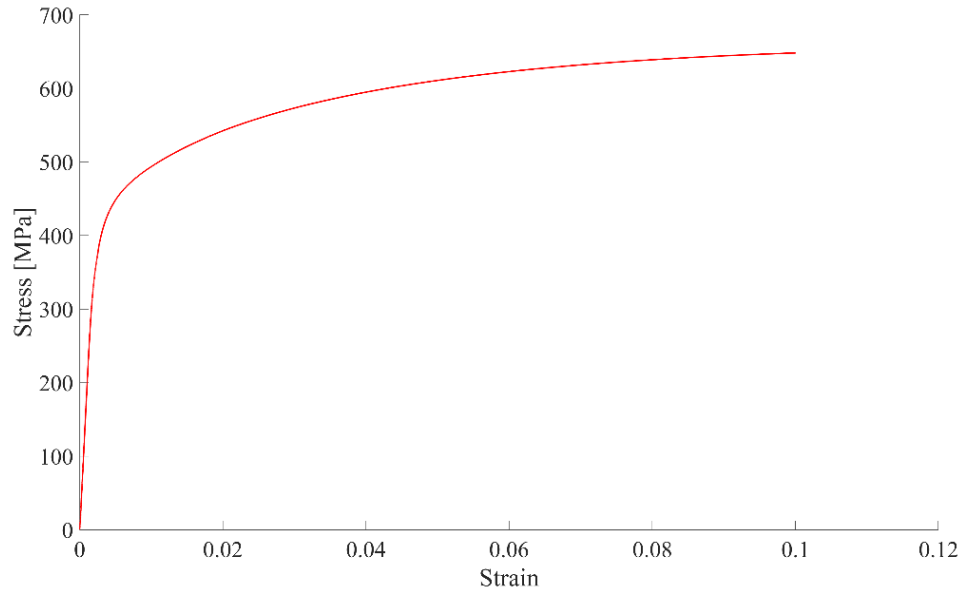
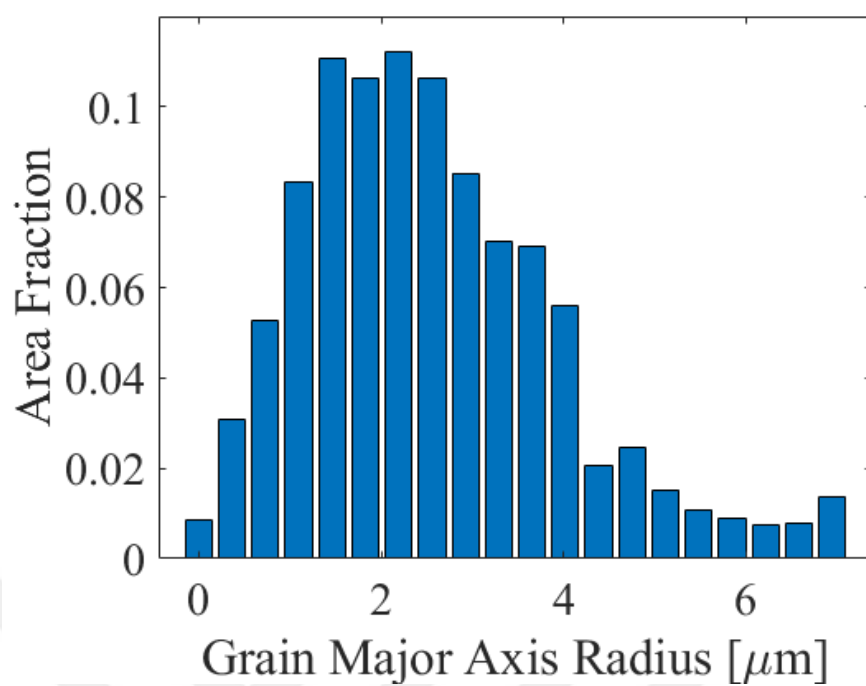


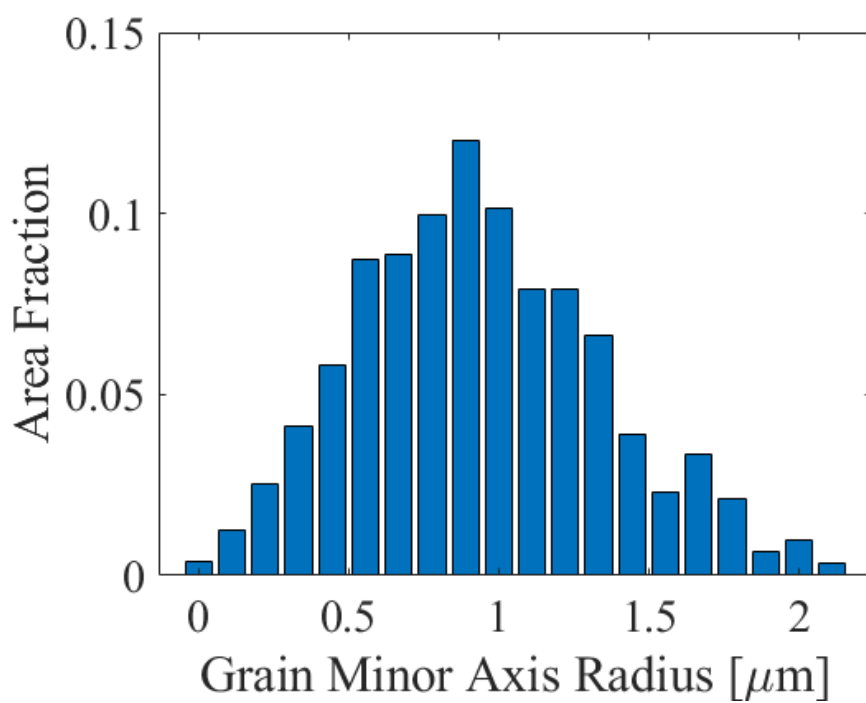
Figure 2-6 Experimental uniaxial tensile test result for DP600 steel sample obtained at a strain rate of 0.00025 s^{-1} . The specimen is tested under 0° inclination. There are 4 tests performed on 4 samples and the average values of these tests are given above.

2.9 Numerical modeling in 2nd phase

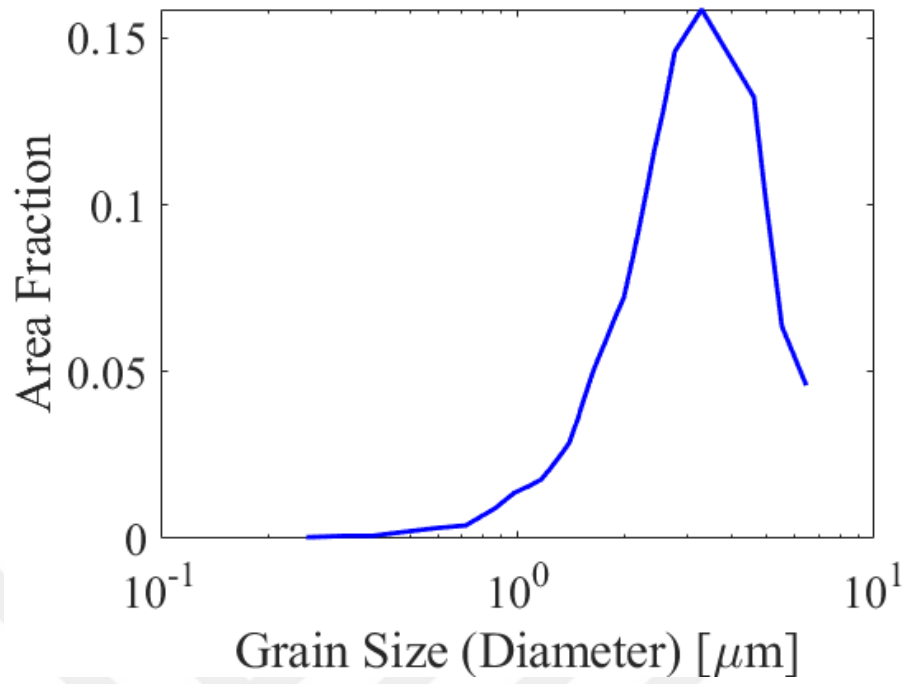
To generate representative microstructure models of the DP material [149, 150], we utilized the MTEX MATLAB script. The EBSD map was created based on the size of the scanned area and the resolution step size of the data, resulting in 1157156 data points. For each data point, the phase and crystal orientation were measured and assigned. The average grain size difference between martensite and ferrite phases in our DP600 steel is considerably high as can be realized by comparing the grain size distribution histograms of the alpha phase to the resized Kikuchi image quality plot as shown in Figure 2-7. According to the histograms the minor and major axis sizes of the grains in α phase are significantly larger than the martensite grains dimensions.



(a)



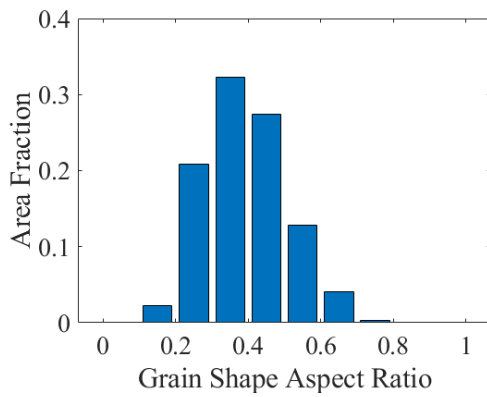
(b)



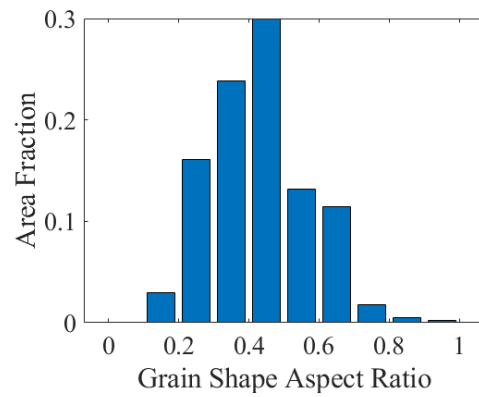
(c)

Figure 2-7 grain size distribution of α phase, (a) major axis length, (b) minor axis length and (c) area fraction distribution

Martensite and ferrite grains crystallographic characteristics including shape aspect ratio, major axis orientation and grain orientation experimentally measured through EBSD technique are provided in Figure 2-8.



(a)



(b)

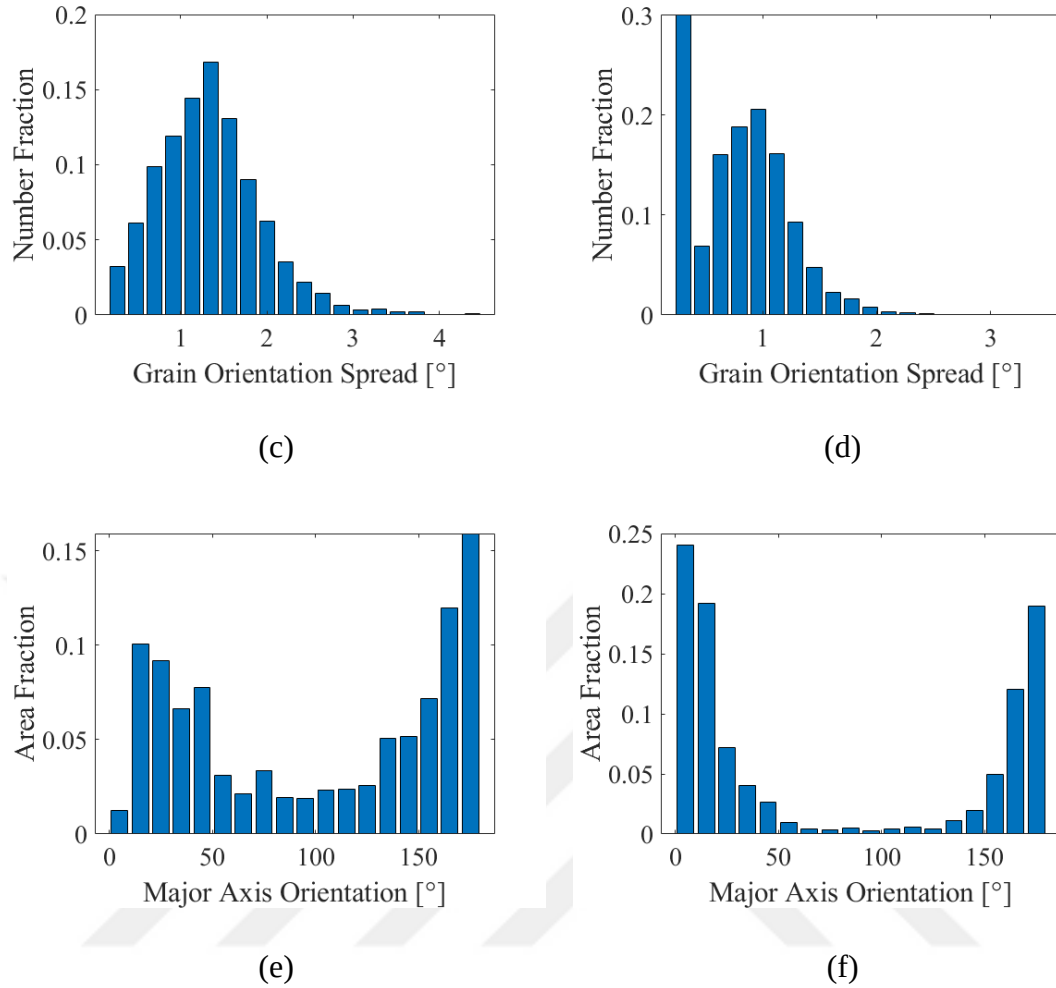


Figure 2-8 crystallographic state of the phases as per EBSD measurement results (a) grain shape aspect ratio ferrite, (b) grain shape aspect ratio martensite., (c) grain orientation spread ferrite, (d) grain orientation spread martensite., (e) major axis orientation ferrite, (f) major axis orientation distribution martensite

In order to improve the computational efficiency of the simulation, the data points in the EBSD map were initially converted into square forms, as illustrated in Figure 2-10. The processed map was then scanned to locate the optimal representative region that will be used as the foundation for the simulations to construct the model.

To accomplish this task, the MTEX toolbox's standard embedded calcError function was utilized. Table 2-5 contains specific information regarding the selected area within the entire EBSD map. The models were created with taking IPF plots of the original material and the numerical models compatible. The IPF plots of the main EBSD map and created models are shown in Figure 2-9.

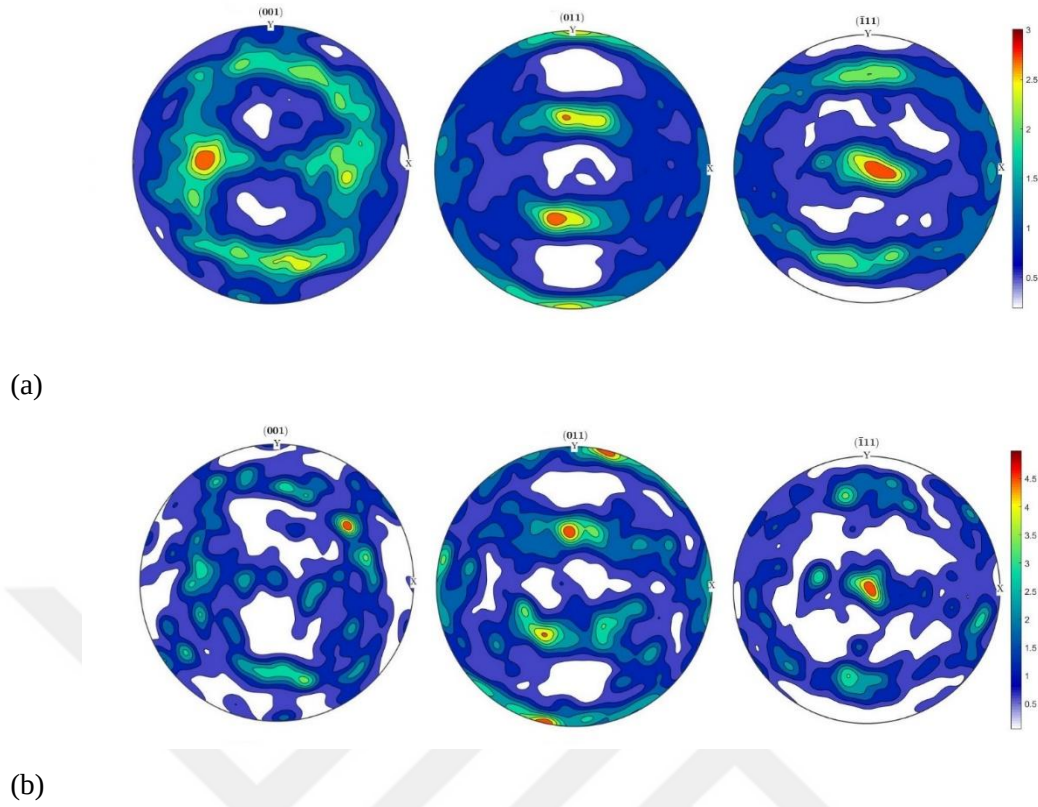


Figure 2-9 IPF plots in three pole directions $[0\ 0\ 1]$, $[0\ 1\ 1]$ and $[-1\ 1\ 1]$, (a) for original EBSD, (b) for the created RVE models

After the representative area is chosen, the numerical simulations are carried out by considering three key variables: the loading direction (which is applied as displacement boundary conditions on one end of the sample grid in the loading direction), RVE discretization, and phase distribution. Table 2-5 presents the implementation of the simulation for three distinct loading cases in order to comprehend the impact of boundary conditions on the resulting stress-strain distribution in the material. The size of the martensite particles is considered a crucial factor in the flow behavior of the DP steel models. Therefore, the mesh cell size in the model is examined as a variable to study this phenomenon. To understand the effect of the martensite phase on both the material as a whole and the ferrite matrix, simulations are performed on two material systems: the single ferrite phase and the multiphase (DP) material. This allows for a better understanding of the heterogeneous nature of the plastic deformations in the DP alloy in comparison to the SP ferrite state.

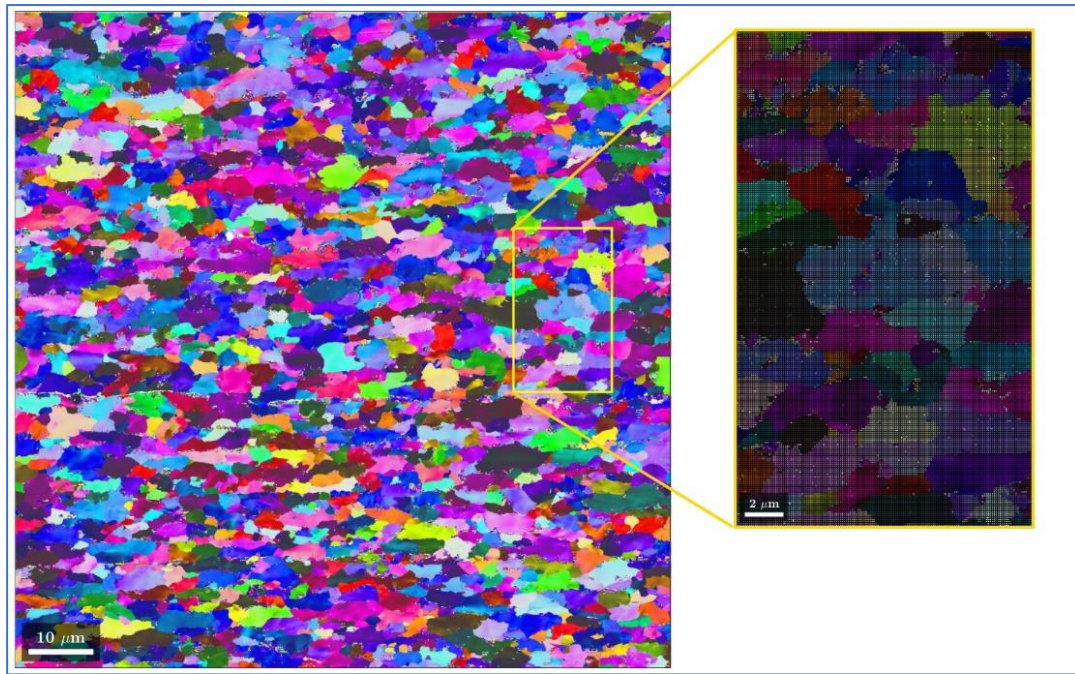


Figure 2-10 square meshed (converted) selected area

Table 2-5 Numerical simulation setting and microstructural features of the DP600 RVEs

RVE box size ($\mu m \times \mu m$)	25*25		
Grains distinction angle	0.1°		
Loading condition	Uniaxial Tensile	Uniaxial Compression	Simple shear
Grain size ($\mu m \times \mu m$)	0.247*0.25	0.373*0.373	0.49*0.49
RVE layer thickness (μm)	0.25	0.373	0.49
Phases	Ferrite	Ferrite and Martensite	

The CP analyses entail dividing each grain into cells (bricks) and assigning a distinct crystallographic orientation to each cell. This approach enables the prediction of the evolution of deformation texture and the possibility of stress-strain flow localization. The bricks with similar color spectrums exhibit almost identical crystallographic orientations with slight angular variations. MSC MARC, a commercial finite element software, is employed to generate the numerical models. To simulate the ferrite matrix, a crystal

plasticity CP constitutive law is used, which is based on the dislocation density that updates during deformation. The evolution of this dislocation density causes strain hardening at each loading increment. Conversely, an isotropic J2 plasticity model is responsible for the plastic deformation of the martensite islands. Since there are two types of models in terms of phases (Single-phase and dual-phase), two distinct HYPELA2 user subroutines are created to include the calculation schemes in the models. Two separate FORTRAN subroutines are written and utilized in the MSC MARC package to implement the numerical approaches. The microstructure models are named based on the constitutive model employed in the subroutine for the phases, which involves adding a suffix to the HYPELA2 term. In the case of the single ferrite phase, the suffix "-CP" is added, while in the composite ferrite-martensite phases, the suffix "-CPJ2" is added.

In the 2nd phase of the research study, the HYPELA2-CP model, which was developed by Joudivand and Demir [151] and explained in section 2.5, is used to simulate the homogeneous deformation of the ferrite matrix. It is important to note that the CP calibration parameters for the HEPLA2-CP scheme are determined through an iterative calibration process to ensure the best fit between the macroscopic stress-strain data. Additionally, the flow properties reported in Ref. [152] are utilized for the isotropic part in the HYPELA2-CPJ2 model. The HYPELA2-CPJ2 calculation method considers both the isotropic J2 plasticity and CP aspects to consider the martensite phase. Numerical simulations are conducted by applying three boundary conditions (BC) on the RVE's; uniaxial tensile, compression and simple shear. The BCs are displayed in Figure 2-11, and are explained in Table 2-6. In the case of uniaxial tension, the experimental stress-strain curve of the material is used for the CP parameters calibration, then the same values are used in other loading modes

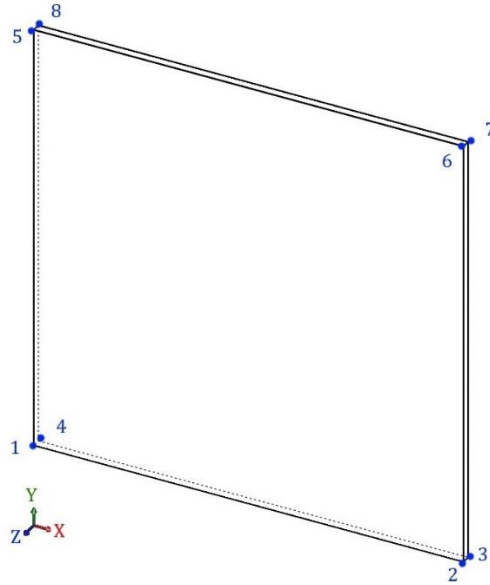


Figure 2-11 Nodes ID in the CPFEM model for assigning the boundary conditions.

Table 2-6 Description of the boundary conditions used in the simulations for various stress loadings.

Node No.	Tensile	Compression	Shear
1	<i>Fixed</i>	<i>Fixed</i>	<i>Fixed</i>
2	$U_x = +0.0025 \text{ 1/s}$	$U_x = -0.0025 \text{ 1/s}$	<i>Fixed</i>
3	$U_x = +0.0025 \text{ 1/s}$	$U_x = -0.0025 \text{ 1/s}$	<i>Fixed</i>
4	<i>Fixed</i>	<i>Fixed</i>	<i>Fixed</i>
5	<i>Fixed</i>	<i>Fixed</i>	$U_x = +0.0025 \text{ 1/s}$
6	$U_x = +0.0025 \text{ 1/s}$	$U_x = -0.0025 \text{ 1/s}$	$U_x = +0.0025 \text{ 1/s}$
7	$U_x = +0.0025 \text{ 1/s}$	$U_x = -0.0025 \text{ 1/s}$	$U_x = +0.0025 \text{ 1/s}$
8	<i>Fixed</i>	<i>Fixed</i>	$U_x = +0.0025 \text{ 1/s}$
<i>Fixed</i> means zero displacements, i. e. $U_x = 0, U_y = 0, U_z = 0$			

To account for the martensite phase in the simulations, the image quality (Kikuchi diffraction pattern quality) distributions were utilized to differentiate between ferrite and martensite in the EBSD map, which are both typically indexed as BCC regions. These distributions display a bimodal pattern, as demonstrated in Figure 2-13. Next, the martensite phase was included in the simulations by using the Kikuchi diffraction pattern

quality distributions to differentiate and assign ferrite and martensite in the EBSD diagram, which are typically identified as BCC regions. The RVEs were generated from the resulting martensite distribution by utilizing an image processing technique that maintains the overall martensite/ferrite fraction. Figure 2-12 illustrates the process of assigning the martensite phase to the grains at each discretization level.

After following the procedure described above, we are now focusing on the mechanical properties of the models used. For the BCC structure, the model had 24 slip systems, and the primary one selected was $\{110\} \langle 111 \rangle$. The CP calibration parameters previously outlined were utilized in both the HYPELA2-CP and HYPELA2-CPJ2 schemes. These parameters, which were responsible for generating results that were most consistent with the experimental stress-strain graphs, are listed in Table 2-3.

For clarity in the terminology used in this research study and in the simulation, it's important to distinguish elements from grains. In the models, ferrite grains consist of several elements with orientations that are close to each other. In contrast, martensite grains, being significantly smaller than ferrite grains, can be composed of either a single element or a few neighboring elements, depending on the texture and corresponding area fraction in the analysis. It can be inferred from the explanations that elements with similar color tones in the processed RVE belong to a specific grain.

It is important to note that throughout the manuscript, the terms 'cell' and 'element' both refer to the smallest cubic units within our simulation model resulting from the discretization and meshing process. The term 'grain' represents a group of elements (cells) that behave as an almost unidirectional entity within the simulation. 'Phase' is used to describe one or several grains that belong to the ferrite phase of the material. A 'region' represents either a single element or a group of elements forming a martensite island within the simulation.

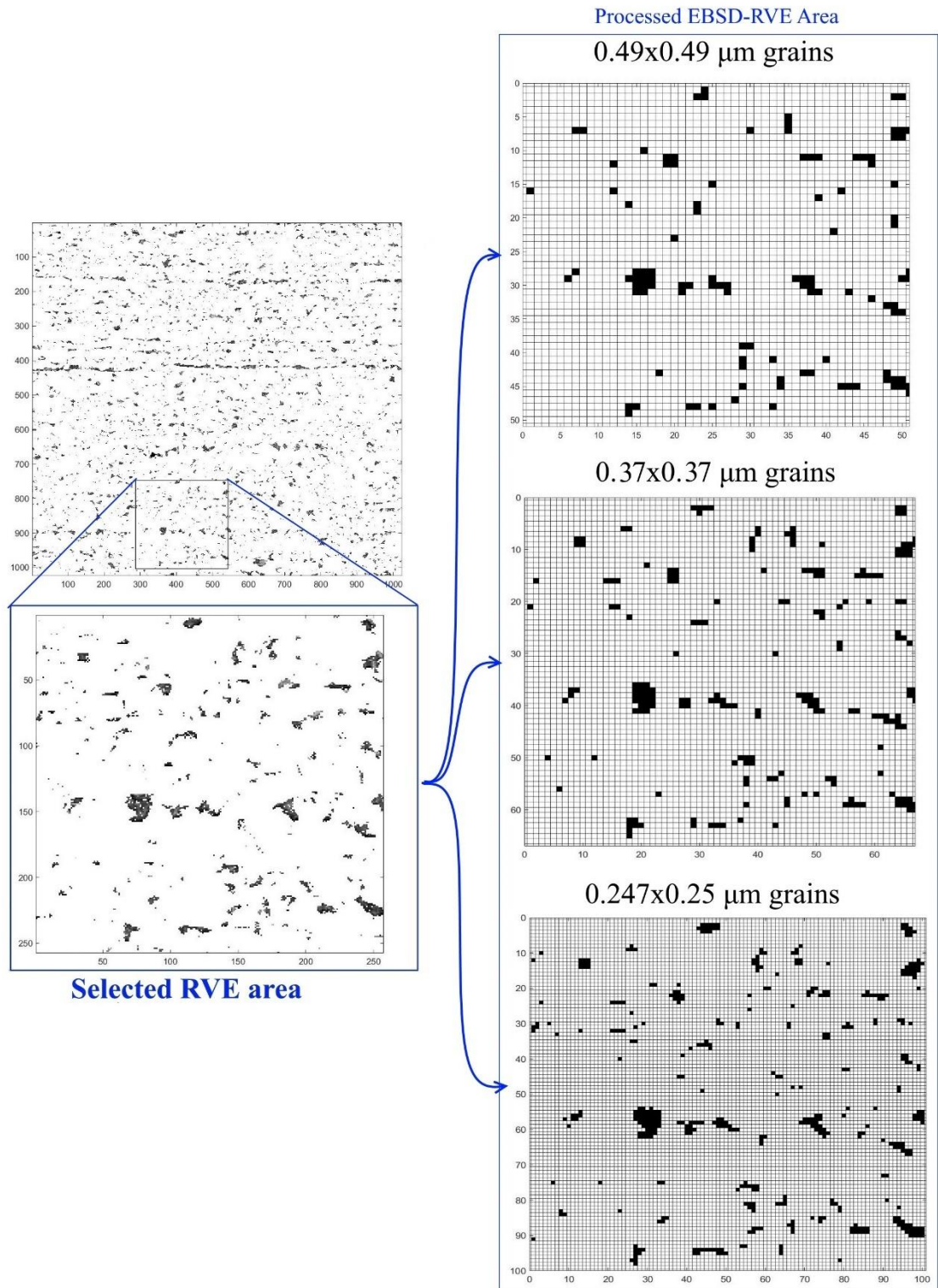
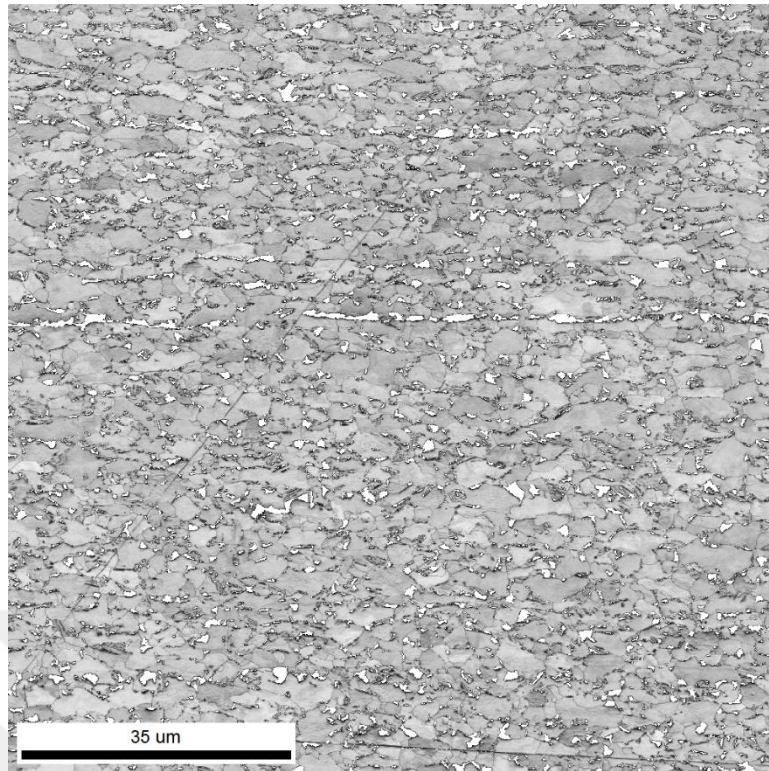
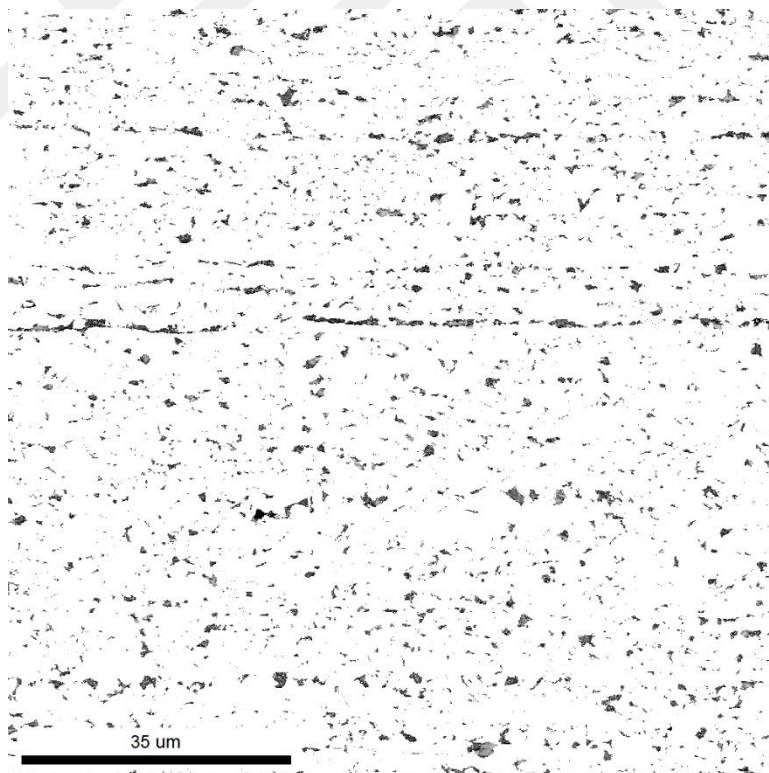


Figure 2-12 Schematic of the procedure for identifying the martensite phase in grains and its utilization in each of the discretization schemes using the image processing method outlined in the text.



(a)



(b)

Figure 2-13 Image quality map in RD-ND plane (a) only ferrite phase distribution with filtered martensite regions (white regions represent martensite phase), (b) only martensite phase distribution with filtered ferrite regions (white background represent ferrite matrix)

As mentioned earlier, in the simulation the very same RVE area is discretized into three different element sizes, $0.49\mu\text{m} \times 0.49\mu\text{m}$, $0.37\mu\text{m} \times 0.37\mu\text{m}$ and $0.25\mu\text{m} \times 0.25\mu\text{m}$. This suggests that as the dimensions of the elements increase, fewer elements come together to constitute a single ferrite grain. In Figure 2-14 two adjacent ferrite elements from each of the discretization size are picked and their corresponding Euler angle combinations are provided. It can be noted that as the size of the elements decreases, the slight angular deviation in the orientation of two adjacent elements belonging to the same grain also diminishes.

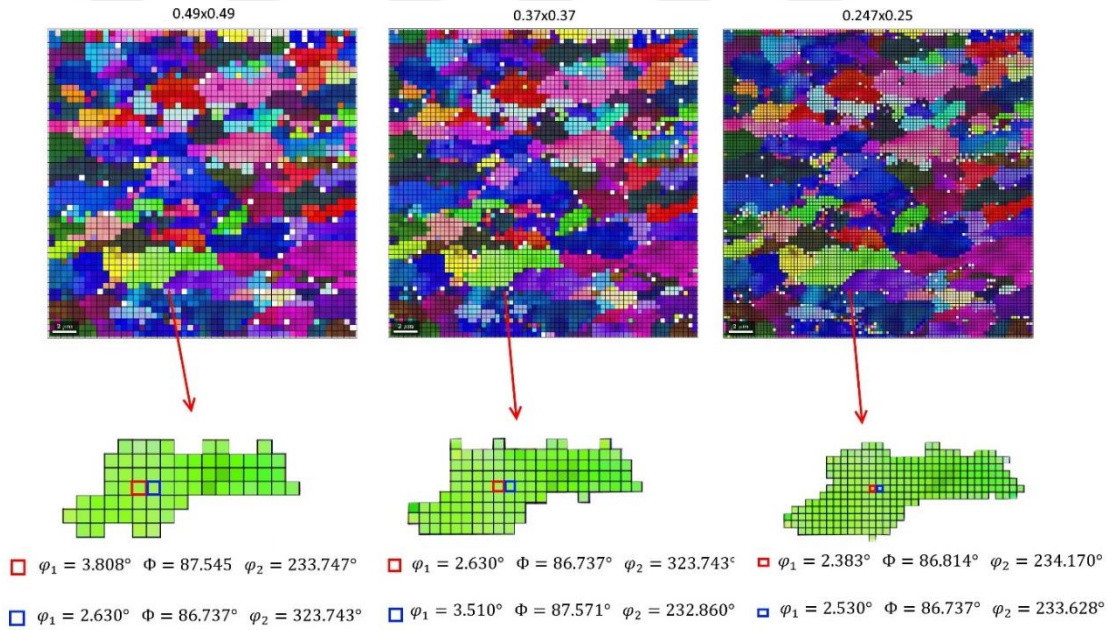


Figure 2-14 Euler angles difference between adjacent ferrite elements in three different discretization size

In Figure 2-15 black squares represent martensite grains distributed within the resized, $0.49\mu\text{m} \times 0.49\mu\text{m}$ alpha phase orientation distribution. For example, comparing green or purple grain sizes, both of which are ferrite grains, with the black martensite grains clearly indicate that there is a noticeable size disparity between the constituents in our DP test material. Given the scale of the simulations and the level of element refinement applied, it implies that the current research study focuses on the structural behavior of combined phases while considering the underlying fundamental physics.



Figure 2-15 martensite grains superimposed on the grain distribution in 51x51 discretization plot, black squares represent martensite, areas with similar color tones represent a single grain of ferrite

In order to confirm the model's accuracy and assess its agreement with experimental findings, this research study utilizes the developed modeling approach to numerically simulate strain distribution. This numerical simulation is then compared to the experiments conducted by Pelligra et al. as reported in their 2022 study [153]. Their data is limited to the distribution of ferrite and martensite phases. Consequently, our modeling process relies on a random orientation distribution to account for this. As indicated in Table 1 of the study by Pelligra et al. [153], the ratio of martensite to ferrite phase

fractions is 40.5%. This ratio is applied as a reference to ensure the resemblance between the simulated model and the experimentally tested sample is maintained. To construct the model, the phase distribution depicted in Figure 2(d) within Pelligra et al.'s publication is digitized through the implementation of a custom MATLAB image processing method. The input image is transformed into a 25×25 distribution, resulting in a total of 625 elements within the simulation. Among these, 258 elements are designated as martensite regions, indicating that the martensite volume fraction in the model is 41.28%. The RVE developing process is shown in Figure 2-16.

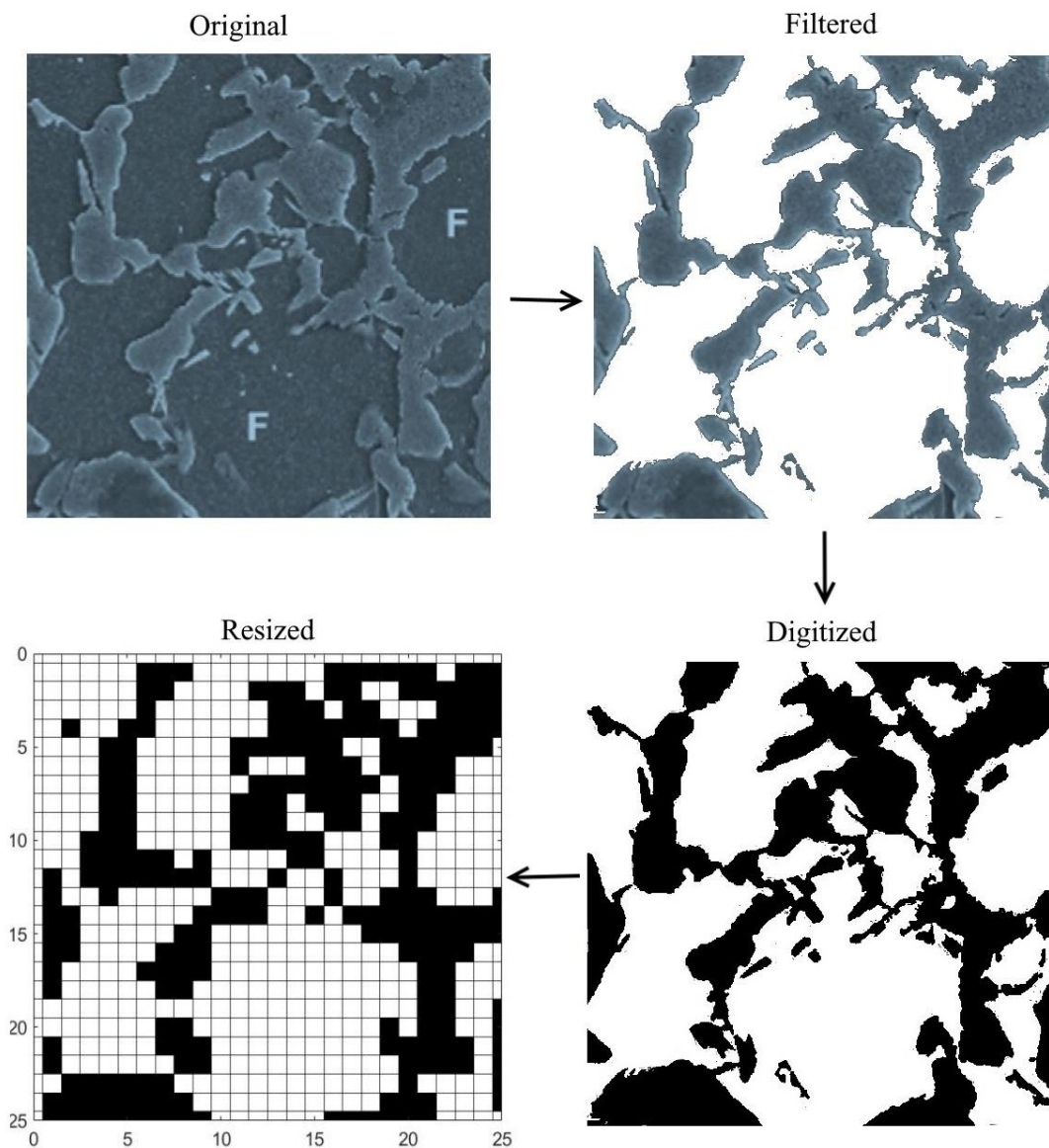


Figure 2-16 The procedure involves processing a screenshot extracted from Figure 2 in Pelligra et al. publication [153] to generate a 25×25 model. In this figure, black cells represent the martensite phase, while white cells depict the ferrite phase. During this process, the martensite-to-ferrite ratio is maintained at 40.5%.

The model is subjected to tensile test conditions, with the strain rate set at 0.0006 s^{-1} , mirroring the experimental conditions detailed in the paper. The chosen region from the sample is a square measuring $17.605 \text{ }\mu\text{m}$ on each side. In the simulation, the model experiences a 10% strain. The paper, on the other hand, has utilized a DIC test on the sample to report the average local strain variation, as seen in Figure C1. However, the FEM package we employ for the simulations, along with our subroutines, is MSC MARC. This software doesn't yield an average strain scalar as an output. Hence, we opt for the equivalent total strain component for comparison to demonstrate the alignment of the heterogeneity pattern. The modeling parameters are adjusted based on a typical stress-strain curve for DP1300 steel. The compression and shear mechanical responses obtained from the simulations are compared with the experimental findings as reported in the study conducted by Ghaei et al. [154].

2.10 Inhomogeneity Definition Tools

The distribution of martensite and the initial orientation of ferrite elements are influential factors in determining the degree of inhomogeneity in the results. In order to compare various conditions of martensite distribution and different cases of initial orientation distribution and to assess the level of inhomogeneity in the results, it is essential to establish quantified measures.

The initial tool is designed to create a technique for determining martensite distribution and assessing its concentration or dispersion state, referred to as the martensite compactness factor (MCF). With taking the martensite distribution layouts we have defined a quantitative parameter named MCF. We developed this factor specifically for our single-layer RVE to offer a quantified measure of the dispersion quality of martensite in the model. The main purpose of this is to attribute the model's phase morphology to the results. It is the root mean square distance of all the martensite elements distributed in our RVEs. The following equation is applied to calculate the MCF:

$$MCF = \sqrt{\frac{1}{n+1} \sum_{j=1}^{n+1} \left(\sqrt{\frac{1}{n} \sum_{i=1}^n d_i} \right)} \quad (45)$$

In the equation, d_i represents the distance of every martensite element from the other martensite elements (shown in) in the RVE. To achieve this, the root mean square of all martensite grains is computed relative to each other, as defined by the following equation.

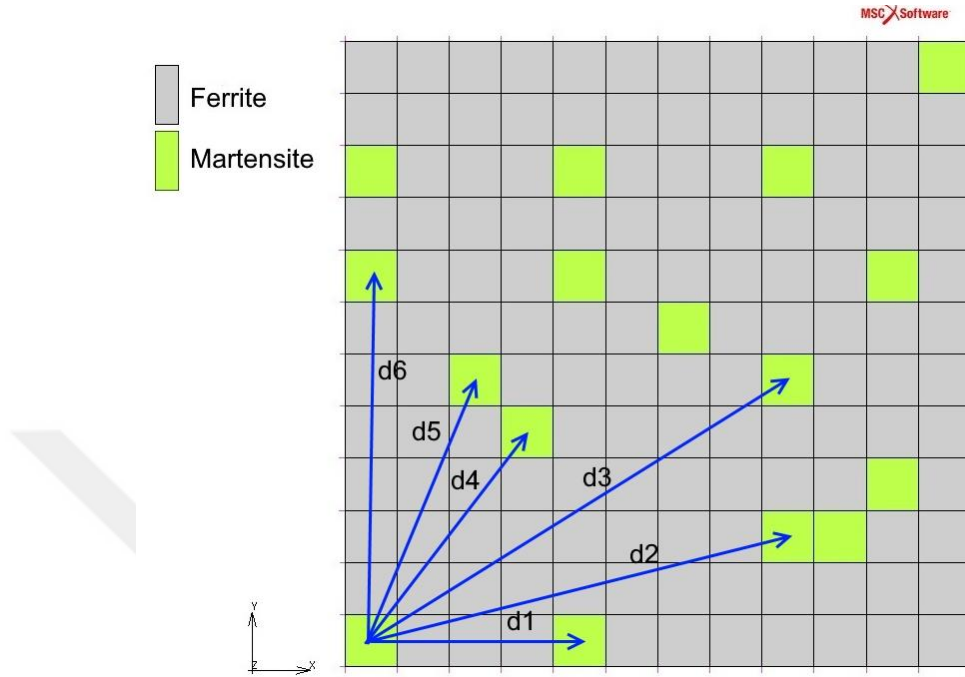


Figure 2-17 distances in the MCF calculation method

The Von Mises-Fisher concentration parameter is a statistical measure used in the context of orientation distribution analysis, specifically when dealing with Euler-Bunge angles to represent crystallographic orientations. This parameter quantifies the degree of concentration or dispersion in the orientation distribution of crystallographic elements, such as grains or phases in a polycrystalline material.

In the context of orientation distribution functions (ODF), the Euler-Bunge angles describe the orientation of crystals in three-dimensional space. These angles are commonly used to represent the rotation needed to transform a crystal from a standard reference orientation to its actual orientation. The ODF provides information about the probability distribution of crystallographic orientations within a polycrystalline material.

The Von Mises-Fisher distribution is a probability distribution on the sphere that is often used to model directional data. In the context of orientation distribution, it helps characterize the spread or concentration of crystallographic orientations. The concentration parameter (often denoted as κ) in the Von Mises-Fisher distribution measures how tightly the orientations cluster around a central axis.

The analytical expression for the Von Mises-Fisher distribution is given by the probability density function (pdf):

$$f(X|\mu, \kappa) = C(\kappa) \exp(\kappa \mu^T X) \quad (46)$$

where:

- X is the orientation vector (in Euler-Bunge angles, for instance).
- μ is the mean direction vector.
- κ is the concentration parameter.
- $C(\kappa)$ is a normalization constant.

In the context of orientation distribution analysis, the concentration parameter (κ) is crucial. A higher κ value indicates a more concentrated or focused distribution, while a lower κ value signifies a more dispersed distribution. The calculation of κ involves considering the Euler-Bunge angles and their relationship to the orientation vectors. The specific equations for calculating κ would depend on the precise representation of orientation.

In summary, the Von Mises-Fisher concentration parameter in orientation distribution analysis helps quantify the degree of concentration or dispersion of crystallographic orientations within a material, particularly when represented using Euler-Bunge angles. The parameter is valuable in characterizing the microstructure and mechanical properties of polycrystalline materials and we have applied it in our research study.

Eventually we need a tool to measure the inhomogeneity level in the plastic deformation distribution contours to compare and analyze the results with considering the fed parameters. The coefficient of variation is a dimensionless measure of relative variability. It is calculated as the ratio of the standard deviation to the mean. CV is a statistical measure used to assess the relative variability or inhomogeneity of a data set. It is expressed as the ratio of the standard deviation to the mean and is often represented as a percentage. In the context of measuring the inhomogeneity of a deformation distribution contour plot, the coefficient of variation provides insights into the uniformity or disparity of deformation across the analyzed region.

Let's break down the key concepts related to the coefficient of variation in the context of deformation distribution contour plots:

The coefficient of variation is defined as the ratio of the standard deviation Γ to the mean μ_{cv} of a dataset. Mathematically, it is expressed as:

$$CV = \left(\frac{\Gamma}{\mu_{cv}} \right) \times 100\% \quad (47)$$

where Γ is the standard deviation and μ_{cv} is the mean. A higher CV indicates greater relative variability in the dataset, suggesting higher inhomogeneity or non-uniformity. Conversely, a lower CV suggests more homogeneity or uniformity.

In the context of a deformation distribution contour plot, the CV can be applied to assess how consistent or variable the deformation is across the analyzed region. For example, if the CV is low, it implies that the deformation values are relatively close to the mean, indicating a more uniform deformation. On the other hand, a higher CV suggests greater variation in deformation, indicating potential inhomogeneity. In this study CV is a valuable metric for quantifying the inhomogeneity of a deformation distribution contour plot.

3 Results and discussions

3.1 Crystal plasticity analysis

Using the developed framework in Section 2, the results of the following investigations are provided in this section:

- The effect of orientation on single crystal model response
- The effect of temperature on single crystal model response
- The effect of strain-rate on single crystal model response
- Tension-compression asymmetry in single crystal model response
- The effect of martensite on strain and stress heterogeneity in the RVE
- Effect of martensite regions concentration on the RVE response

- The effect of boundary condition on the hardening evolution of RVE
- The effect of martensite on the misorientation in ferrite phase in the RVE
- The effect of texture on the RVE response

3.1.1 Single crystal orientation

For materials with a body-centered cubic (BCC) structure, the mechanical response is greatly influenced by the crystal orientation. Keh investigated the relationship between crystal orientation and the behavior of individual α -iron crystals at room temperature, studying strain levels of around 5%. [6]. The experimental conditions for the tests are presented in the corresponding reference [6], as provided in Table 3-1.

Table 3-1 The reference [6] employs experimental test cases to investigate the impact of crystal orientation on α -iron.

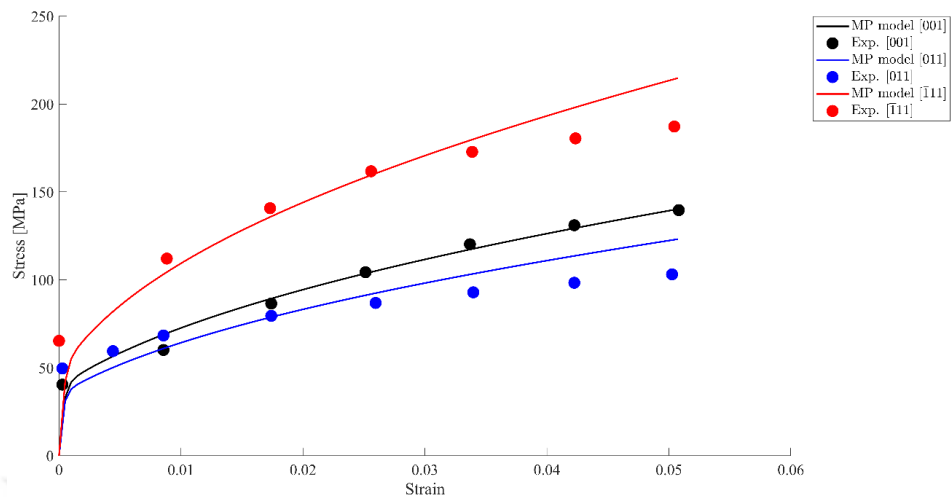
Temperature (K)	Strain rate (s^{-1})	Loading direction	Load type
298	3.3×10^{-4}	[001], [011], $[\bar{1}11]$	Tension

To achieve the most accurate matches with the experimental data, the proposed model was calibrated at various scales (MP, SE, and ETG), as demonstrated in Table 3-7. As utilized in reference [6], Figure 3-1 displays three stress-strain curves for three distinct crystallographic orientations: [001], [011], and $[\bar{1}11]$. As observed in the experimental results, the [011] direction exhibited the lowest strain hardening rate, while the highest strain hardening rate was observed for the $[\bar{1}11]$ tensile direction.

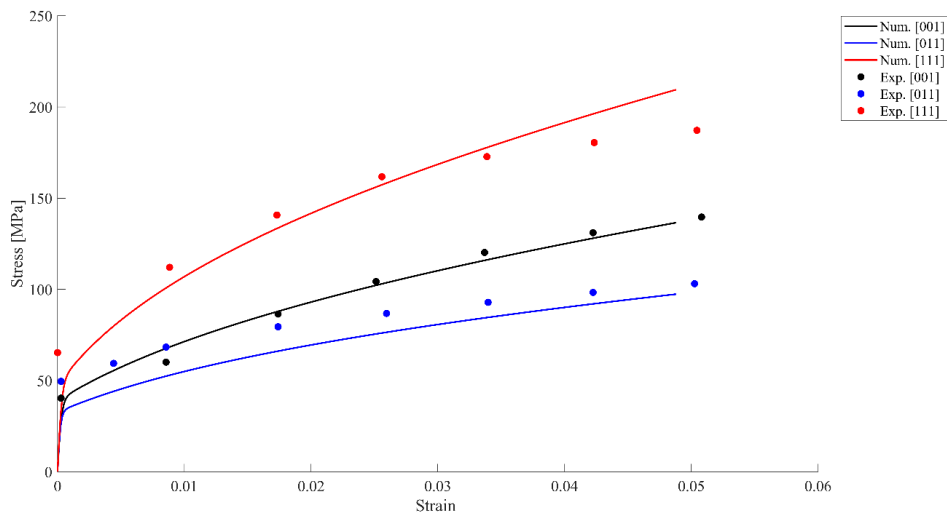
In order to reproduce the orientation dependence, the forest projection operator in equation (12) and the dislocation interaction matrix in equation (34) are the two significant factors utilized in the proposed model. Since the experiments were conducted at around room temperature [6], the impact of non-Schmid terms is anticipated to be relatively small. The anisotropy is attributed to the interactions between slip systems in strain hardening. In the proposed scheme for dislocation evolution or strain hardening, the forest projection operator plays a critical role.

After a meticulous calibration process, all of the graphs demonstrated a reasonable correspondence with the experimental outcomes. The parameters utilized to calibrate the MP, SE, and ETG models are provided in Table 3-7. In terms of hardening rate and yield

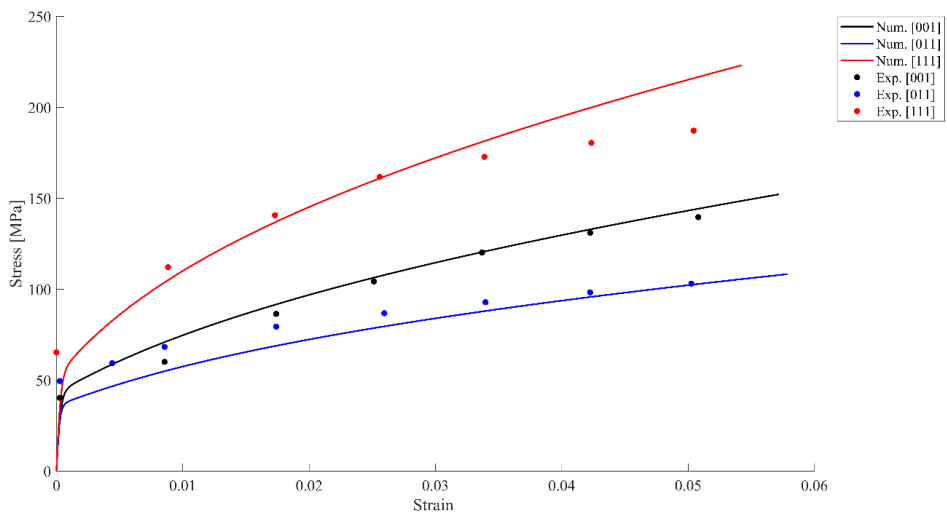
behavior, the MP model exhibited a slightly better ability to capture the experimental deformation pattern compared to the other model scales.



(a)



(b)



(c)

Figure 3-1 The stress-strain relationship for various crystal orientations under tensile loading is dependent on the model scale, as shown in (a) MP, (b) SE, and (c) ETG. The experimental data used in this analysis is from [5]

The reason for this is that there are inaccuracies in the finite element models due to numerical errors and the way the stress is calculated by averaging the nodes. To get accurate results, additional processing of the finite element models is required, making the calibration process more difficult. As a result, the MP model is preferred for modeling simple deformation states such as the uni-axial strain state in this case because it is easier to calibrate and provides faster responses. In Figure 3-2(a), the distribution of the maximum RSS is shown. Based on the contour plot and color bar, it can be seen that the $[\bar{1}11]$ crystal direction has the lowest Schmid factor compared to the $[001]$ and $[011]$ orientations. This means that if the crystal were loaded in tension along the $[\bar{1}11]$ direction, the flow stress would be the highest. However, for the other two orientations, there is little or no difference in the flow stress at room temperature, which was also observed in the experiments [6].

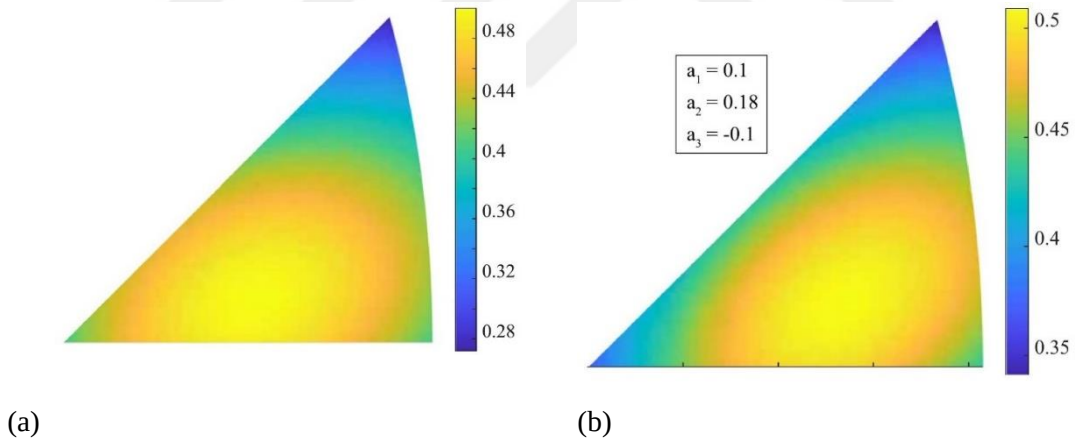


Figure 3-2 The maximum distribution of (a) resolved shear stress (RSS) ($\tau\alpha$) and (b) the cumulative RSS, including non-Schmid projections ($\tau\alpha + \sigma\alpha_{ns}$) for tensile loading, are displayed. The values for the non-Schmid constants are also provided.

The impact of non-Schmid effects on the yield magnitudes for different orientations is displayed in Figure 3-2(b), where non-Schmid factors of a_1 , a_2 , and a_3 with values of 0.1, 0.18, and -0.1 were employed for the uni-axial tensile loading scenario. These non-Schmid terms enable the yield magnitudes to differ for various orientations. In this instance, the yield is anticipated to transpire at a higher level of stress for the tensile

direction of [001], also known as the cube orientation, than for the $[\bar{1}11]$ or [001] orientations. This is because the maximum resolved cumulative shear factor is lower for the [001] direction compared to the [011] direction. Nonetheless, the influence of strain hardening must also be taken into consideration and cannot be disregarded.

The impact of interaction coefficients, which are similar to latent hardening coefficients, on the crystal orientation dependency of BCC materials was also explored. Various hardening coefficients were utilized instead of the Franciosi constants in Equation (34), resulting in a considerably better correlation, as illustrated in Figure 3-3. The initial yield of the cube orientation observed in the experimental data can also be reproduced by considering the proportional limit to determine the yield stresses.

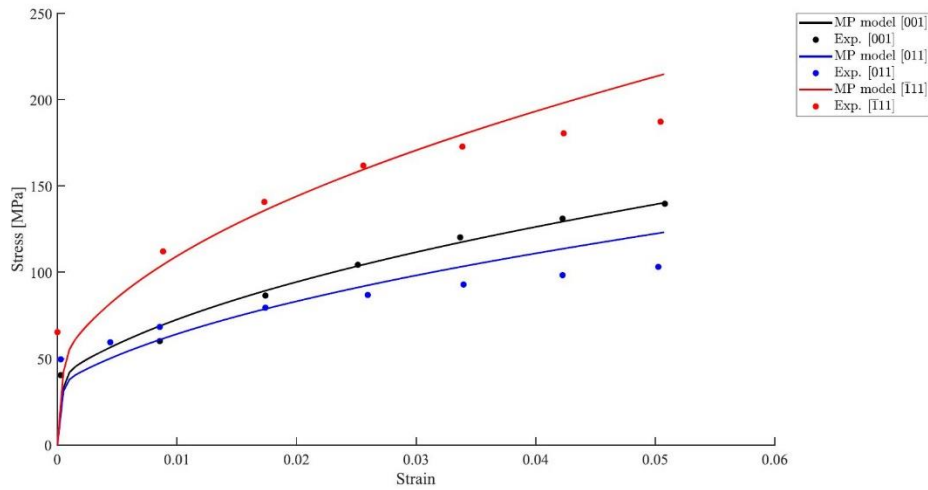


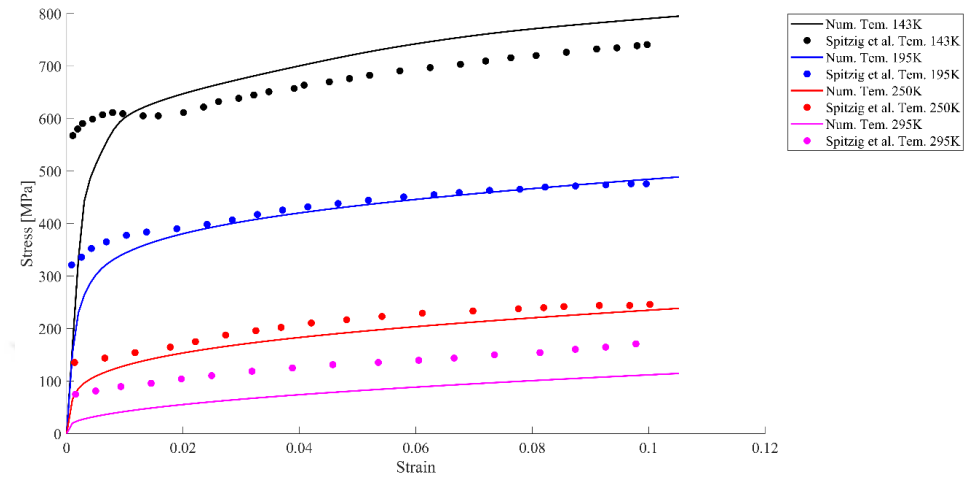
Figure 3-3 The impact of crystal orientation on α -iron, through the calibration of hardening interaction coefficients, is examined using experimental data obtained from [6]

3.1.2 Temperature

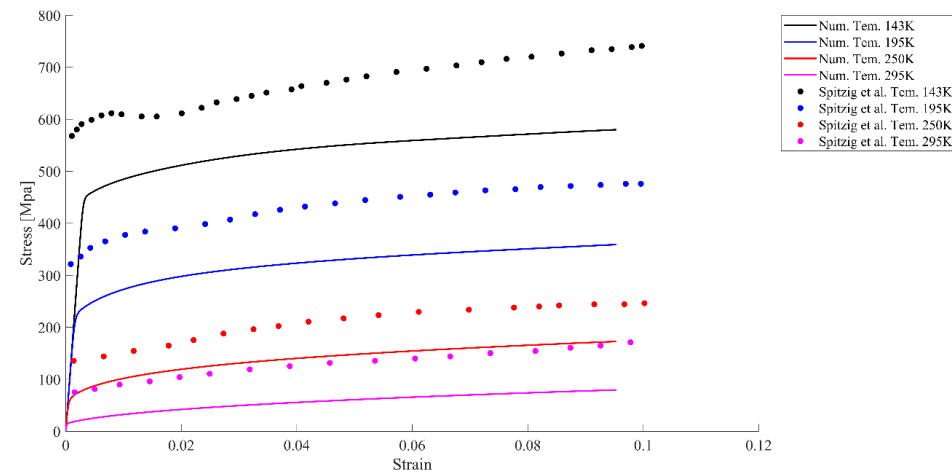
The proposed modeling scheme utilized Peierls activation energy barrier in equation (15) to account for the effect of temperature on the model. The validity of the model was tested separately for tension and compression scenarios using two different experimental observations. Spitzig and Keh conducted an experimental study to investigate the relationship between temperature and mechanical response of single crystals α -iron [155]. They conducted tensile experiments at various temperatures, namely 143 K, 195 K, 250 K, and 295 K. The test cases they performed are presented in Table 3-2.

Table 3-2 The reference [70] utilized experimental test cases to examine the temperature dependency of α -iron [155].

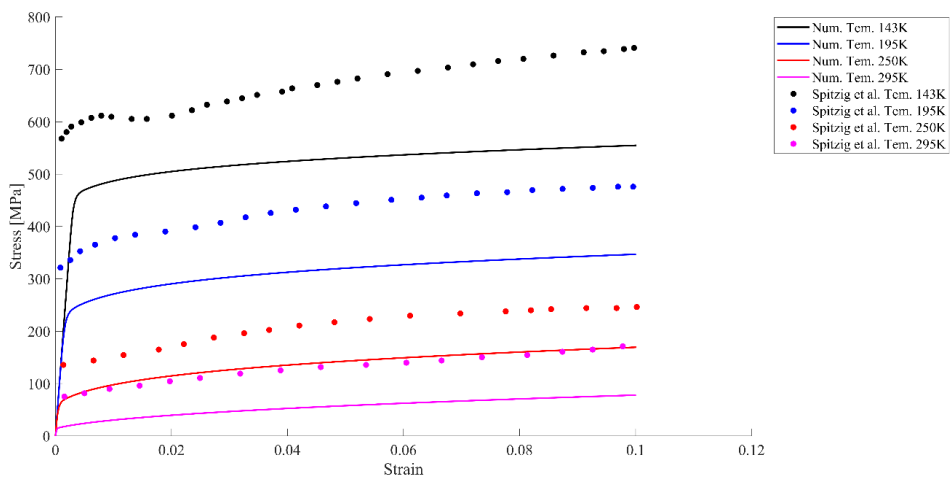
Temperature (K)	Strain rate (s ⁻¹)	Loading direction	Load type
143, 195, 250, 295	5.6×10^{-4}	[$\bar{1}49$]	Tension



(a)



(b)



(c)

Figure 3-4 The stress-strain graphs of MP, SE, and ETG models were analyzed in relation to temperature, and were compared to experimental data [155].

The model constants for various modeling scales (MP, SE, and ETG) were calibrated and presented in Table 3-7. The simulation results and experimental findings [155] are compared in Figure 3-4, and a relatively good agreement was observed between the two. The correlation between the model and experimental curves is excellent across all scales of the model, demonstrating that as temperature decreases, mechanical strength increases as anticipated. The physics of plastic flow in BCC metals is governed by the concept of thermal activation overcoming the Peierls barrier, which is why the mechanical response (flow stress) correlates well with temperature. The impact of temperature on the yield stress of individual α -iron crystals was studied and compared to the same reference [155]. Tensile tests were carried out on the crystals at four different temperature levels, and they were loaded along three distinct crystallographic directions during the experiments.

Figure 3-5 displays a comparison of the yield stress findings obtained through our MP model, as well as experimental tests documented by Spitzig and Keh [155], and a numerical investigation conducted by Patra and McDowell [79].

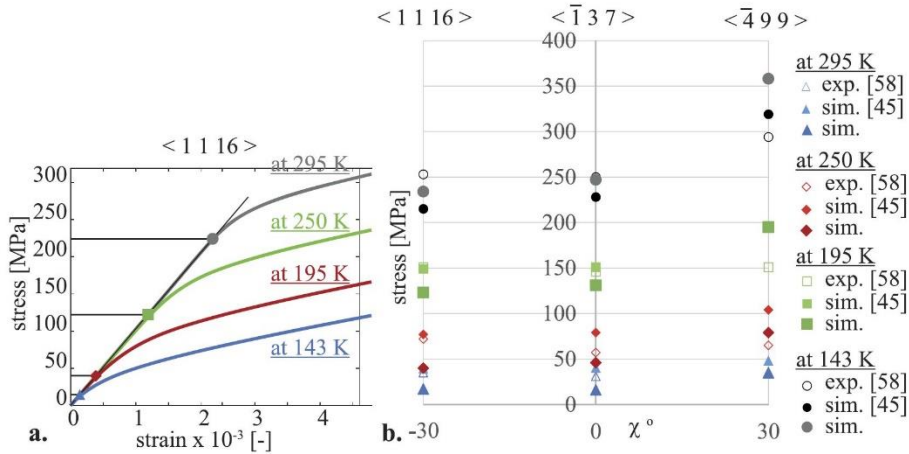


Figure 3-5 The relationship between flow stress, temperature, and crystal orientation is depicted as follows: (a) the computation of flow stress for the $[1\ 1\ 16]$ tensile direction, (b) the results of flow stress for three distinct tensile directions of $[1\ 1\ 16]$, $[\bar{1}\ 1\ 3\ 7]$, $[\bar{4}\ 9\ 9]$ and at four different temperatures of 143 K, 195 K, 250 K, and 295 K. χ refers to the deviation from the single slip orientation, $[\bar{1}\ 1\ 3\ 7]$.

There is a difference between the calculated yield stresses and the experimental results. However, our proposed model at the MP scale is consistent with another model in the literature [79] that considers the temperature effects using a particular approach

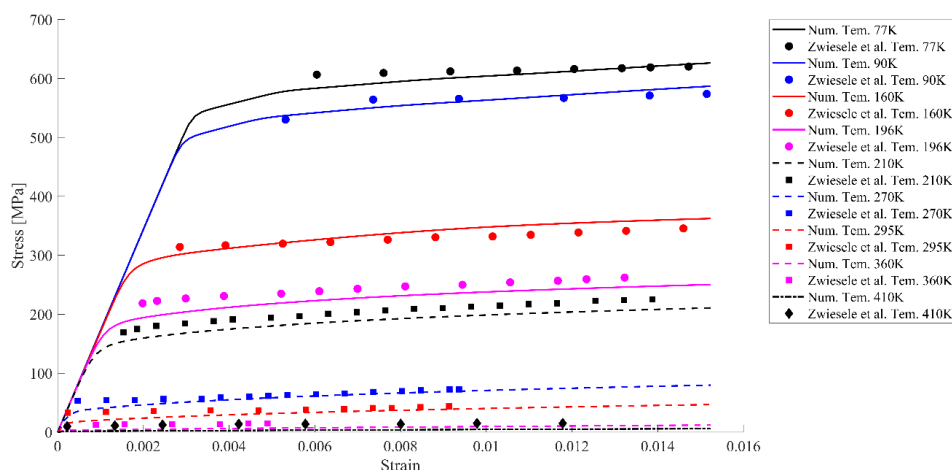
that features non-Schmid behavior. The proposed modeling approach overestimated the experimental yield stresses for the tensile direction of $[\bar{4} 9 9]$, but underestimated them for the tensile direction of $[1 1 16]$ at all temperatures. The main discrepancy is due to the utilization of identical constants for non-Schmid effects at varying temperatures. It is a well-established fact that non-Schmid effects diminish as temperature rises. Therefore, at room temperature, the flow stress is overestimated, leading to a discrepancy from the experimental results.

Zwiese and Diehl [156] carried out compression experiments on individual α -iron crystals at varying temperatures, with the experimental conditions outlined in Table 3-3. Calibration was conducted, and the model constants were derived, as displayed in Table 3-7.

Table 3-3 The reference [156] employed experimental test cases to investigate the temperature dependence of α -iron

Temperature (K)	Strain rate (s^{-1})	Loading direction	Load type
77, 90, 160, 196, 210, 270, 295, 360, 410	5.6×10^{-4}	$[\bar{0.266} 0.492 0.829]$	Compression

The outcomes for three distinct model scales, which were calibrated and then compared to the experimental stress-strain response, are depicted in Figure 3-6. The MP model scale exhibited a somewhat better correlation with the experimental strain hardening response. The proposed model was capable of reproducing the temperature-dependent yield stress and strain hardening response.



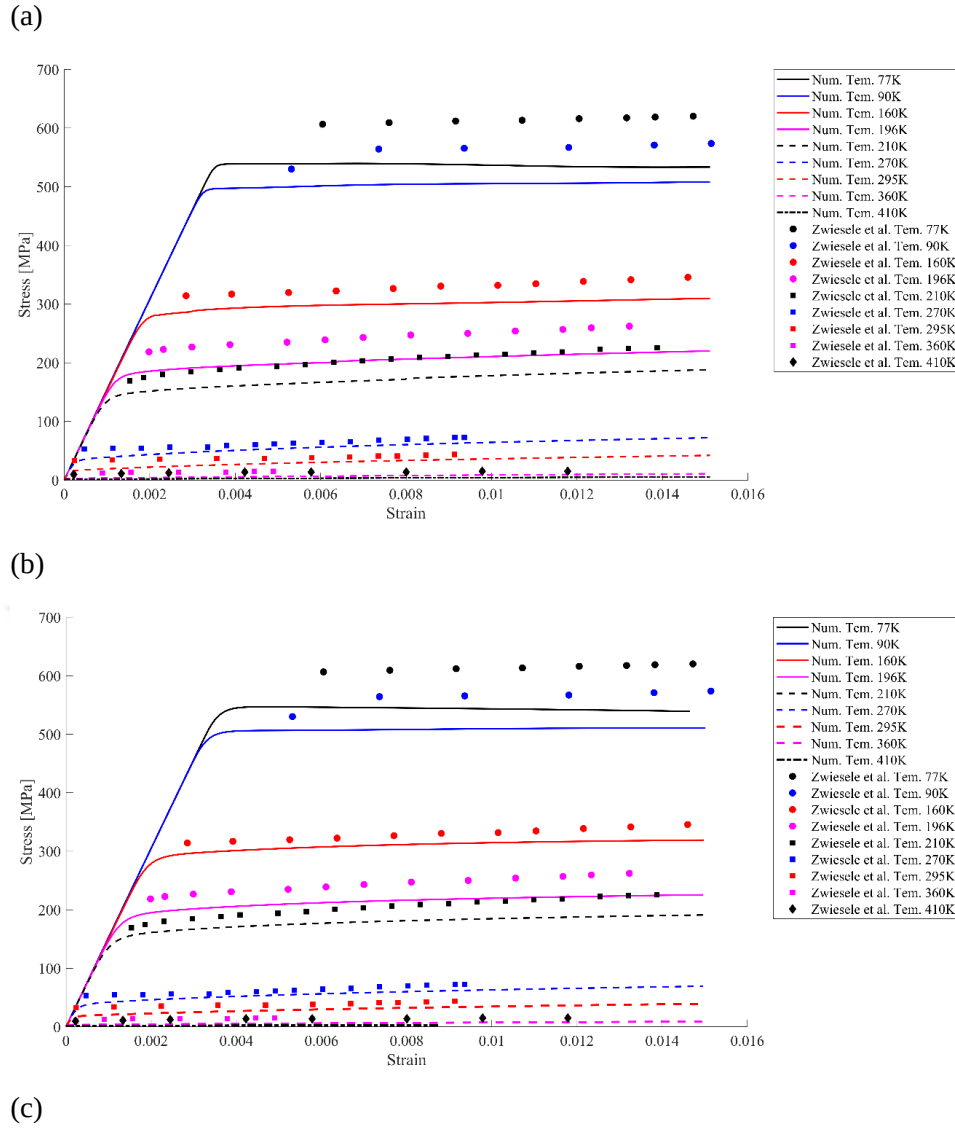


Figure 3-6 The influence of temperature on the stress-strain diagrams is demonstrated for the MP, SE, and ETG models in (a), (b), and (c) respectively, alongside a comparison to the experimental data [156]

3.1.3 Strain-rate

The suggested framework was employed to simulate how the mechanical reaction of an α -iron single crystal is affected by strain rate. The connection between stress and strain that is utilized in the suggested model is not straightforward. The Orowan law controls the slip-rate and is reliant on both the velocity of dislocations and mobile dislocations. The mobile dislocation density is determined using an effective stress function, while the velocity of dislocations is regulated by the leading exponential law presented in equation

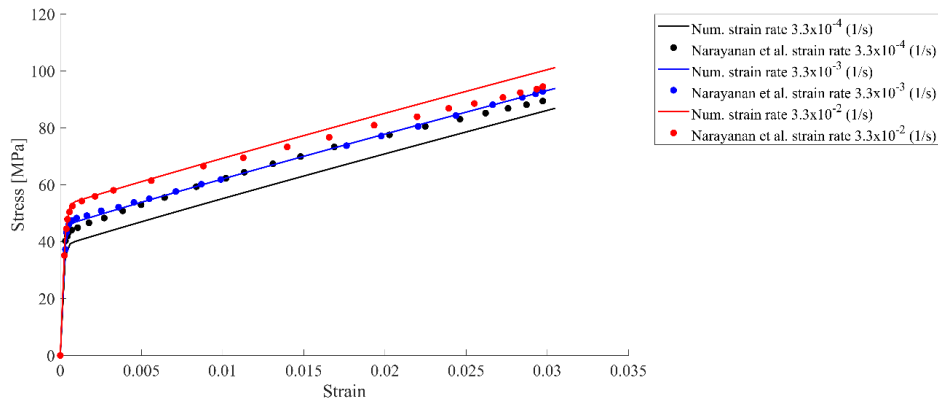
(15). As a result, the suggested model has an inherent reliance on strain rate that is not as straightforward as the power-law method typically used to determine slip rates.

Table 3-4 displays the various strain rates that were implemented in reference [4], while Figure 3-7 illustrates the corresponding engineering stress-strain graphs of the distinct model scales. The table also presents the model constants that were utilized to adjust the outcomes in Table 3-7.

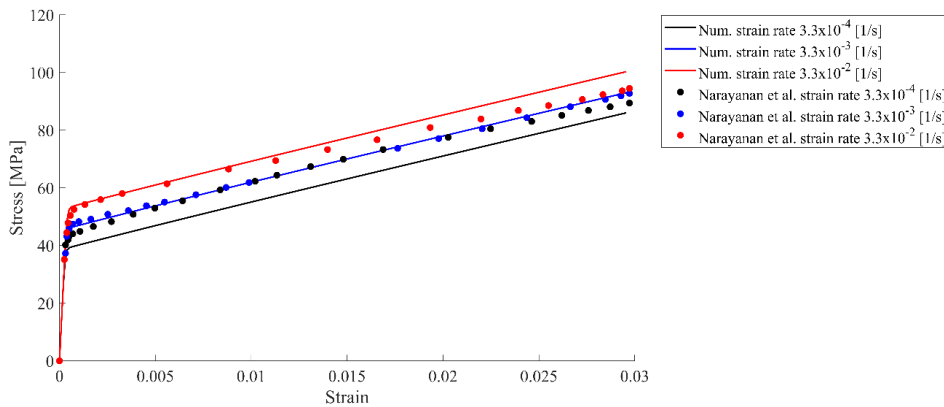
Table 3-4 The reference [4] utilized numerical test cases to examine the strain rate dependency of α -iron.

Temperature (K)	Strain rate (s^{-1})	Loading direction	Load type
300	3.3×10^{-4} , 3.3×10^{-3} , 3.3×10^{-2}	[1 0 0]	Tension

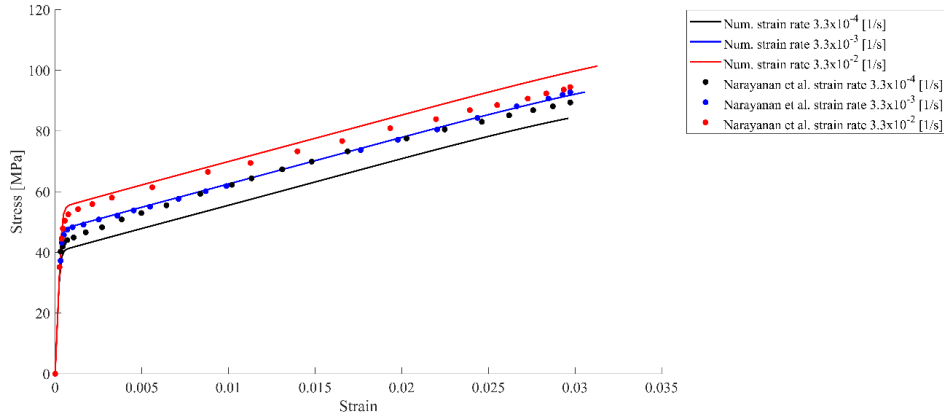
The outcomes for three distinct strain rates are depicted in Figure 3-7. All models exhibit a good correlation with the strain rate dependence of the reference information. It was observed that the MP model curves match the reference graphs more closely.



(a)



(b)



(c)

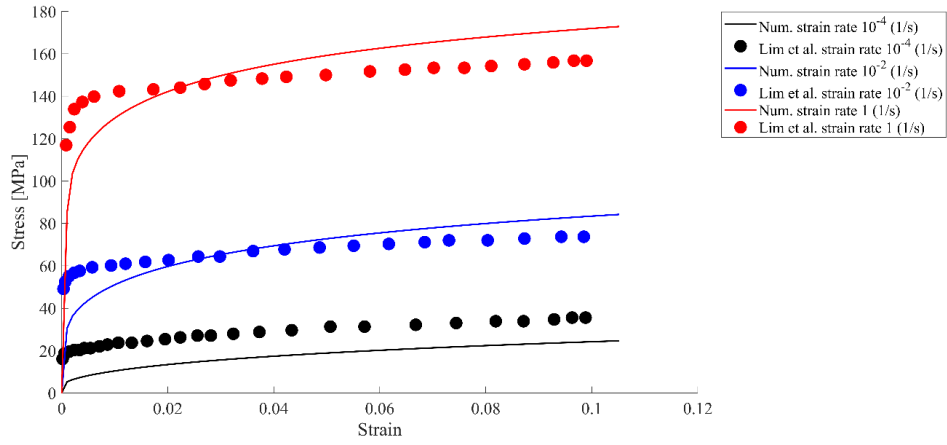
Figure 3-7 The engineering stress-strain diagrams for various strain rates of the MP, SE, and ETG models are presented in (a), (b), and (c), respectively, along with a comparison to the reference [4]

In a comparable manner, the model that has been introduced is utilized to replicate how the mechanical response changes in relation to strain rate reported in another reference [157]. The essential information to run the simulation is given in Table 3-5, and the constants of the model that were adjusted through calibration are recorded in Table 3-7.

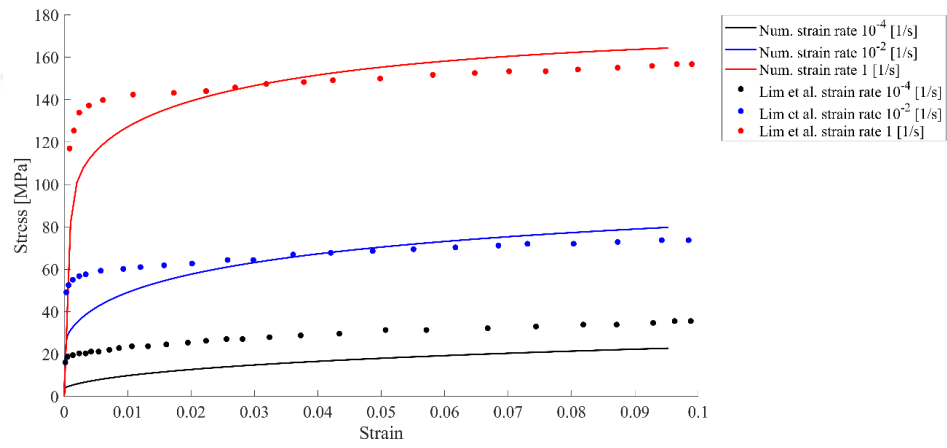
Table 3-5 Numerical test cases used for strain rate dependence of α -iron in the reference [157].

Temperature (K)	Strain rate (s^{-1})	Loading direction	Load type
300	1.0×10^{-4} , 1.0×10^{-2} , 1.0	$[\bar{1} \ 4 \ 9]$	Tension

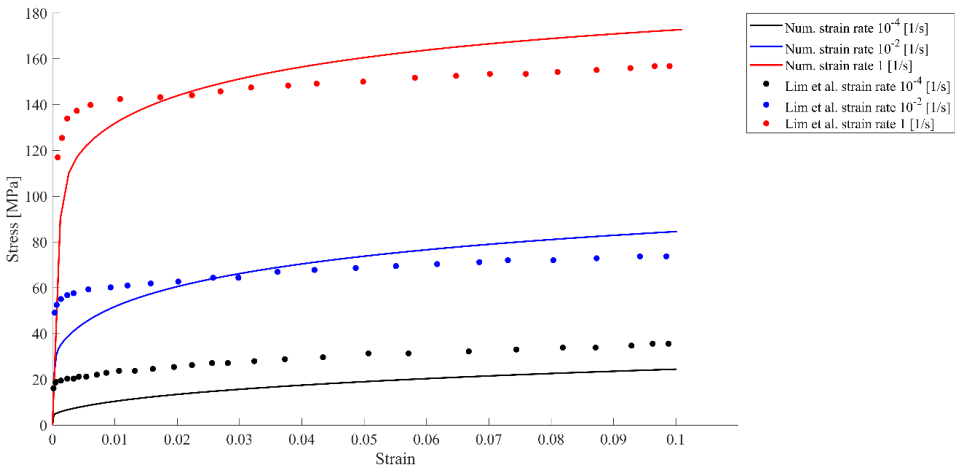
The outcomes of the suggested model and the reference data from [157] are displayed in Figure 3-8. The reference data showed a noticeable contrast for different magnitudes of strain rates. Therefore, this evaluation highlights the improved similarity of the stress-strain reaction.



(a)



(b)



(c)

Figure 3-8 Stress–strain curves for strain rates using different model scales of (a) MP, (b) SE, (c) ETG models and comparison to the reference [157].

3.1.4 Tension–compression asymmetry

Zwiese and Diehl [156] conducted an experimental investigation on the tension-compression response of α -iron and found that there is an imbalance or lack of symmetry in this response. Flow stress versus temperature data was obtained by conducting tension and compression tests on single crystal samples with orientations that were close to each other stereographically. These experiments were conducted at different temperature levels.

Table 3-6 Numerical test cases used for strain rate dependence of α -iron in the reference [156].

Temperature (K)	Strain rate (s^{-1})	Loading direction	Load type
300, 100, 150, 200, 250, 300, 350 and 400	5.6×10^{-4}	[−0.282 0.511 0.811]	Tension
		[−0.309 0.462 0.830]	Tension
		[−0.266 0.460 0.847]	Compression
		[−0.277 0.494 0.824]	Compression

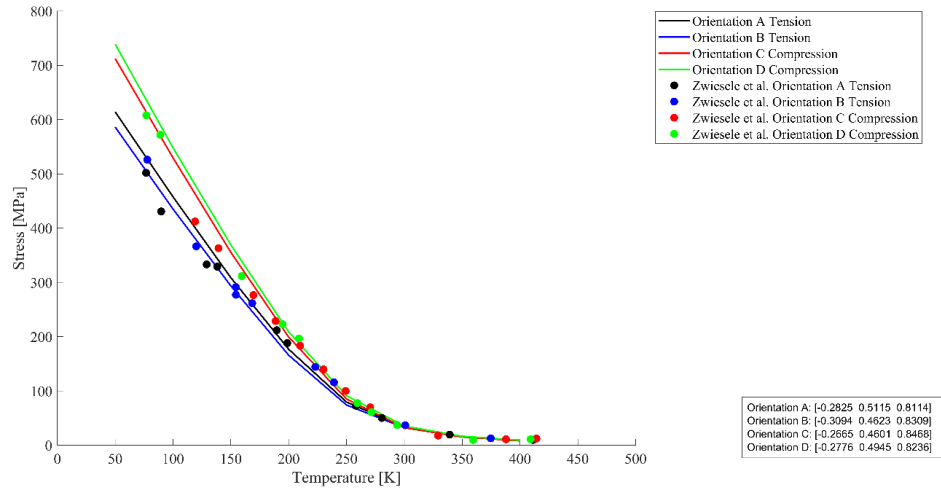
In order to replicate the experimental results obtained by Zwiese and Diehl [156], a model was developed and used to simulate the yield stress of a single crystal of iron under both compressive and tensile loading modes. The model was calibrated using the experimental data and the simulations were conducted applying specific values for strain rate, crystal orientation, total deformation, and temperature that match the ones established by the researchers. Table 3-6 outlines the specific conditions used in the simulations, while the values in Table 3-7 were used to calibrate the model constants.

Table 3-7 Description and values of constants applied for mechanical response calibration.

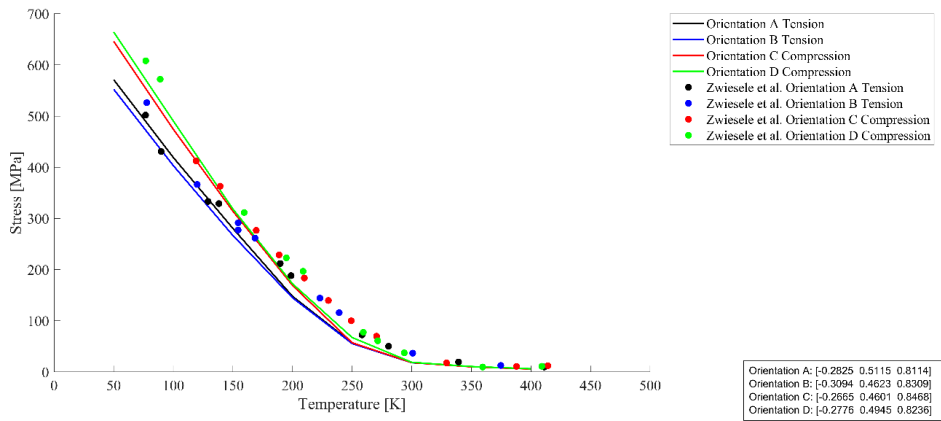
Constants	Ori. Dep. [6]	Temp. Dep. [155]	Temp. Dep. [156]	Rate Dep. [4]	Rate Dep. [157]	Rate Dep. [156]
c_1	0.2	0.03	0.03	0.001	0.03	0.1
c_2	7	11	1	15	1	1

c_3	10	9	155	24	155	16
c_4	0.006	0.03	0.03	0.03	0.02	0.1
c_5	10	10	10	10	10	10
c_6	1000	1000	1000	1000	1000	1000
c_7	0.1	0.1	0.1	0.1	0.1	0.1
c_8	0.08	0.08	0.08	0.08	0.08	0.08
a_1	0.02	0.1	0.08	0.08	0.08	0.35
a_2	-0.02	0.02	0.001	0.02	0.001	-0.4
a_3	-0.58	0.1	-0.0002	-0.5	-0.0002	0.34
ρ^0	1×10^{10}	1×10^{11}	1×10^{12}	1×10^{11}	1×10^{12}	1×10^{10}
$Q_{Pei} \times 10^{-19}$	1.009	0.881	1.041	0.865	1.041	0.865

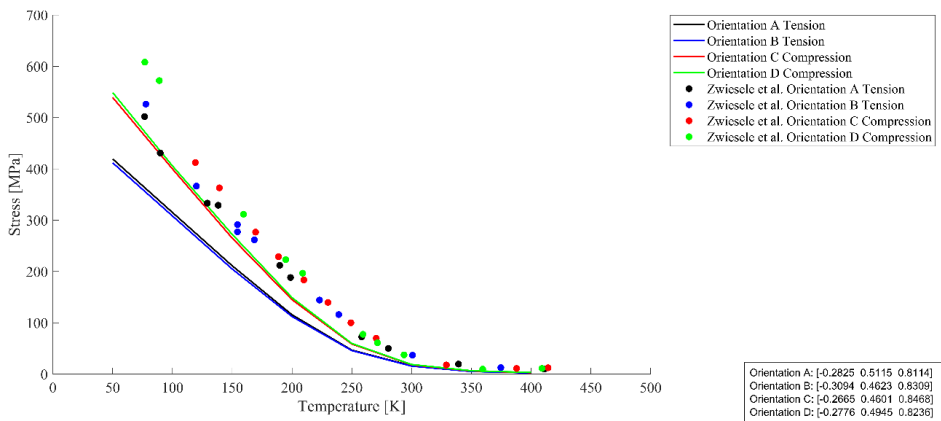
The results obtained from the model in Figure 3-9, along with data from [156], demonstrate that all of the model scales were able to replicate the uneven tension-compression response behavior. Out of the models presented, the SE model provided the closest match. The proposed method of modeling did not incorporate temperature-dependent non-Schmid constants, which accounts for the difference in flow stress estimations. Despite the use of a straightforward modeling approach, it was successful in reproducing the asymmetric stress-strain curves observed during tension and compression.



(a)



(b)



(c)

Figure 3-9 The tension-compression asymmetry of α -iron single crystals was simulated at various temperatures using different model scales, namely MP, SE, and ETG, as well as experimental results from [156]. The purpose of this was to compare the simulated asymmetry across different models and temperatures.

3.2 Dual-phase RVE analysis

3.2.1 Mechanical response

We begin by displaying Figure 3-10, which illustrates the flow characteristics of the SP and DP simulation cases, employing the HYPELA2-CP and HYPELA2-CPJ2 models, across various levels of grain refinement in a tensile mode. The addition of the martensite phase in the numerical models led to enhanced hardening, demonstrated as lower strain values for a given stress. This observation aligns with the findings of Liu et al. [158], which suggest that the inherent hardness of the martensite phase contributes to increasing the material's resistance to plastic deformation. Additionally, it was noted that the reduced plastic deformation ability in the dual-phase steel, attributed to an increased work-hardening rate, exceeded what could typically be anticipated solely from the presence of martensite. This is evident from the equivalent strain distribution depicted in Figure 3-11 for the HYPELA2-CP and HYPELA2-CPJ2 models.

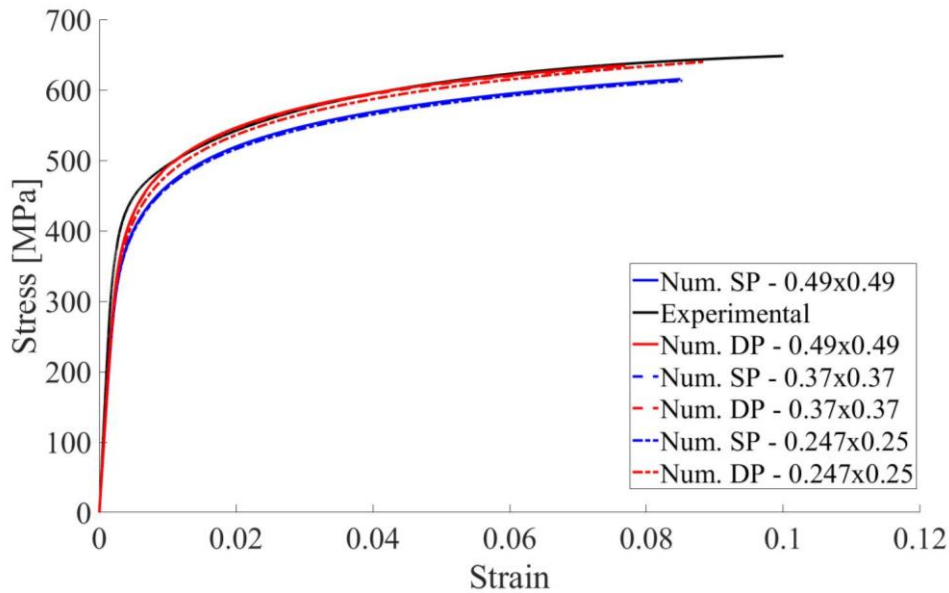
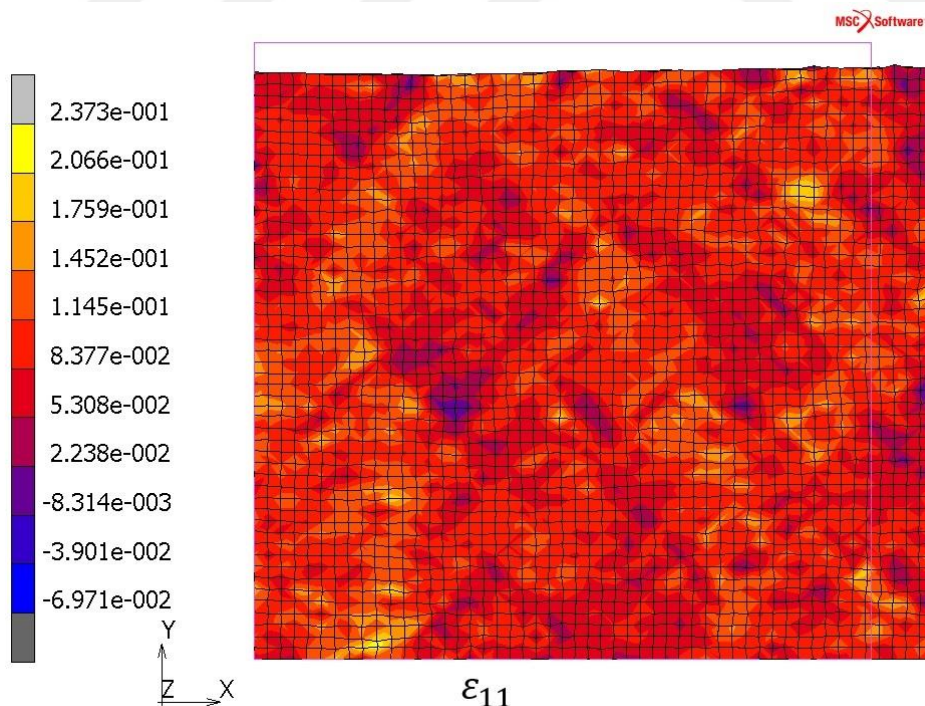


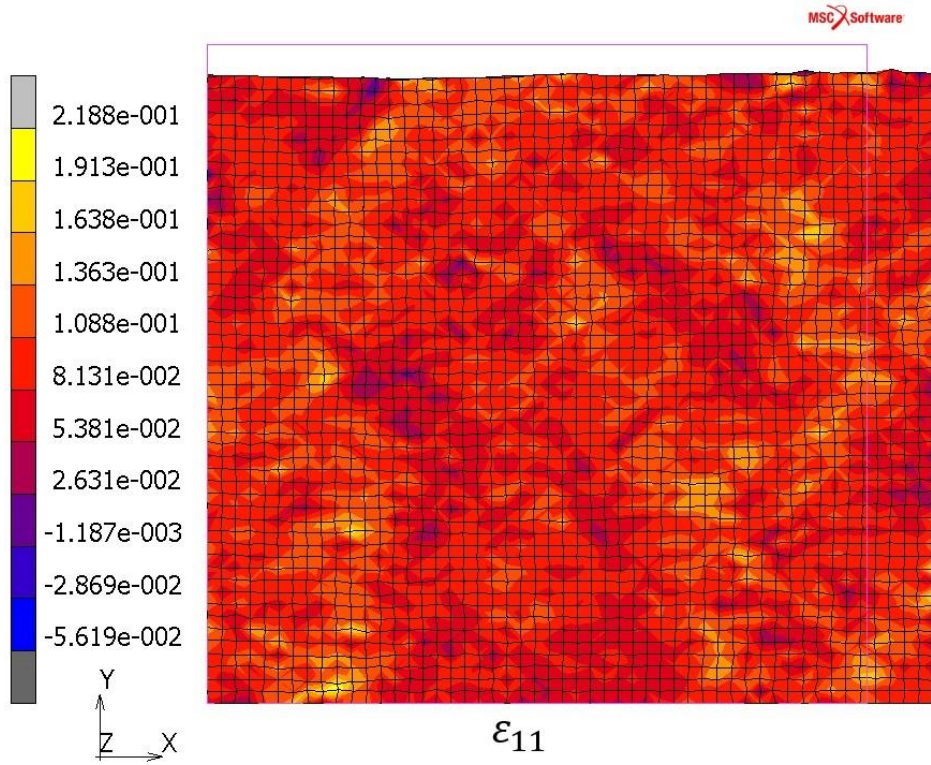
Figure 3-10 overlaid stress-strain graphs of the experimental and numerical tensile tests. The results report the plastic deformation behavior for different grain (element) size in numerical simulations, $0.49\ \mu\text{m} \times 0.49\ \mu\text{m}$, $0.373\ \mu\text{m} \times 0.373\ \mu\text{m}$ and $0.247\ \mu\text{m} \times 0.25\ \mu\text{m}$. In the legend of figure, SP, DP and Num. terms mean single phase, dual phase and numerical, respectively.

The introduction of a small volume fraction of the martensite region (approximately 4.4%) has significantly elevated the material's mechanical strength and its ability to resist plastic deformation through enhanced work hardening. This phenomenon is a result of

the notable discrepancies in hardness between the phases, which prompts an increased formation of geometrically necessary dislocations (GND) and the subsequent rise in Mobile Dislocation Density (MDD) within the softer ferrite grains surrounding the harder martensite regions (refer to Table 2-4 for the mechanical properties of martensite, which indicate its superior strength compared to ferrite). Elevated dislocation densities concentrated in the DP steel led to localized strain, subsequently raising the energy needed for deformation to advance. Localized areas of high strain intensify the material's resistance to deformation prior to failure. To put it differently, the presence of martensite in close proximity hinders the ease of deformation within the ferrite matrix by increasing the concentration of dislocations in those regions. Figure 3-11 demonstrates how martensite influences the localization of strain (plastic deformation) by comparing the mechanical responses of dual-phase and single-phase materials. For an applied strain, a more profound inhomogeneous stress distribution with higher amplitudes is observed in the DP steel compared to the SP steel as given in Figure 3-11. This analysis provides numerical evidence that confirms the widely recognized notion that the presence of martensite and its specific spatial arrangement notably enhances the localized strain within the ferrite phase, as previously acknowledged by Park et al. [159].



(a)



(b)

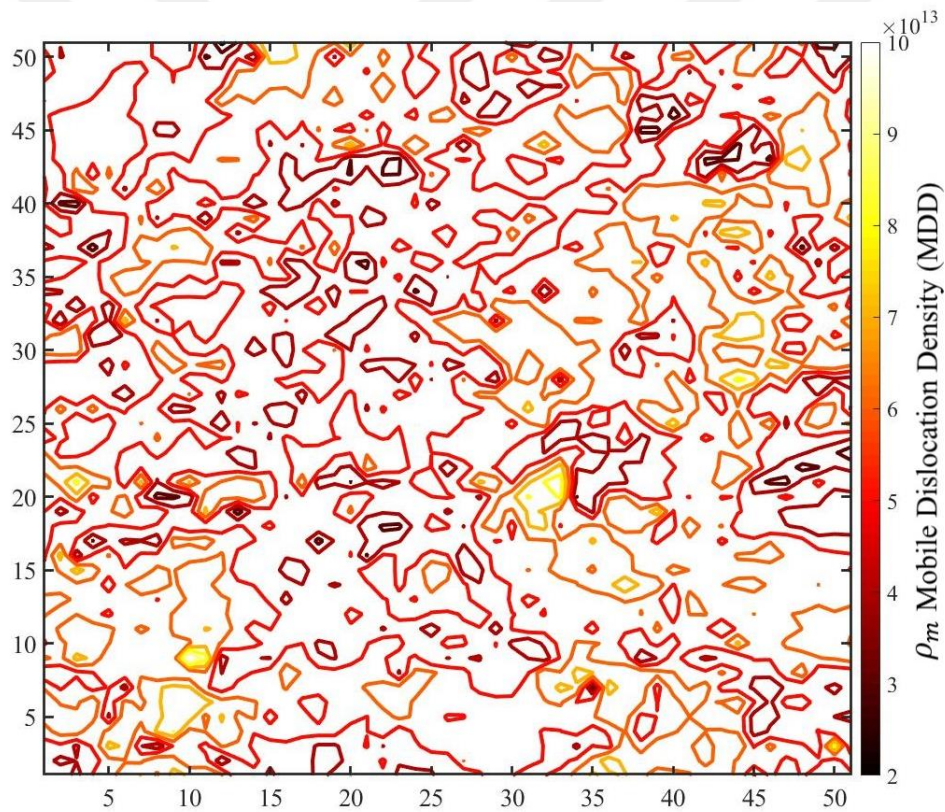
Figure 3-11 The distribution of effective strain in the tensile scenario with a discretization size of $0.49\mu\text{m} \times 0.49\mu\text{m}$ is illustrated as follows: (a) for the HYPELA2-CPJ2 model (representing the dual phase, DP), and (b) for the HYPELA2-CP model (representing the single phase, SP).

Based on the tensile loading outcomes, we can deduce that the initial plastic deformation in the ferrite phase is driven by dislocation glide. This suggests that the ferrite phase's behavior and active slip systems play a predominant role in the early stages of plastic deformation. This is primarily because the ferrite phase is mechanically weaker, exhibiting relatively lower stiffness characteristics, when compared to the martensite phase, as indicated by prior studies [45, 95, 160].

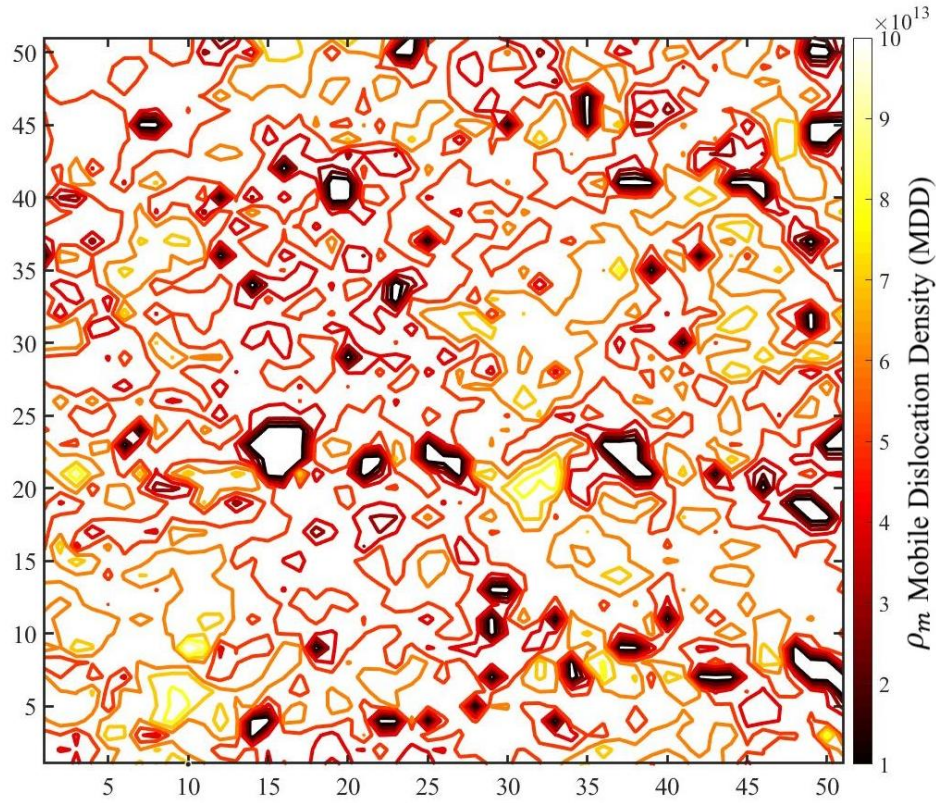
The fundamental physical characteristics of the model that has been created are responsible for the impact of martensite on the ferrite phase and, consequently, on the behavior of the dual-phase material. Hardening behavior in the model evolves based on MDD and GND density of the active slip systems, which are fundamental physical parameters in the crystal plasticity context. MDD of a slip system is a parameter dependent upon the GND and effective shear stress. A slip system is activated due to adequate resolved shear stress and subsequently its density is determined based on the corresponding GND density level. The formulation of hardening is provided in (16). To address this, we provide additional explanations about the deformation analysis.

Formation and increase of the MDD and GND densities are also a product of the restraining effect of neighboring hard martensite phase. Hard martensite region imposes extreme plastic deformation conditions on its surrounding soft ferrite grains. That is why we observe enhanced strain concentration in the proximate areas of the martensite regions. This analysis is further substantiated by the more stretched or significantly suppressed misorientation pole figures, which reveal that grains adjacent to martensite regions exhibit a different evolution of misorientation compared to those not in contact with martensite regions.

In Figure 3-12 the effect of martensite on the distribution of mobile dislocation density is illustrated in the form of contour plots. For martensite distribution in the $0.49\mu\text{m}\times 0.49\mu\text{m}$ RVE model refer to Figure 2-12.



(a)



(b)

Figure 3-12 mobile dislocation density distribution in $0.49\mu\text{m} \times 0.49\mu\text{m}$ RVE model, (a) single phase simulation result (HYPERLA2-CP), (b) dual phase distribution (HYPERLA2-CPJ2), martensite regions distribution can be distinguished with small white areas encompassed with wide black borders.

As indicated by the strain distribution graphs in Figure 3-11 for the system represented by a cell size of $0.49\mu\text{m} \times 0.49\mu\text{m}$ under tensile loading, it's evident that the strain distribution in a DP material is notably more varied compared to the single-phase (SP) case.

Even though strain concentrations in specific areas exceed the average strain state within the ferrite, causing plastic deformation to occur, it hinders overall deformation unless elevated stress is applied, as will be demonstrated. To clearly illustrate the impact of martensite regions on strain and stress distribution, an image that clearly highlights these martensite regions is overlaid onto the corresponding distribution images in Figure 3-13. The overlay reveals that the areas where stress and strain values peak are surrounded and adjacent to martensite phase regions. It's evident in the distribution plot that there are channels of heavily plastically deformed ferrite regions stretching around the martensite regions.

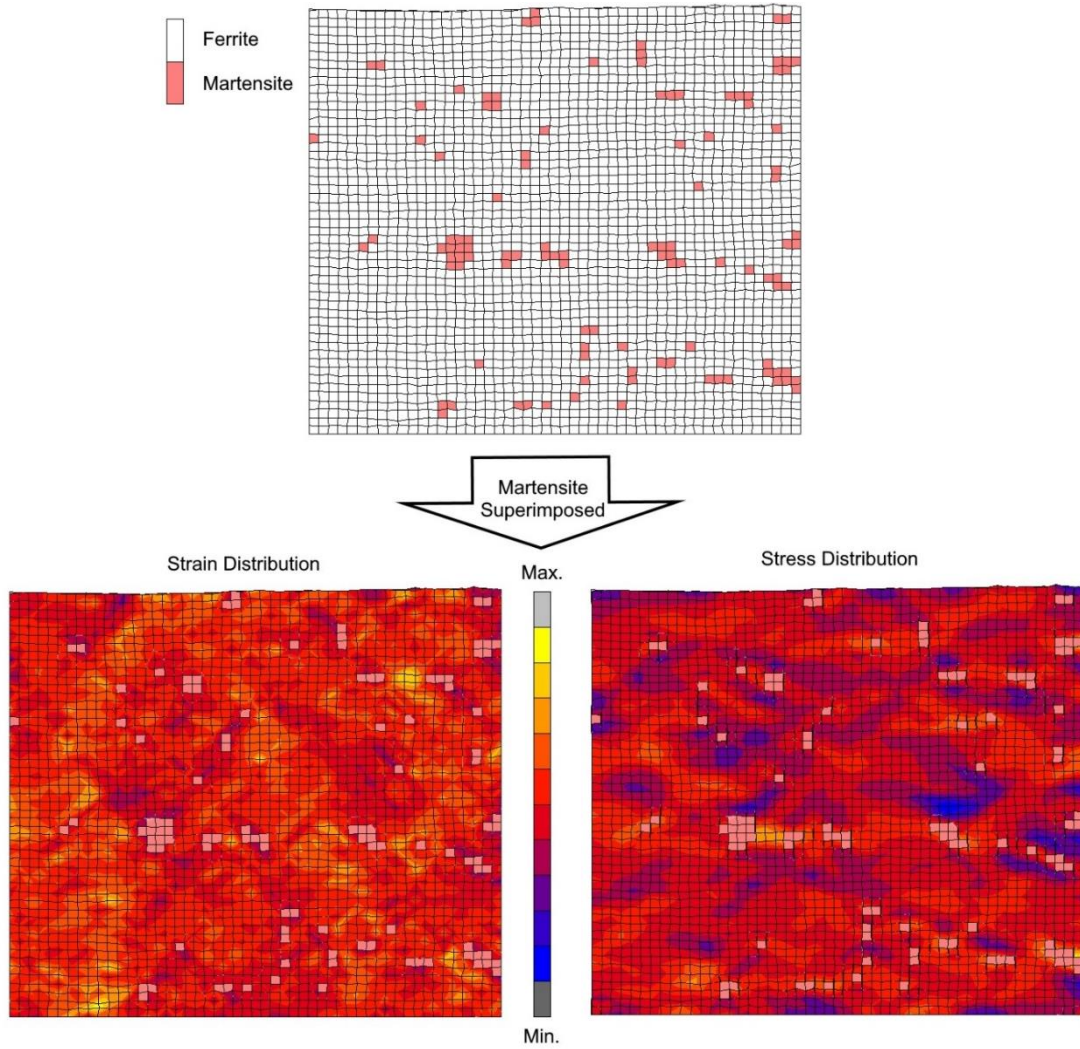
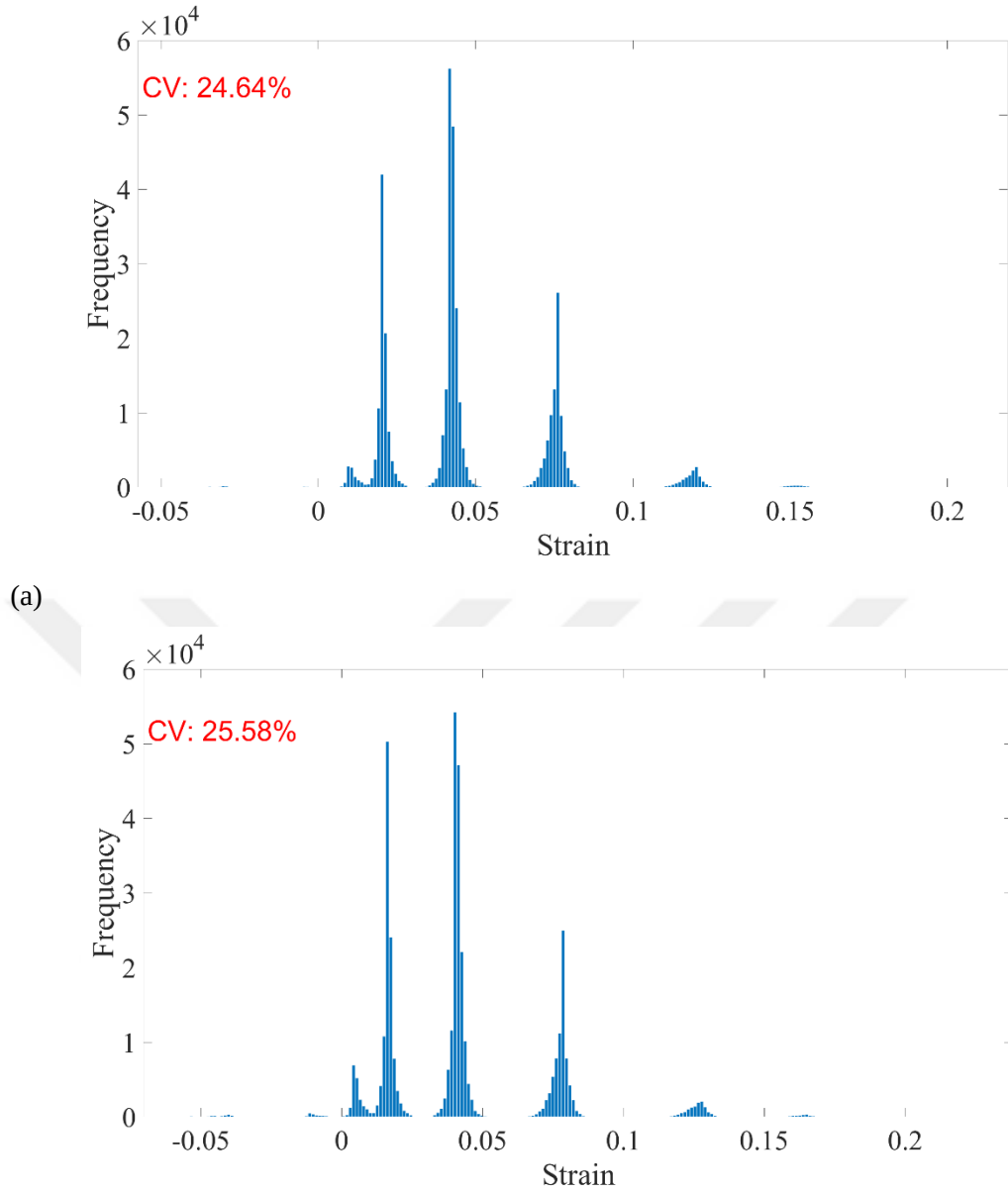


Figure 3-13. Martensite region distribution is superimposed on the deformed RVE model (for stress and strain distribution) in tensile simulation for the $0.49\mu\text{m} \times 0.49\mu\text{m}$ grain size model. The figure show that how regions with extreme strain/stress values are stretched/compressed between the martensite regions. In the figure Dis. Stands for distribution.

To quantify the effect of a 4.4% martensite addition on the inhomogeneity in plastic deformation, the histograms of the strain distribution in both DP and SP cases are presented in 3-14. It can be observed that the coexistence of ferrite and martensite has increased the inhomogeneity of plastic deformation, ultimately leading to the localization of the response.



(b)
3-14 inhomogeneity of plastic deformation distribution histograms and CVs, (a) Single Phase case, (b) Dual Phase case.

Formation of MDD concentration points can also be significantly influenced by the orientation of the ferrite grains due to their cubic stiffness properties. In simpler terms, the initial orientation is a key factor affecting the inhomogeneity in deformation distribution and localization behavior, much like the arrangement of ferrite-martensite phases.

To clearly illustrate the martensite phase on the MDD distribution and filter out the grains orientation effect, we have created a synthetic model in which all ferrite elements are unidirectionally oriented along the cubic axis, with Euler angles of 0° , 0° , 0° . The

model is simulated under two different conditions. In the first condition, it is in a single-phase state where all elements are set to be ferrite. In the second condition, a dual-phase case is created, where five elements are randomly selected and assigned to the martensite phase in order to establish a 5% area fraction of martensite. The model is illustrated in Figure 3-15.

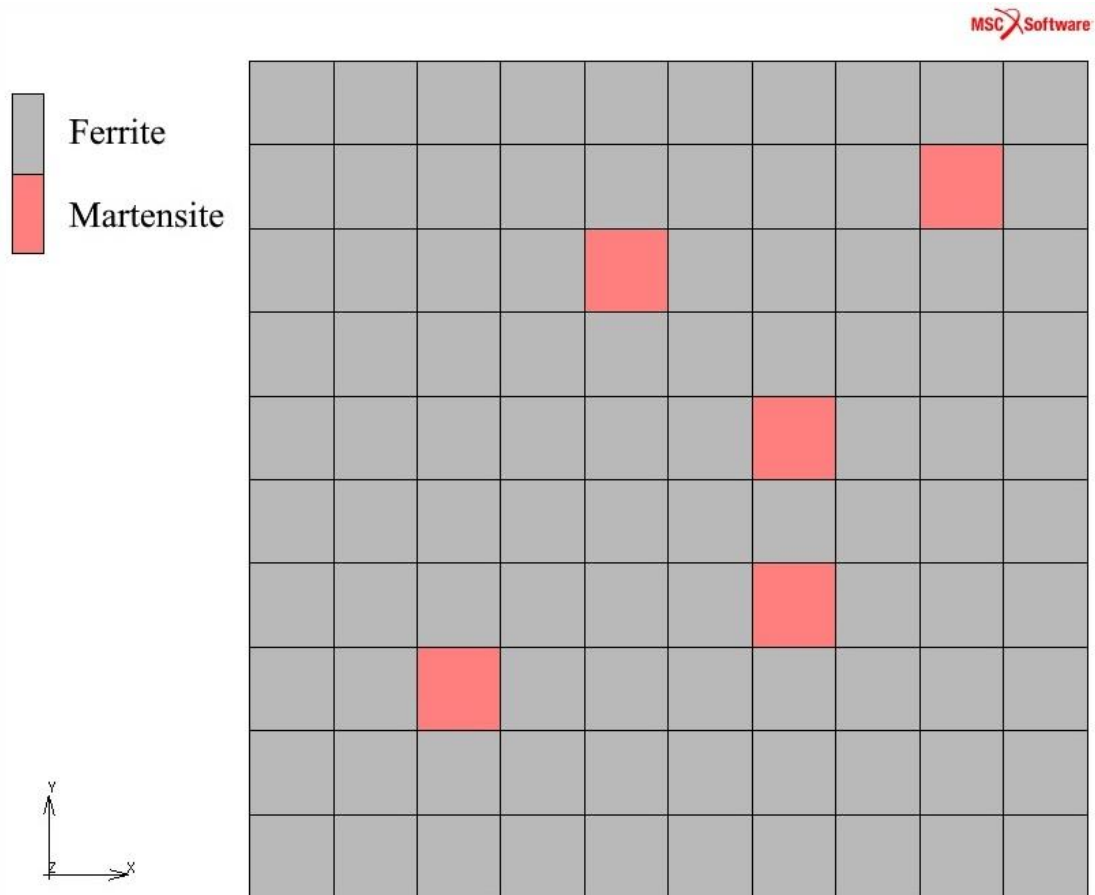
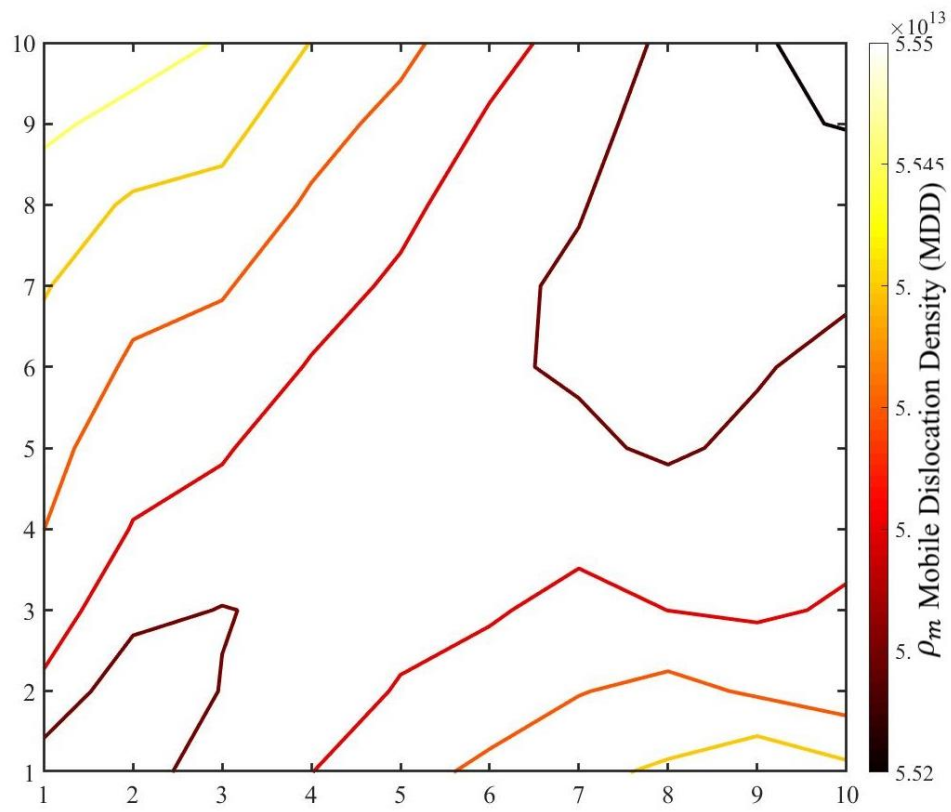
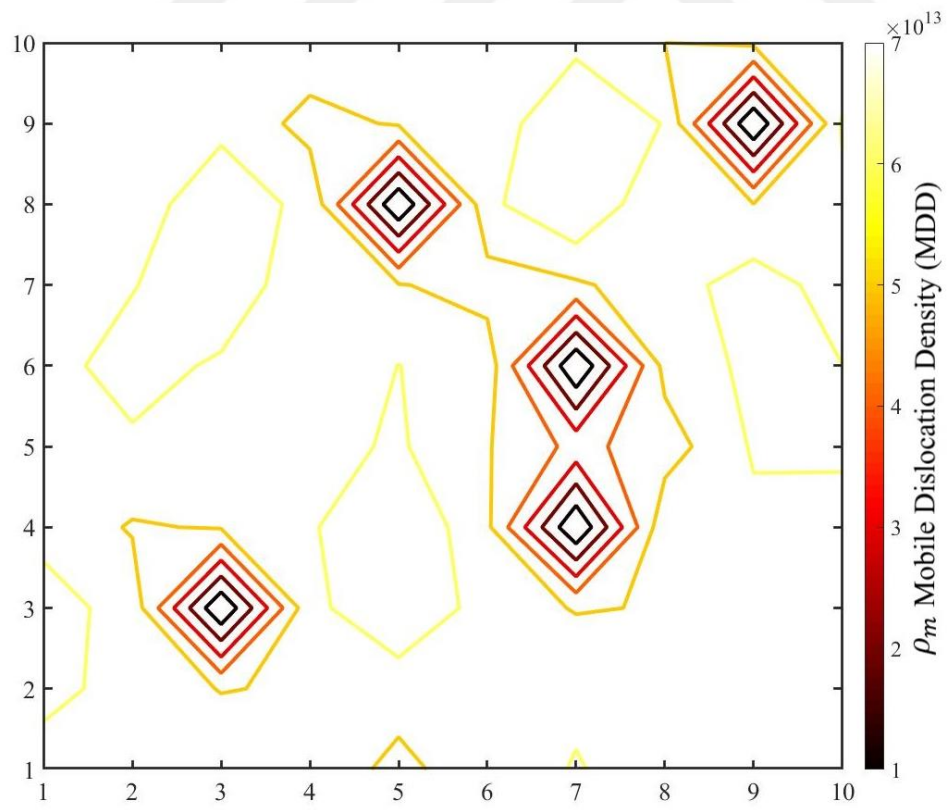


Figure 3-15 A synthetic model with all ferrites in a cubic orientation is used. Elements number 23, 48, 64, 66, and 89 are randomly selected and assigned to the martensite phase in order to establish a 5% martensite area fraction.

The MDD distribution contours of the simulation cases under 10% tensile in the synthetic model are presented in Figure 3-16. As it can be clearly observed addition of martensite significantly influences the homogeneous MDD distribution. In unidirectional loading condition, an almost homogeneous MD distribution can be observed. However, addition of the martensite elements considerably contributes to the concentration and inhomogeneity within the distribution.



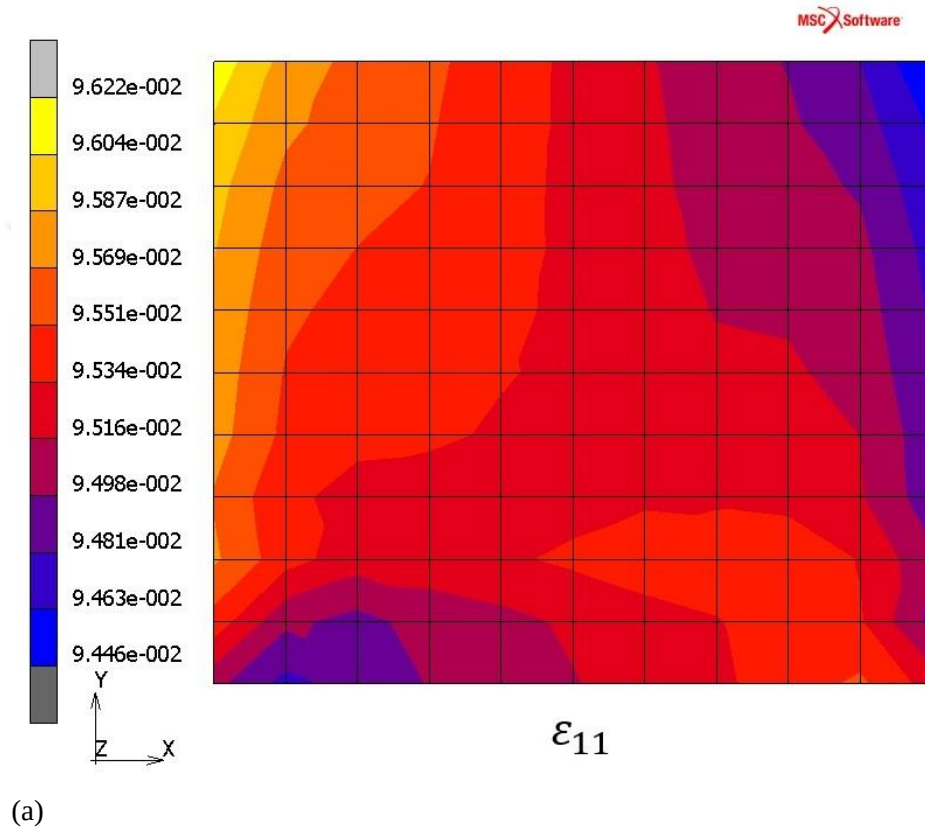
(a)

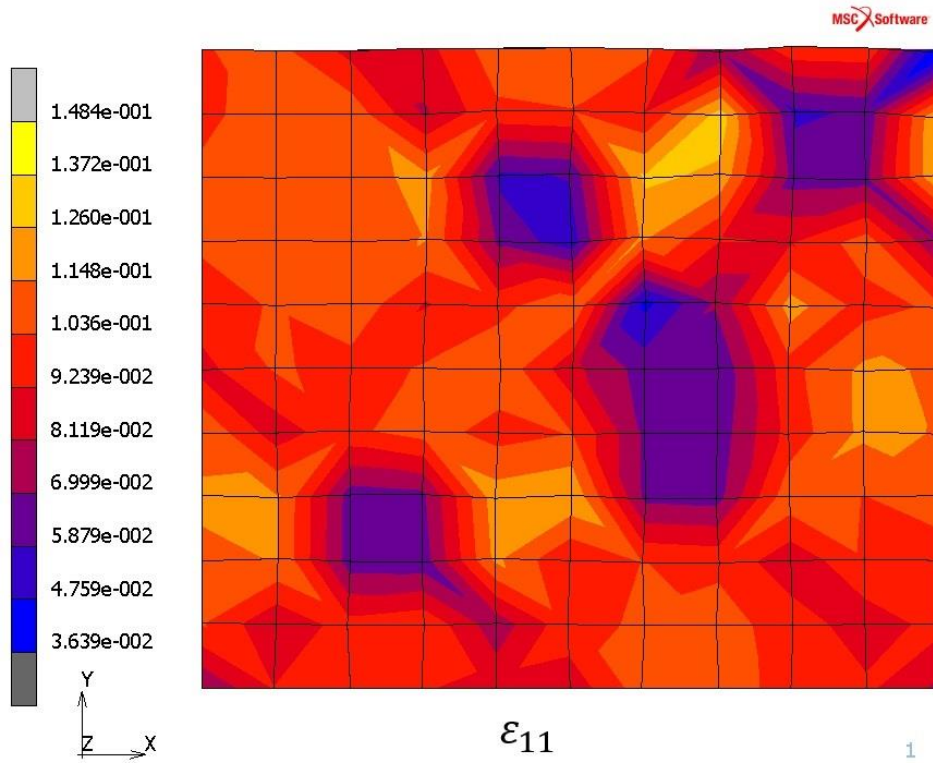


(b)

Figure 3-16 MDD contour plots in a 10×10 synthetic model, (a) single phase simulation case, (b) dual phase simulation case.

The increase in MD is supported by a significant increase in strain observed in the adjacent martensite elements, as shown in Figure 3-17. This indicates that the regions surrounding martensite have undergone substantial plastic deformation, and the severity of this deformation has also increased.

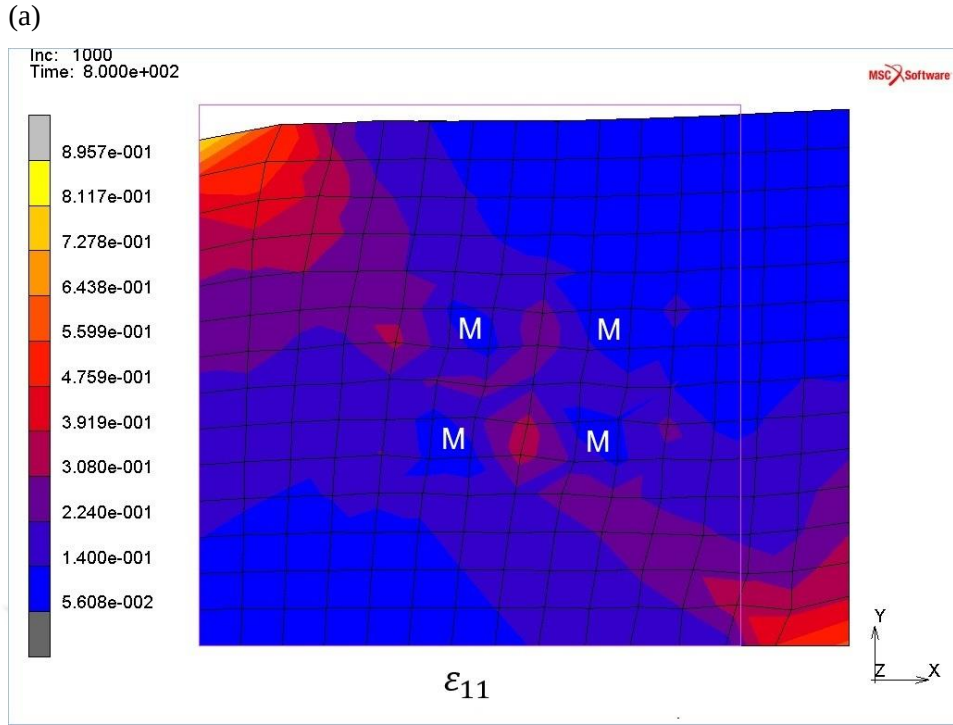




(b)

Figure 3-17 strain distribution in the synthetic unidirectional oriented model, (a) single phase simulation result, (b) dual phase simulation result

Additionally, to clearly show the martensite effect on the disruption in the stress and strain flow, a synthetic model containing 196 elements (14 by 14) is created and simulated in two different conditions, single phase (without martensite) and dual phase (with martensite). The initial orientation of all ferrite elements is deliberately set to 45°, 45°, 0°. This choice of orientation is made to minimize alignment with cubic directions and enhance the susceptibility of the ferrite grains to misorientation during deformation. The model is shown in Figure 3-18.



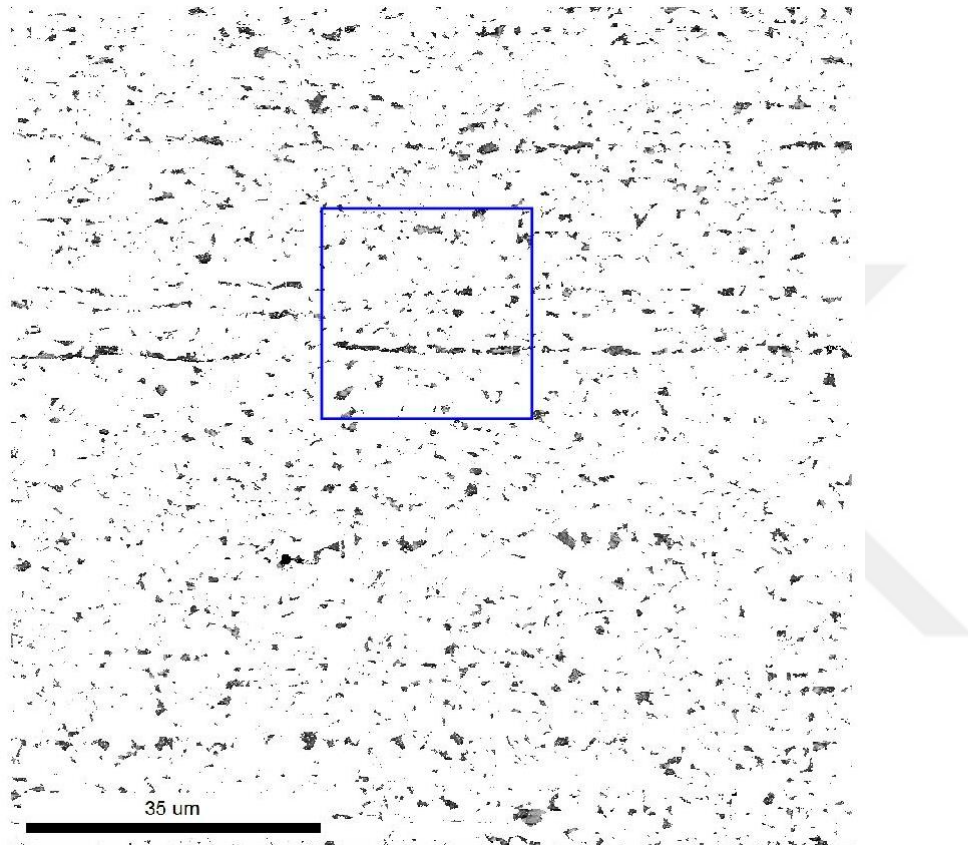
(b)

Figure 3-19 strain and stress distribution in 14×14 synthetic model, (a) strain distribution in single-phase ferrite, (b) strain distribution in dual-phase ferrite-martensite, M in the figure shows martensite regions.

This disruption in strain (deformation) evolution is evidently caused by the difference in mechanical strength between the phases and is consequently the primary factor contributing to strain localization.

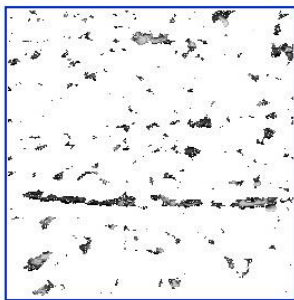
Although, the factor that affects the overall mechanical response (stress-strain curve) of the constituent phases in DP steel is rather area fraction of the phases than the distribution of hard martensite regions, the exact martensite-ferrite phase distribution is one of the main parameters in pinpointing the localization hotspots. The presence of martensite concentration, forming a stretched and continuous region known as a banded distribution, impacts the local stress flow. This ultimately results in a distinct pattern of deformation hotspots. However, this would not affect the behavior that as we similarly observe in our chosen 25 μm x 25 μm RVE. The RVE was discretized into various element refinement sizes, leading to distinct martensite distribution patterns for each case. Implicitly, the analysis output indicates that martensite distribution plays a crucial role in driving stress and strain hotspots, a significant factor in studies related to crack formation and failure.

To substantiate our assertion, we chose a region that encompasses a section featuring banded martensite distribution. We conducted simulations on this area and subsequently compared the response with our primary models. The selected area where overlaps partially with martensite banding is presented in Figure 3-20. As it can be seen in Figure 3-22, strain concentration points are again located in the martensite regions vicinity and there is no significant difference in terms of the parameters investigated in this study.

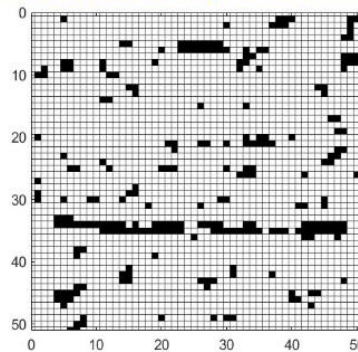


(a)

Original Selected Area



Processed Binary 51x51 RVE



(b)

Figure 3-20 selected martensite banded area, (a) Martensite distribution with the selected area boxed, (b) processed area to extract 51x51 RVE. Martensite area fraction is 9.4%.

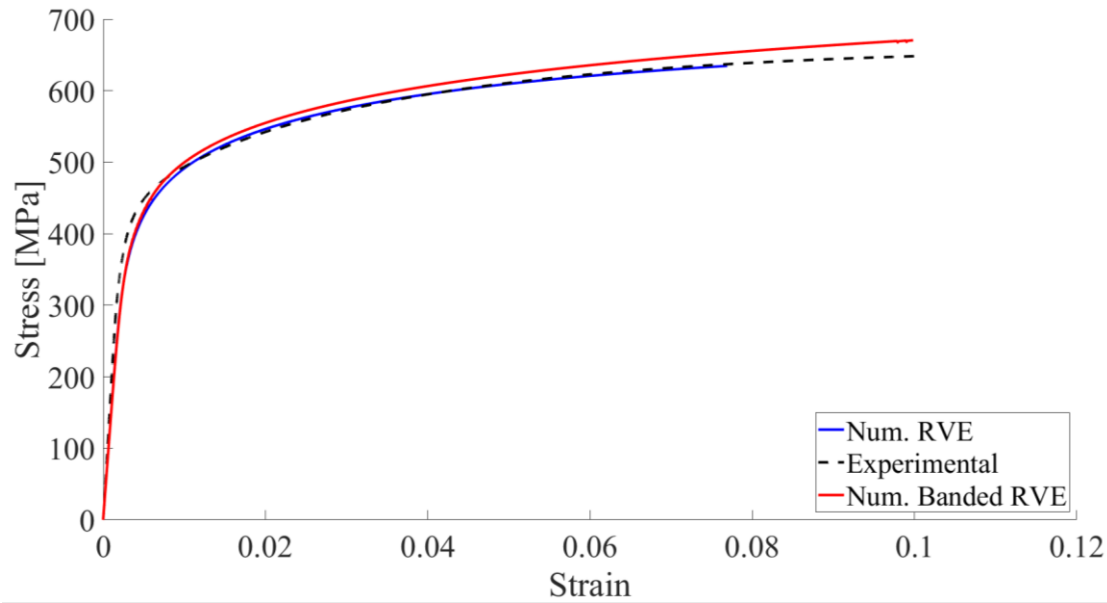


Figure 3-21 comparison of the mechanical response of the main RVE (with 4.4% martensite) and banded RVE in tensile (with 9.4% martensite).

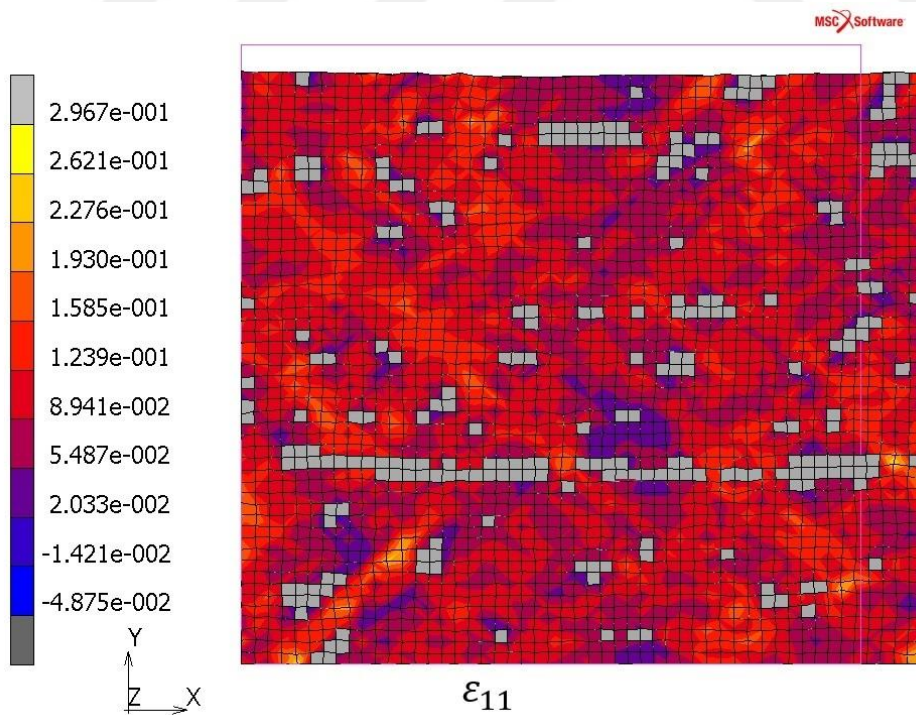


Figure 3-22 strain distribution in the selected martensite banded area, gray cells represent martensite lathes overlaid on the DP RVE.

The substantial enhancement in strain hardening observed in DP models can be explained by the constrained elastic interaction between the martensite and ferrite phases, which, as reported by Tomota and TAMURA [161], is expected to increase the overall flow stress of the DP materials. Put differently, when comparing a dual-phase model to a single-phase model, it becomes evident that in the dual-phase material, the primary factor driving deformation is the ferrite grains, even though their volume fraction is lower than that in a single-phase model of equivalent size. Increased deformation intensity leads to a higher degree of observed strain hardening in the mechanical behavior. This aligns with the research conducted by Avramovic-Cingara and colleagues [162], where the martensite constituent was identified as a key factor contributing to the enhancement of work-hardening. However, the strength of martensite alone does not account for the significant strain hardening, which will be elaborated upon in the forthcoming paragraphs.

3.2.2 Loading conditions effect

In this section we analyze how loading conditions, or loading mode, impact the deformation behavior. Based on the data we've gathered in the previous sections, we can plot simulated stress-strain diagrams for the DP600 alloy that's been examined in this study. Figure 3-23 displays the stress-strain diagrams for tension and compression simulation cases, using the developed HYPELA2-CP and HYPELA2-CPJ2 models. Experimental graphs are derived from Ghaei et al.'s study [154], in which a similar DP600 material was used to examine the plastic deformation response under compression condition, in addition to the tensile case.

The simulation findings suggest that the DP steel's hardening response is notably influenced by the loading direction. This can be explained by the presence of the non-Schmid effect within the ferrite phase, which results in an asymmetry between tension and compression behaviors. This phenomenon takes place when alterations in the loading direction (boundary conditions) lead to changes in the degree to which secondary shear planes in the correlated slip systems contribute. The incorporation of the Non-Schmid effect is achieved by including the non-Schmid stress terms, as detailed and formulated in section 2.1 and (10) respectively. Figure 3-23 also demonstrates that as plastic deformation advances, there is a consistent increase in strain hardening during

compression. As evident from Figure 3-23, the model appears to slightly underestimate the increase in the hardening of the compression case when compared to the experimental findings reported in (Ghaei et al. [154]). This variation is ascribed to the disparity in strain rates used between the model and the actual experiment. In the reference paper the strain rate of the tests is not provided. Furthermore, there is no information about the morphology details including grains initial orientation and phase configuration. With having those details, the model could be properly calibrated to simulate the mechanical response with high accuracy.

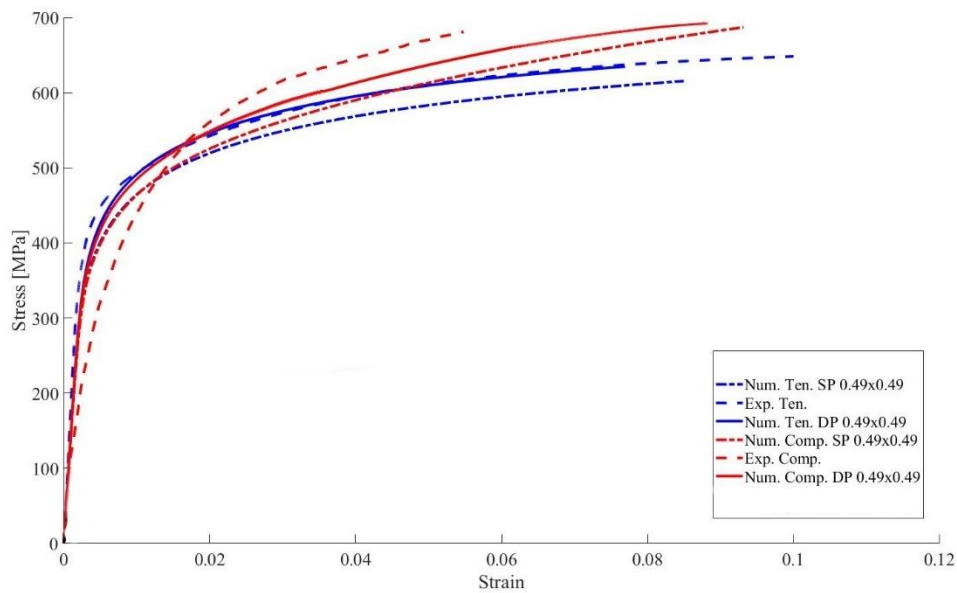


Figure 3-23 Stress-strain results in $0.49 \times 0.49 \mu\text{m}$ grain size. Num. Ten.: Numerical result for tensile simulation, Num. Comp.: Numerical result for compression, Num. Shear: Numerical results for shear simulation, Exp. Ten: Experimental data from tensile testing, Exp. Comp.: Experimental data from achieved Figure 2 in Ghaei et al. study [154], Exp. Shear.: Experimental data taken from Figure 3 reported in Ghaei et al. study [154].

The strain distribution in the DP steel for the three simulated loading case studies are shown in Figure 3-24. It's apparent that the strain heterogeneity is more severe under compression compared to the strain distribution observed in the tensile loading simulation (see Figure 3-11(b)). This phenomenon is linked to an enhanced interaction among dislocations, whose density increases during the deformation processes. This increased dislocation interaction is the primary mechanism responsible for the hardening of the ferrite phase.

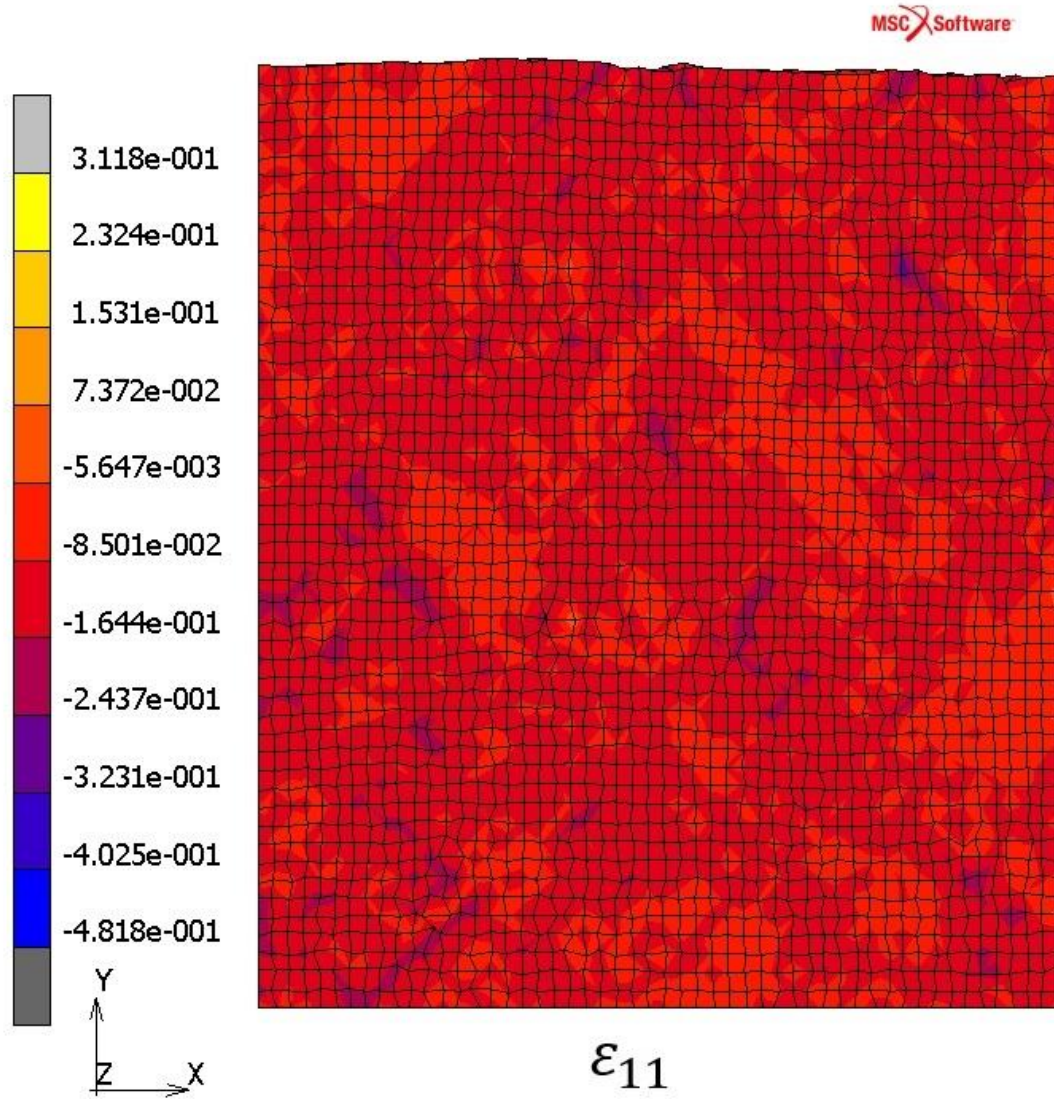


Figure 3-24 ϵ_{11} distribution in the DP steel sample with $0.49\mu\text{m} \times 0.49\mu\text{m}$ grain sized HYPELA-CPJ2 (DP) model under compression.

Section 2.4 provides a comprehensive explanation of the interaction matrix formulation, which takes into account the primary dislocation reactions among the BCC slip systems in the simulations. This is elaborated in Eqn. (32) to Eqn. (34). It outlines the four key interactions that elevate the energy needed for dislocation activation during each deformation increment, ultimately leading to strain hardening. The loading mode misorientation diagrams presented in Figure 3-25 distinctly illustrate that during plastic deformation, the ferrite grains in the DP sample simulations, whether under tension or compression, undergo reorientation and gather at particular locations. These convergence points can aptly be referred to as "focal points." In all the loading modes, including tension and compression, and shear, we have utilized the same set of grains as shown in

Figure 3-30. However, it's important to note that in the case of shear loading, there isn't a distinct focal point that can be pinpointed.

Moreover, it's worth noting that the focal points in the tensile and compression cases are situated in opposite directions (see Figure 3-25). This indicates that the direction of loading has a substantial impact on the orientation changes of the grains, which underscores the reliability of the results, as the ferrite grains tend to reorient along their cubic strength axes. It's important to highlight that when the material is under compression, the interactions among dislocations play a more significant role in terms of hardening. This is because compressive stresses lead to increased dislocation annihilation through mechanisms such as dipole formation, climb, and cross-slip. Furthermore, under compression non-Schmid stress terms contribution is more than that of tensile. Therefore, one would anticipate a more pronounced hardening effect in this stress state compared to other loading modes, and this aligns with the observations in our simulations. Conversely, a reduction in the number of active dislocations within slip systems necessitates additional energy to sustain deformation, leading to increased hardening as the strain accumulates. Ultimately, this increased hardening in the localized areas of deformation results in a relatively more inhomogeneous stress distribution under compression loading compared to tension.

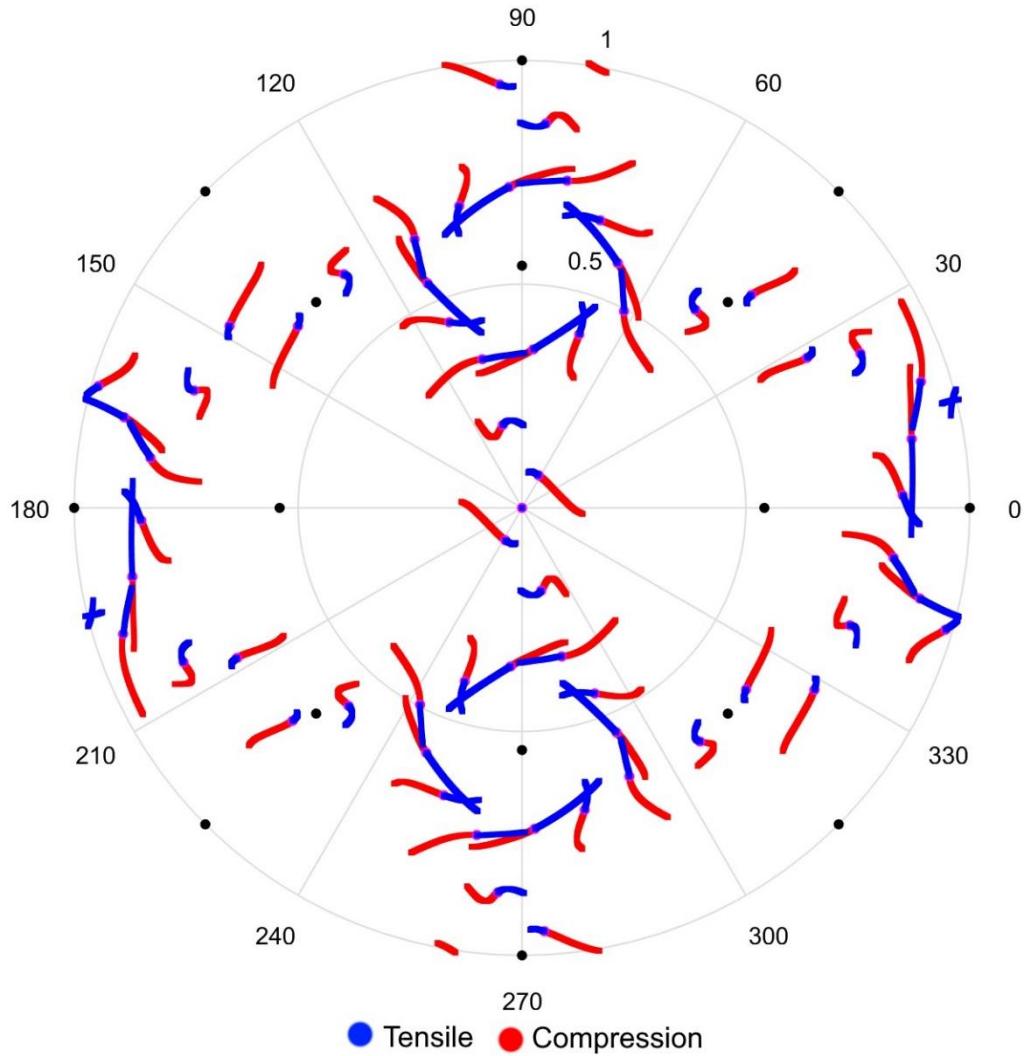


Figure 3-25 Misorientation pole figures of the $0.49\mu\text{m} \times 0.49\mu\text{m}$ HYPELA-CPJ2 model.

In the previous paragraph the difference between heterogeneity in the responses between tensile and compression loading cases were attributed to the non-Schmid contribution and active slip systems effects. To demonstrate this, we examine the evolution of non-Schmid contribution within a 10×10 synthetic RVE with random orientation distribution, as illustrated in Figure 3-26.

The constructed synthetic 10×10 model is subjected to tensile and compression loading conditions. To accomplish this objective, it is necessary to obtain the distribution of pure non-Schmid shear stress terms for both tensile and compression cases. These distributions should be extracted and presented in the form of contour plots. To capture DP steel behavior random element number 23, 48, 64, 66 and 89 are assigned to be martensite phase to establish 5% volume fraction (close to the original RVE).

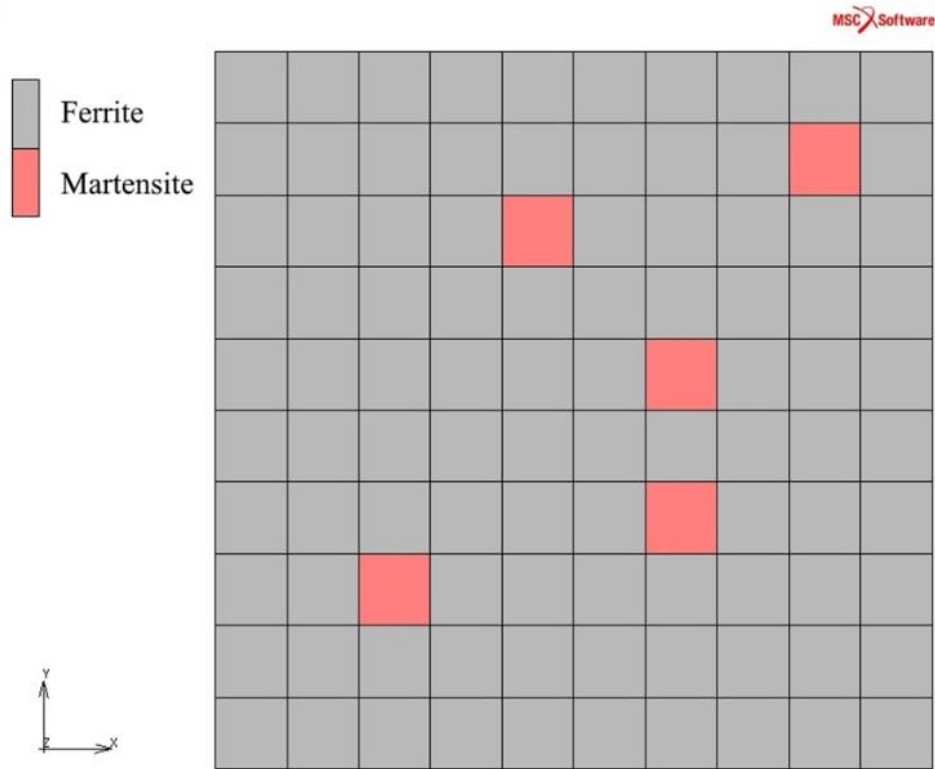


Figure 3-26 10 x 10 RVE sample where the element numbers are designated. Overall sample size is 25 x 25 μm . The orientation of the grains is defined randomly.

Figure 3-27 illustrates the contour distributions of resultant strain and pure non-Schmid stress. It is evident that in the compression case, there is an almost 25% increase in the non-Schmid stress contribution to deformation. This phenomenon is linked to variations in the regime of slip system activity and their interactions between compression and tensile cases. Furthermore, there exists a direct correlation between the locations of deformation hotspots and the positions of the highest values of the non-Schmid term. This correlation elucidates the primary reason for the more pronounced deformation observed in the compression case.

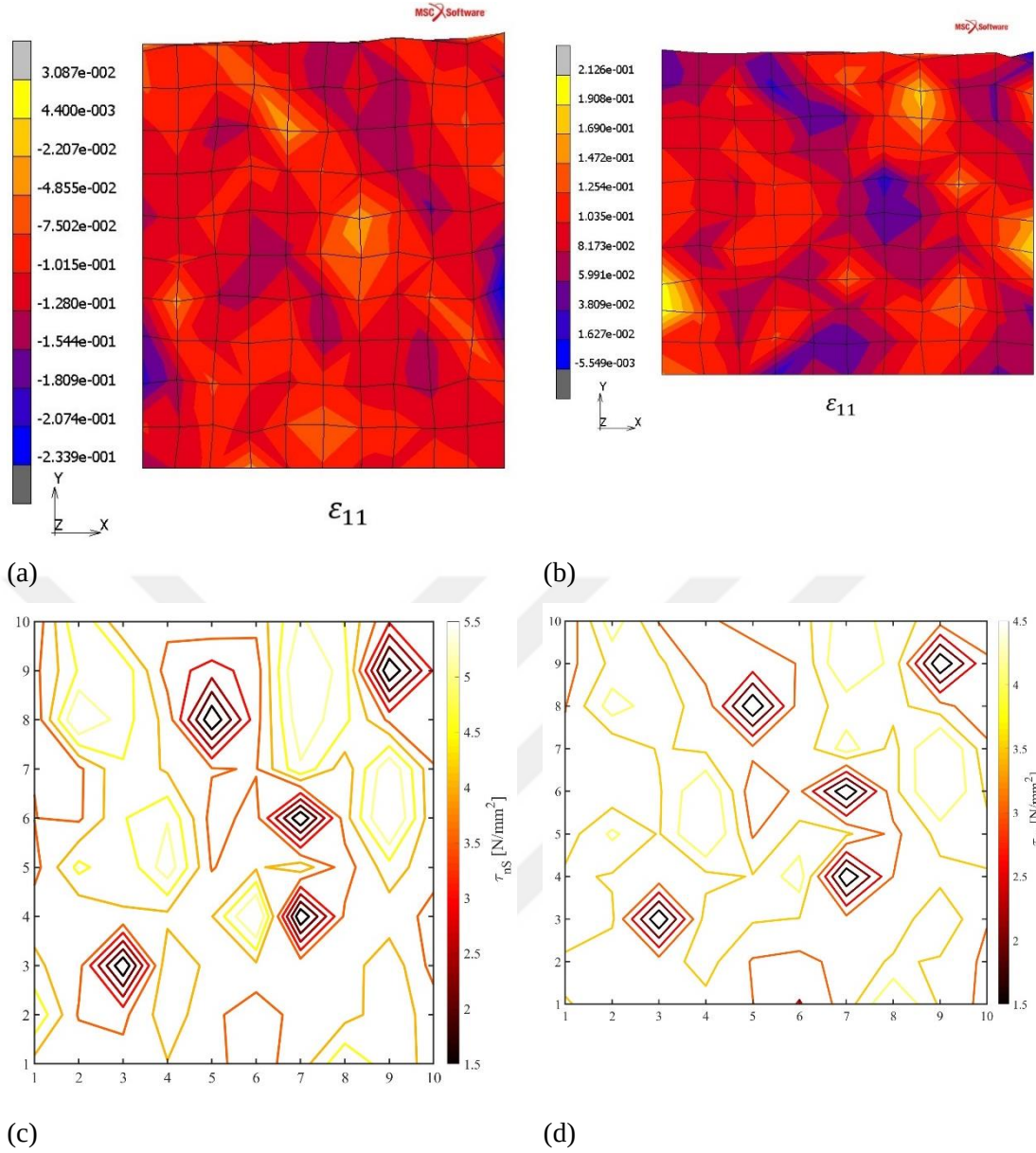


Figure 3-27 resultant distributions of the simulated 10×10 RVE, (a) strain distribution in compression, (b) strain distribution in tensile, (c) pure non-Schmid stress term contour plot for compression, (d) pure non-Schmid stress term contour plot for tensile.

Basic physics of the simulation including slip systems action status, dislocations interaction, non-Schmid stress terms contribution and dislocation density annihilation leads to another distinctive deformation conditions in compression and tensile case. Under compression, elements are more susceptible to misorientation (retexturing) than the tensile case. To show this the average ferrite elements misorientation angle versus strain (growth of the deformation) is plotted along with that of tensile and the graphs are

compared in Figure 3-28. As can be noticed in the figure, as deformation proceeds, ferrite elements undergo more misorientation than the tensile case.

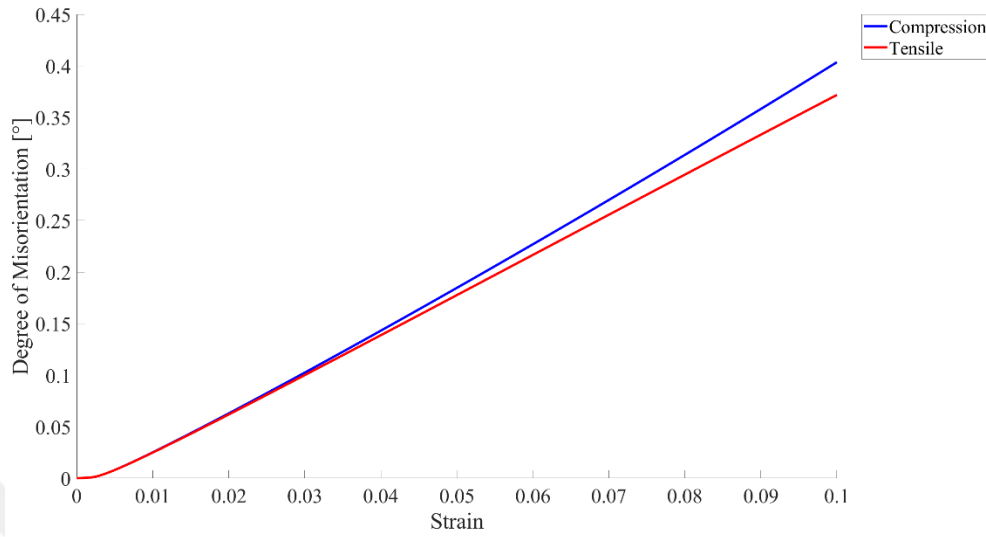


Figure 3-28 average degree of misorientation in the simulated 10×10 RVE under compression and tensile cases.

It is a well-established observation [160] that plastic deformation initiates and propagates in ferrite regions in DP steels, as supported by our simulation results. To establish a correlation between our observations regarding the evolution of fundamental physical parameters and the deformation response under various loading conditions, we have additionally computed the overall evolution of mobile dislocation density across all ferrite elements within the simulated 10×10 synthetic RVE. Regarding the primary physical factors contributing to the asymmetry in the strength response between tensile and compressive cases, our analysis of deformation and post-processing of results revealed a notable divergence. Difference in the contribution of the non-Schmid stress terms between compression and tensile cases resulted in a higher stress level requirements for the slip systems activation under compression and subsequently much more MDD generation in the ferrite regions. This is clearly depicted in the overall average evolution of MDD versus strain in Figure 3-29, wherein the compression case surpasses the tensile case in terms of MDD growth as plastic deformation progresses. The contour plots of the MDD distribution achieved from the simulations under tensile and compression loading cases are also provided in Figure 3-29,

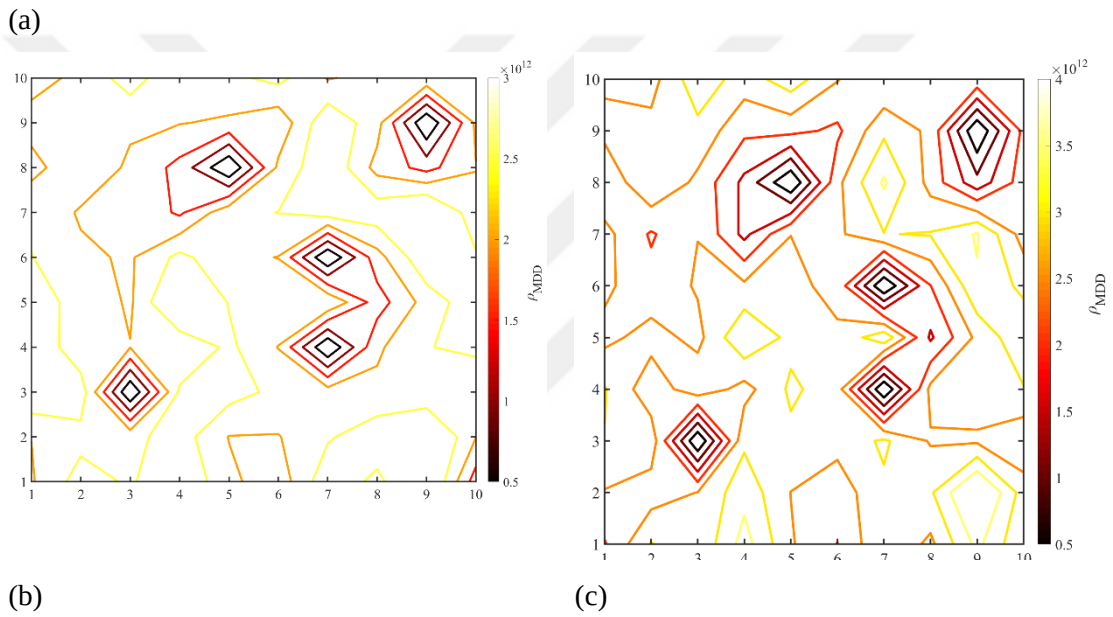
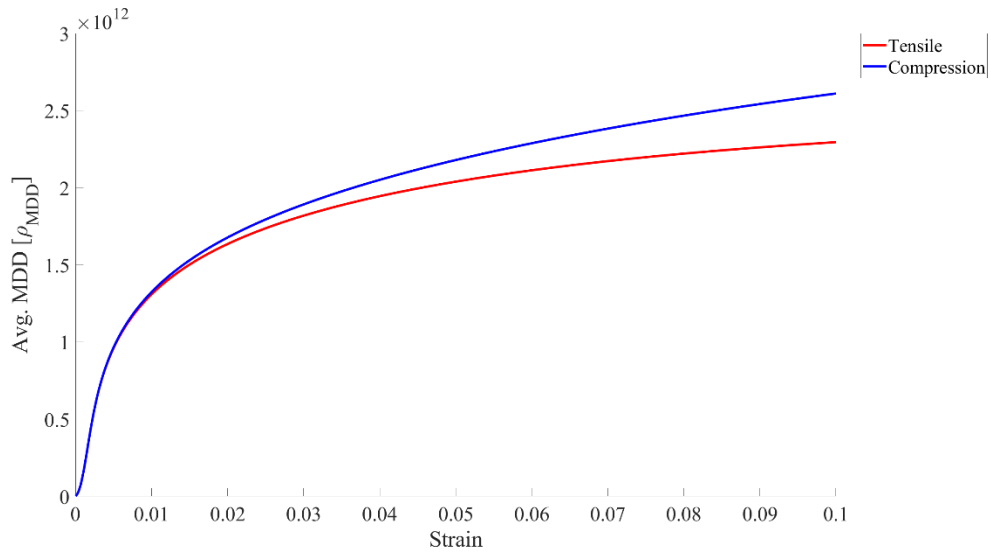


Figure 3-29 (a) average mobile dislocation density versus strain in compression and tensile cases, (b) MDD distribution contour plot in tensile, (c) MDD distribution contour plot in compression.

All these physical phenomena explain the more localized distribution of stress and strain in the case of compression compared to tension, particularly within the ferrite regions and, specifically, in the elements neighboring martensite regions.

3.2.3 Microstructure orientation and phase configuration effect

If we subdivide the RVE into 16 domains. The geometrical distribution of martensite is scattered or concentrated.

Examining the progression of deformation in numerical simulation cases under different orientation conditions and various martensite phase distribution styles is important for understanding the significant effects of the initial orientation of ferrite grains and the geometrical dispersion of martensite regions on the evolution of basic physical parameters. Subsequently, this understanding contributes to elucidating plastic deformation behavior. The main objective is to investigate the influence of phase distribution styles and initial orientation on deformation behavior, with a particular focus on determining the circumstances that lead to more intensive behaviors, especially heterogeneity.

To achieve the mentioned goals, we designed a series of simulation cases using a 12×12 synthetic RVE with randomly distributed initial orientations. Two primary styles of martensite dispersion were considered: fully scattered and concentrated. The diagram for the test cases is illustrated in Figure 3-30. In the simulations, to validate fully scattered and concentrated conditions, the 12×12 RVE is subdivided into 16 domains, each measuring 3 by 3. For fully scattered martensite distribution, one element from each domain is randomly selected to be martensite. However, in the case of concentrated martensite distribution, the domains on the edges of the RVE are excluded, and only 4 central domains are utilized to assign martensite phases. To maintain an identical martensite fraction to that of the fully scattered dispersion case (11.1% martensite volume fraction), 4 elements from each of the 4 central domains are designated as martensite, totaling 16 martensite elements.

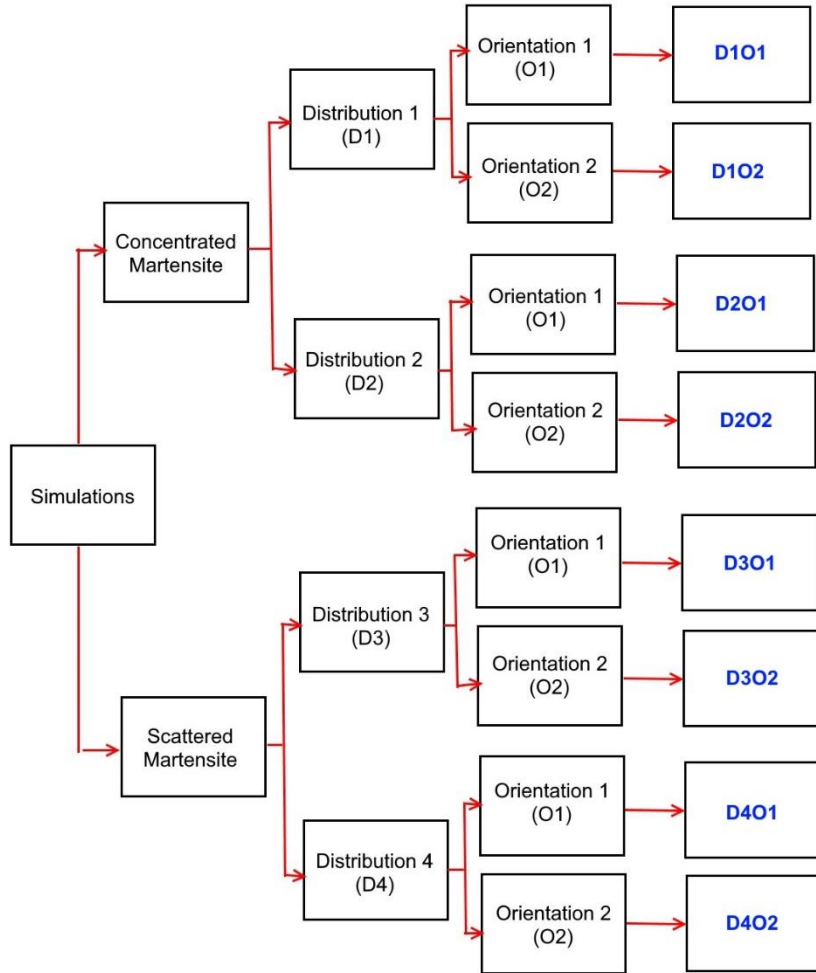


Figure 3-30 simulation cases diagram, totally 8 different simulation combinations are performed.

The domains considered in each of the martensite distribution scenarios (fully scattered and concentrated) to assign the martensite elements are shown in distinctive colors in Figure 3-31. To ensure the reliability of the simulation results, two different configurations are employed for each of the martensite distribution styles, namely fully scattered and concentrated. The distributions of martensite in each of these styles are illustrated in Figure 3-32.

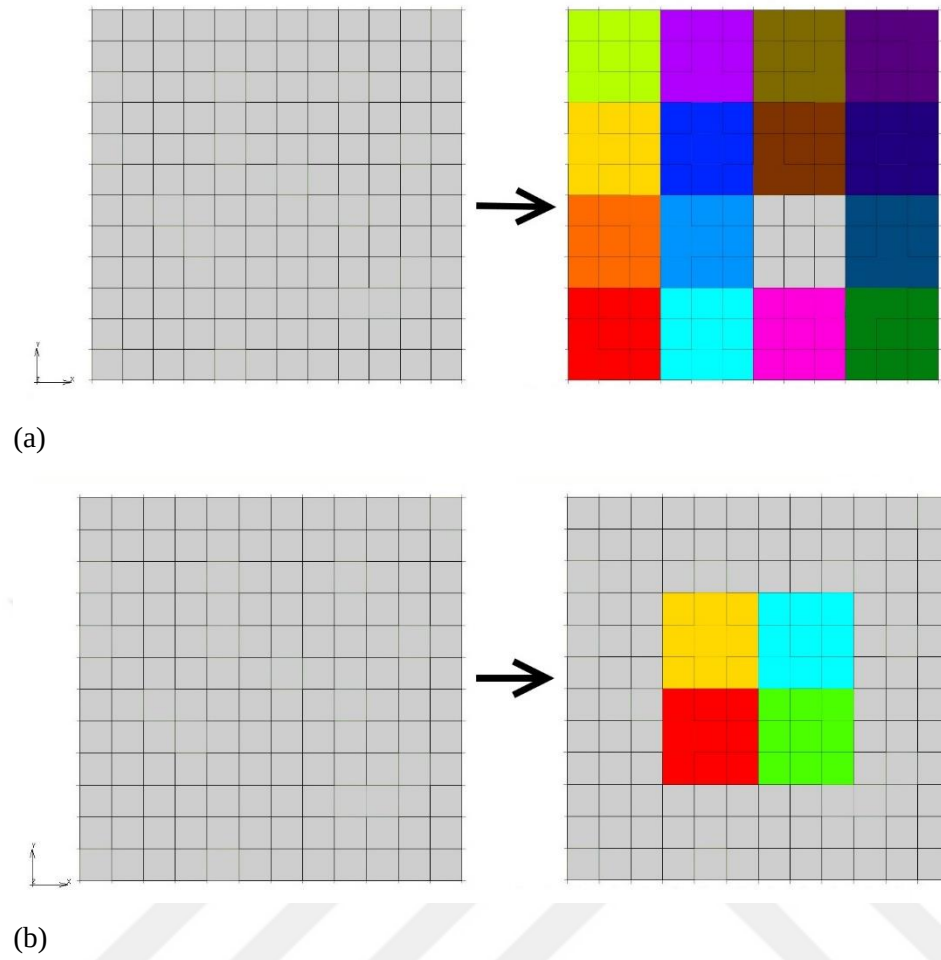


Figure 3-31 domains shown with distinctive colors, (a) domains to define one martensite element randomly for each of them in fully scattered style, (b) domains employed to define four martensite elements randomly for each of them in concentrated martensite dispersion style.

To highlight the impact of the initial orientation, two distinct types of random orientations, O1 and O2, are employed. These orientations are universal and consistent across all martensite dispersion styles, resulting in a total of 8 different simulation scenarios. The O1 orientation case defined to exhibits more inhomogeneity in the initial orientation of elements compared to the O2 case, with creating higher amount of angle deviation between the neighboring elements. The corresponding ODF plots of the applied orientations are provided in Figure 3-33.

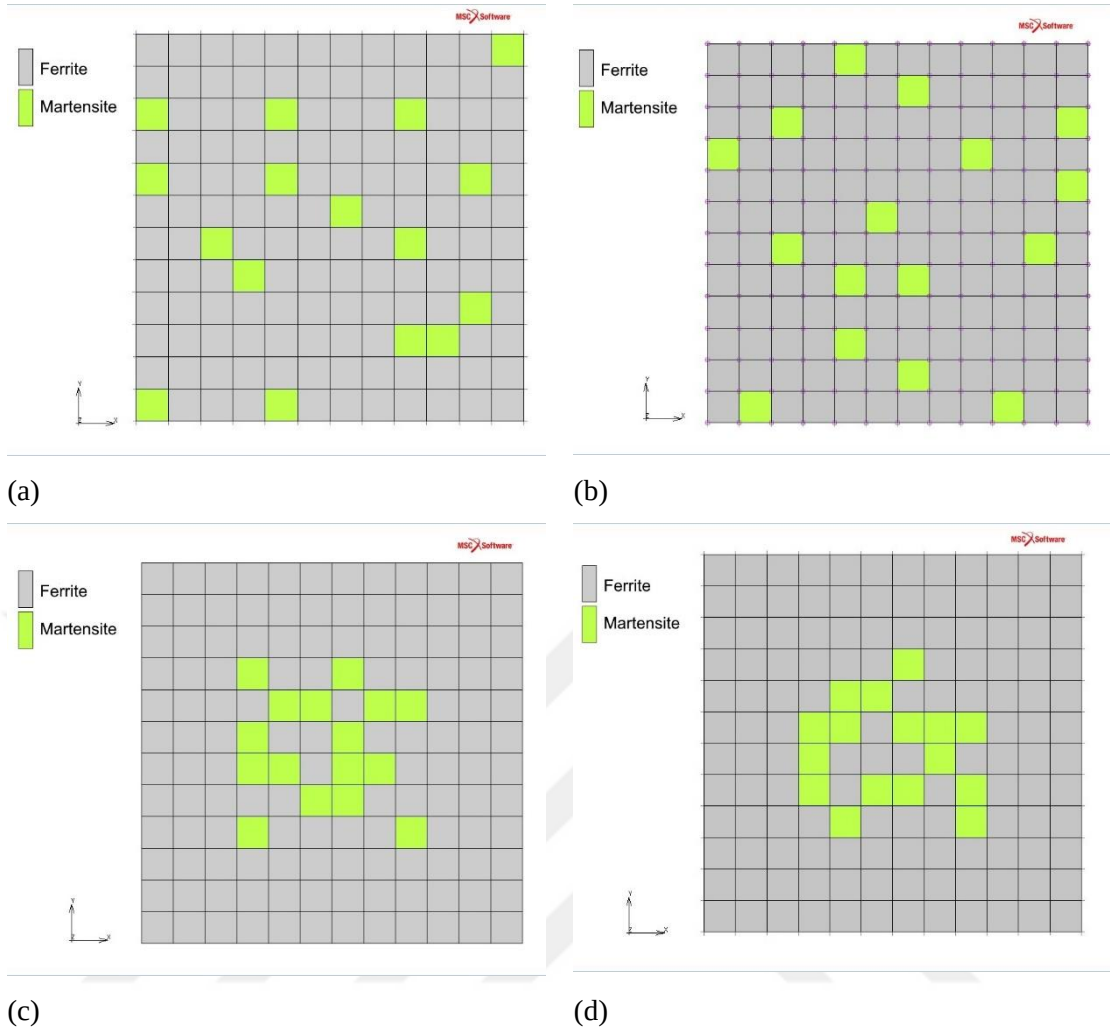
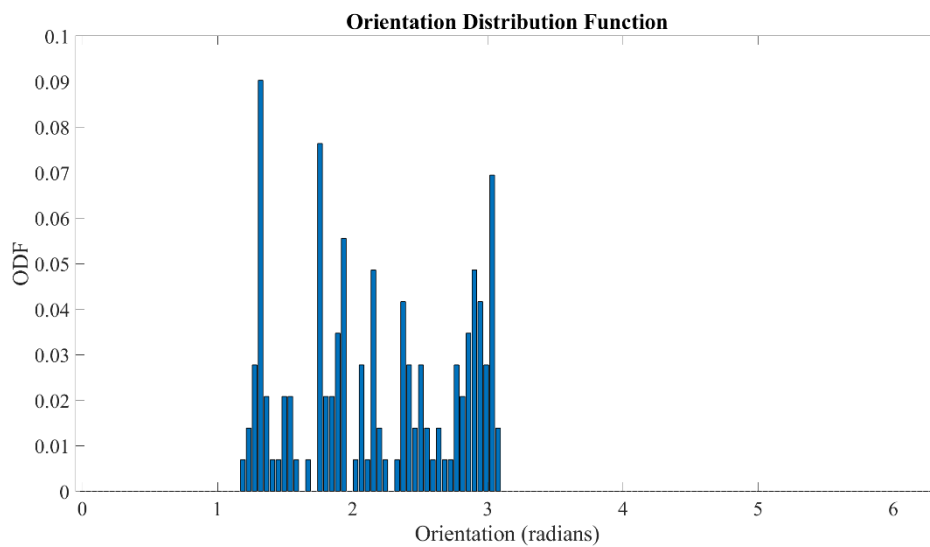
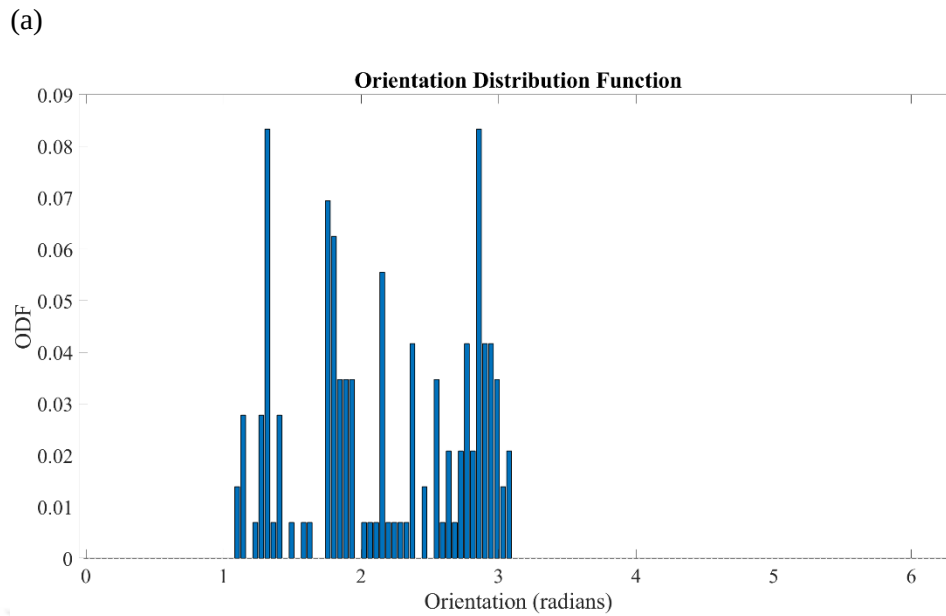


Figure 3-32 martensite distribution in each of the styles of the simulations, (a) scattered style D1, (b) scattered style D2, (c) concentrated style D3, (d) concentrated style D4, in all of the style the martensite volume fraction is maintained at 11.1%.





(b)

Figure 3-33 ODF functions distribution of (a) O1 case and (b) O2 case.

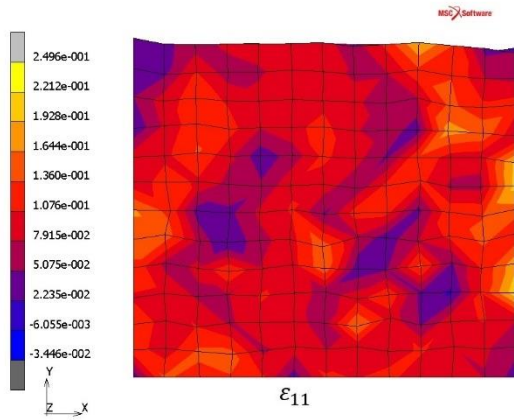
As mentioned earlier the O1 orientation case is more inhomogeneous than the O2 case. To quantify the level of dispersion or heterogeneity in the orientation distribution function (ODF) and compare it with other arrays of angles, we can use metrics that capture characteristics such as spread, concentration, or uniformity. One commonly used metric is the von Mises-Fisher concentration parameter, also known as the circular variance. The calculated von Mises-Fisher concentration parameters κ for the O1 and O2 cases are provided in Table 3-8. They are dimensionless numbers. The concentration parameter for O1 is smaller than O2 which implies less homogeneity.

Table 3-8 O1 and O2 von Mises-Fisher concentration parameter κ

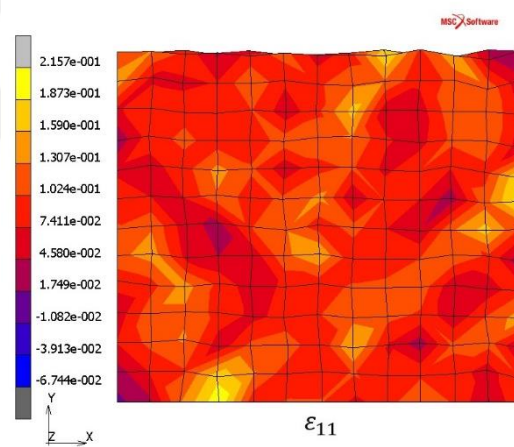
Case	Von Mises-Fisher concentration parameter
O1	322.496
O2	334.556

The strain distribution results achieved from the models are provided in Figure 3-34. The following pairs of figures (a-b, c-d, e-f, and g-h) in Figure 3-34 depict the outcomes obtained by running the models with an identical martensite-ferrite phase configuration. It can be observed that, although the distribution of the phases is identical within the pairs, in the case of the initial orientation, deliberately designed to have a more inhomogeneous

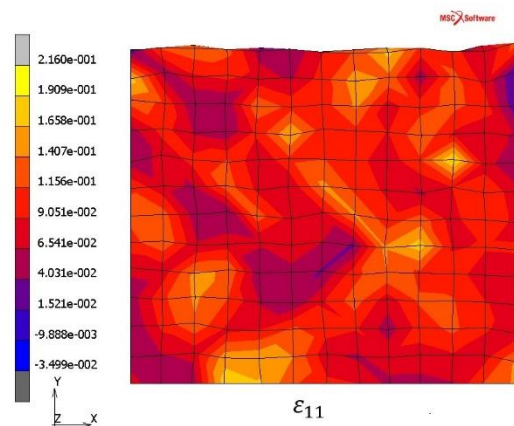
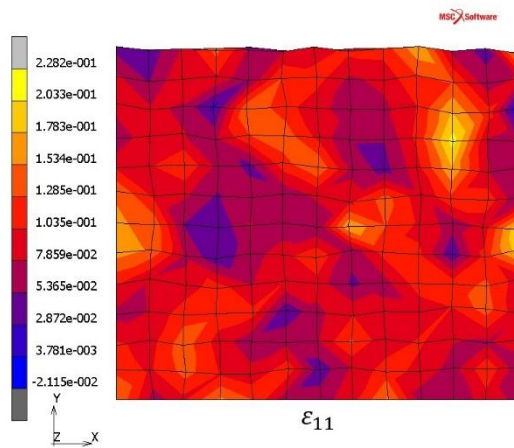
distribution, the heterogeneity in the plots has increased. This is because, by changing the initial orientation distribution of the ferrite elements, they experience varying mechanical constraints applied by their adjacent ferrite elements or their interface with the martensite elements. These constraints are related to the cubic stiffness properties of the ferrite elements and the significant discrepancies in the mechanical properties of the constituent phases. With the increase of the inhomogeneity in the initial orientation, the increase of MDD due to proceeding of the deformation gets accelerated because of more restrictive geometrical conditions more GND introduction to the elements. Eventually the increase of MDD leads to much more localized plastic deformation in specific spots and much more heterogeneity in the deformation distribution. This clearly shows the significant importance of the initial orientation on the heterogeneity of the deformation responses which implies that for determining the accurate spots of the heterogeneity peaks in addition to the precise phase configuration, the model needs to be supplied with the actual and representative initial orientation distribution.



(a)



(b)



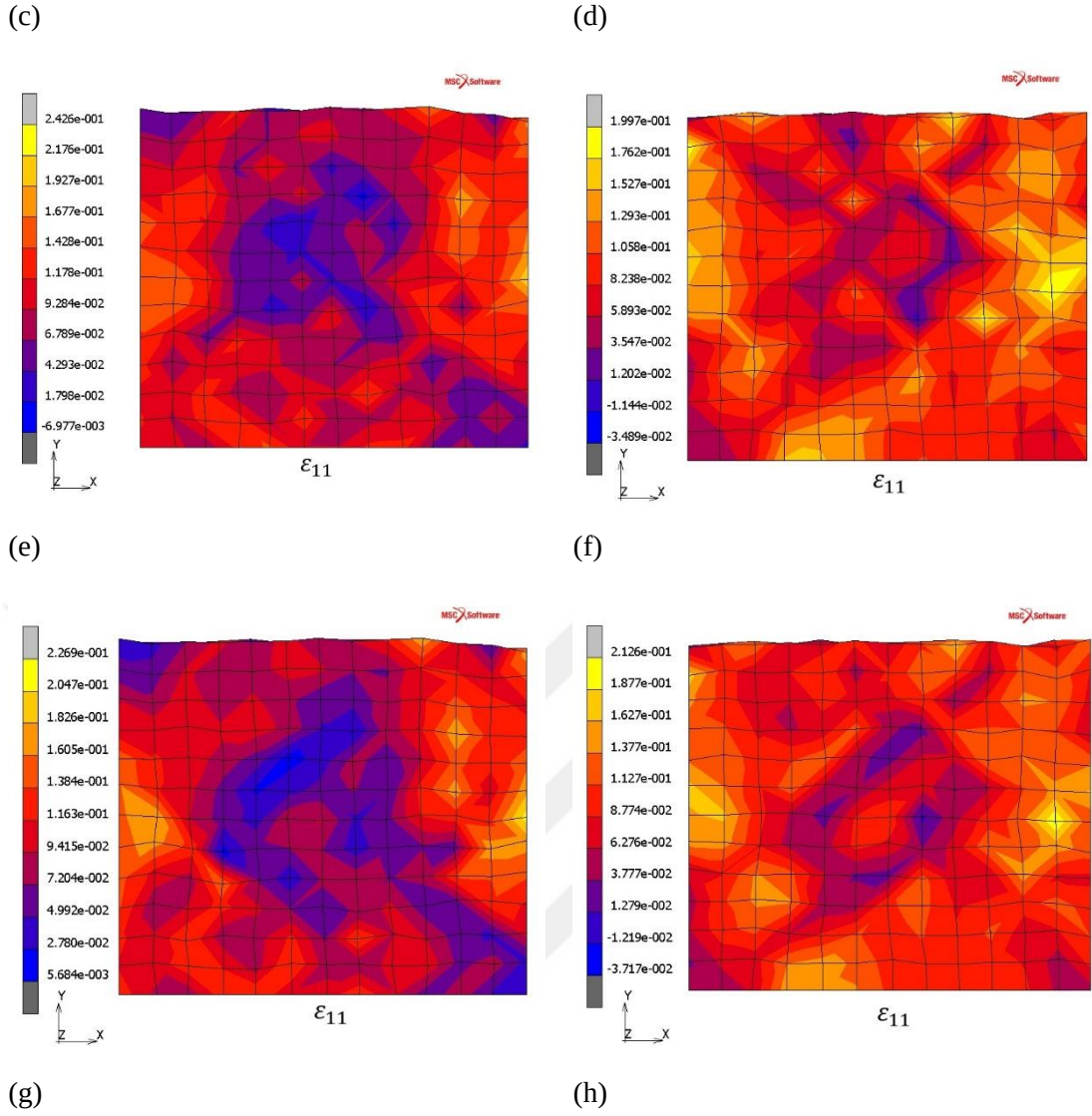
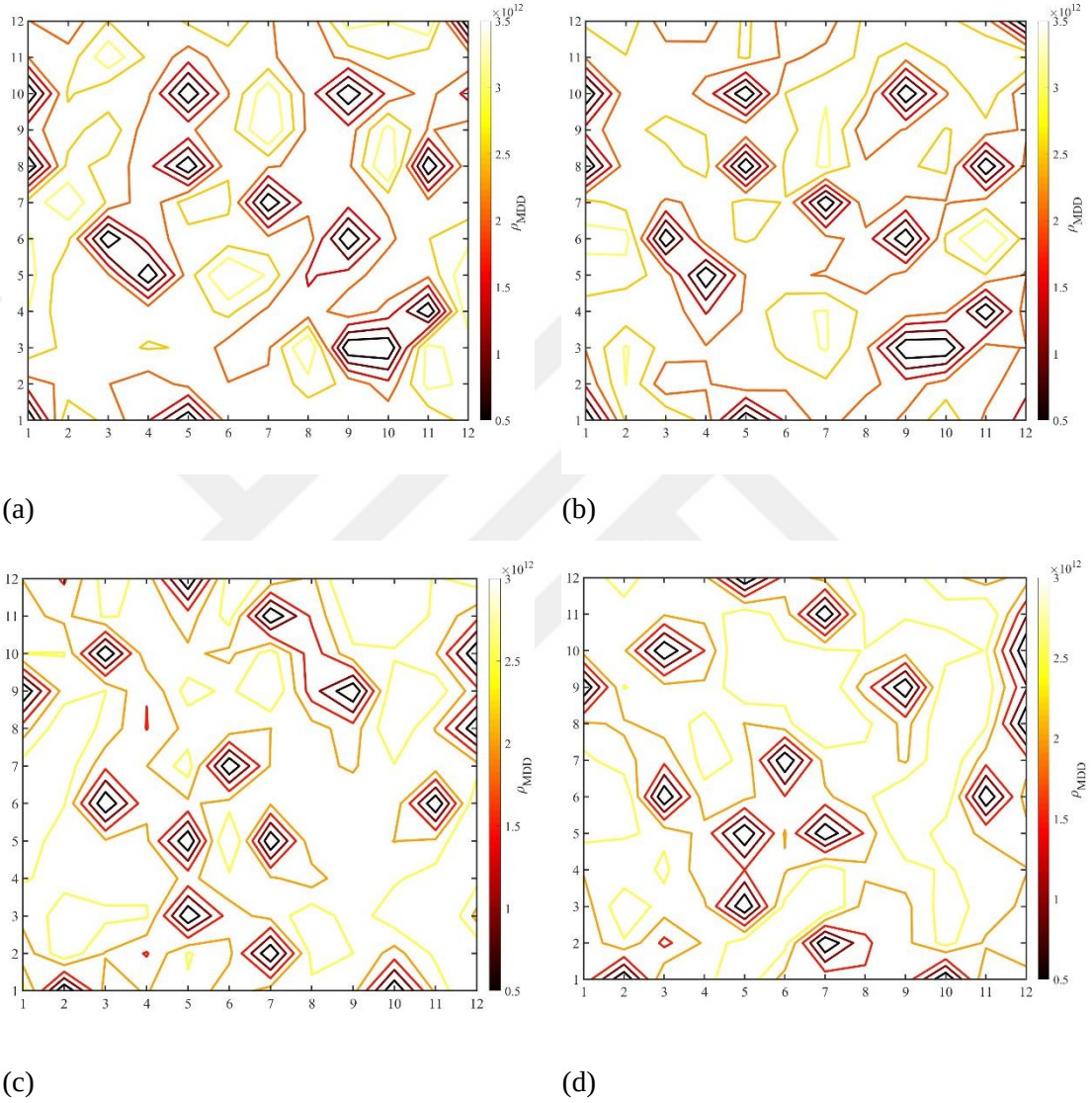


Figure 3-34 strain distribution in 12×12 RVE, (a) scattered D1O1 (distribution 1 orientation 1), (b) scattered D1O2, (c) scattered D2O1, (d) scattered D2O2, (e) concentrated D3O1, (f) concentrated D3O2, (g) concentrated D4O1, (h) concentrated D4O2.

As it was mentioned earlier, O1 and O2 initial orientation cases served as global sets in these simulations and remained unchanged during alterations in the martensite distribution. This means that in Figure 3-34, the simulations of the following pairs were conducted using the exact same initial orientation conditions (but with different martensite layout), a-c, b-d, e-g, and f-h. When comparing within each pair of strain distribution plots, certain hotspots of deformation overlap, revealing a similarity in the overall distribution of deformation and concentration points. This discovery suggests that the initial orientation has a more pronounced effect than the martensite configuration in pinpointing spots of deformation concentration.

The MDD distribution contour plots obtained for each of the simulation cases and presented in Figure 3-35. As observed, a direct correlation exists between the deformation plots and MDD distribution contour plots. There is a noticeable convergence and overlap between spots with maximum and minimum deformation values and those with the highest and lowest MDD concentrations.



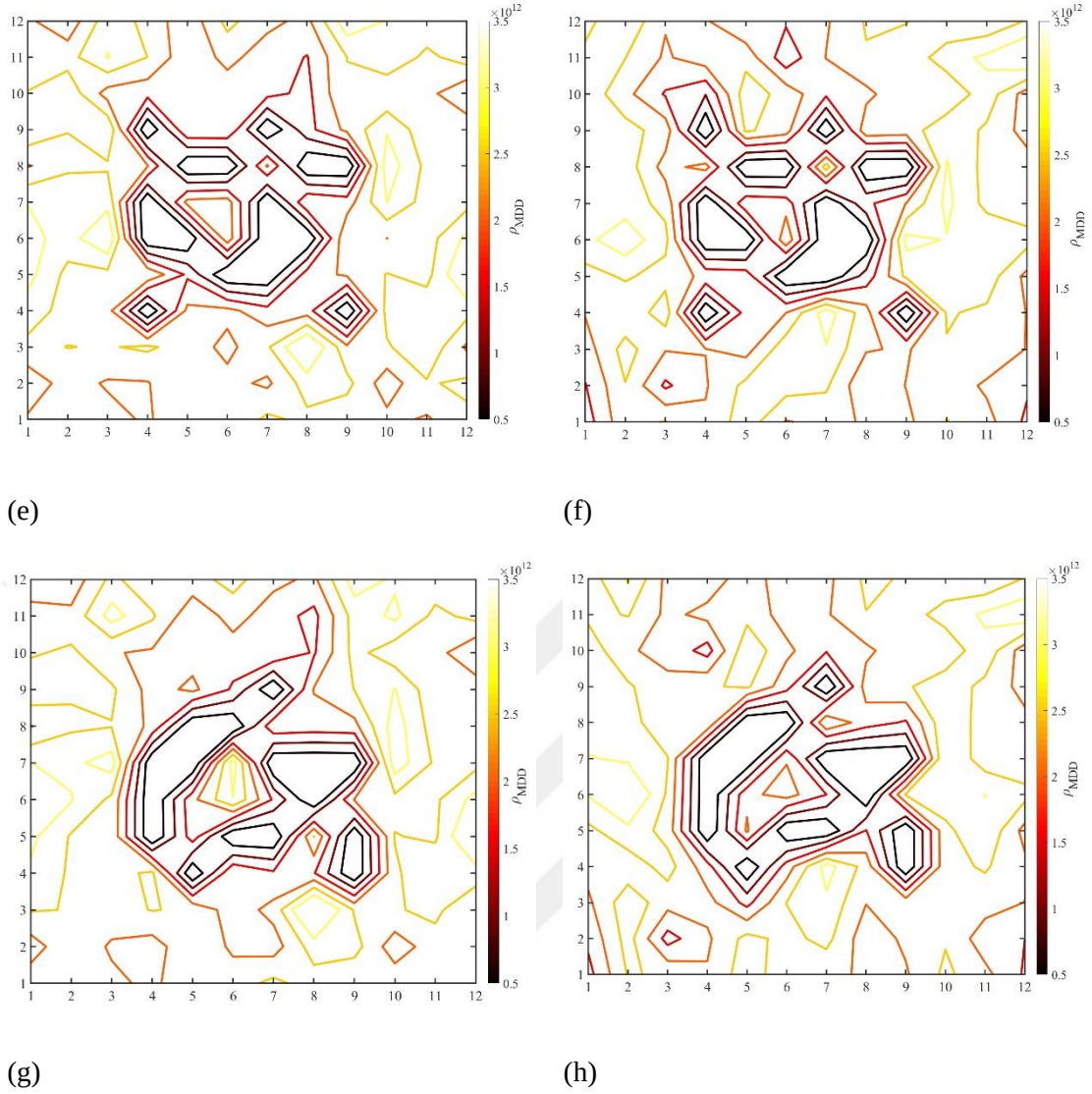


Figure 3-35 MDD distribution contour plots of the simulation combinations, (a) scattered D1O1, (b) scattered D1O2, (c) scattered D2O1, (d) scattered D2O2, (e) concentrated D3O1, (f) concentrated D3O2, (g) concentrated D4O1, (h) concentrated D4O2.

To establish a connection between the level of heterogeneity and the RVEs we expanded the martensite layout to include synthetic distribution cases, which can still be categorized as fully scattered distribution. The distributions of these synthetic models are coded as D5 (perfect), D6 (shifted), D7 (cross) and D8 (cross-stacked) as illustrated in Figure 3-36.

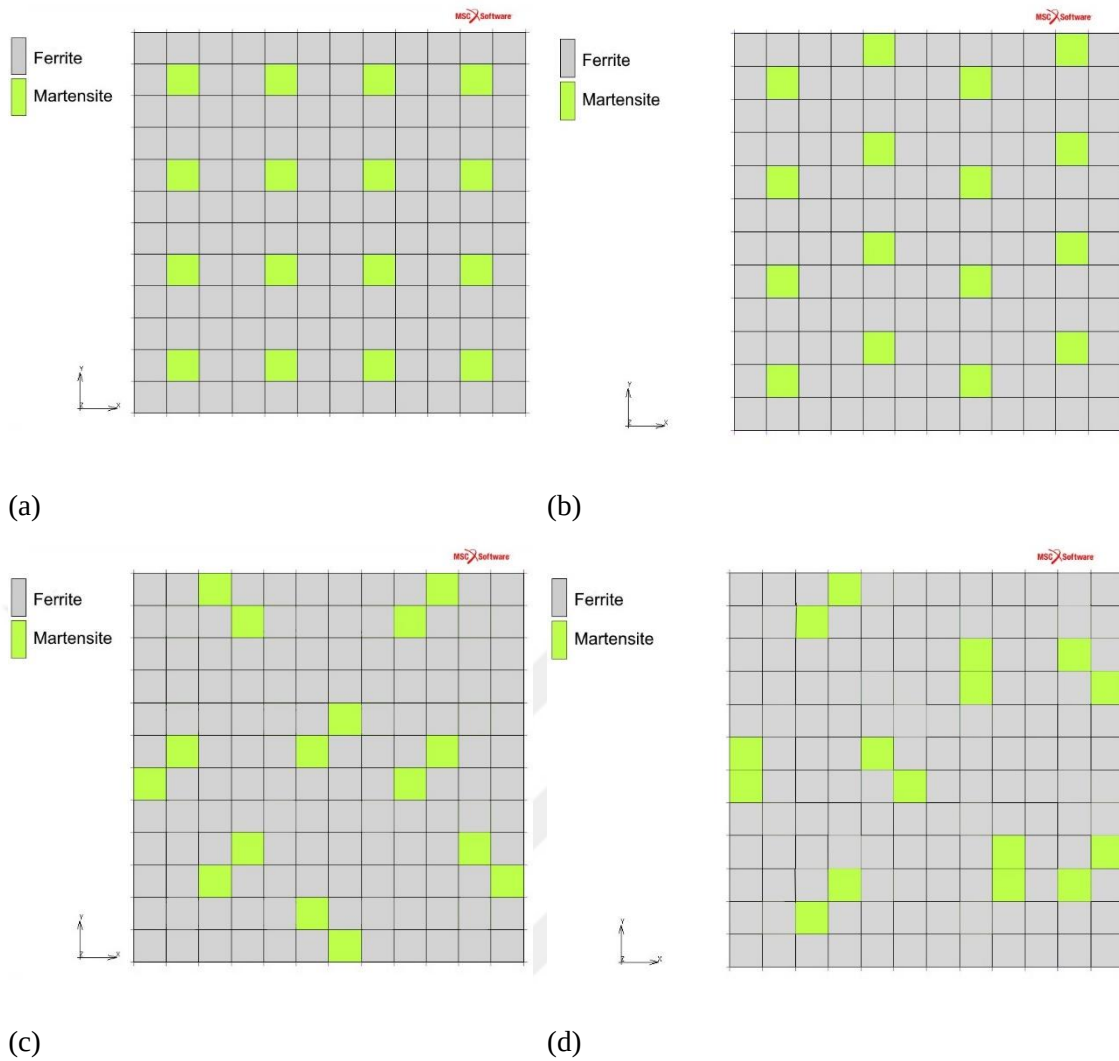


Figure 3-36 synthetic fully scattered martensite distribution, (a) D5 case, (b) D6 case, (c) D7 case, and (d) D8 case.

MCF for each of the martensite distribution styles in the RVEs are provided in Table 3-9.

Table 3-9 calculated MCF for the simulation combinations

Simulation case	MCF (μm)	Simulation case	MCF (μm)
D1	6.9434	D5	6.7082
D2	6.7996	D6	6.7454
D3	3.2930	D7	6.7823
D4	3.2198	D8	6.8482

Simulations conducted for D5, D6, D7 and D8 distribution cases under both O1 and O2 orientation totally 8 combinations. Table 3-10.

Table 3-10 resultant κ/MCF in each of the simulation combination

Combination	$\kappa/MCF \left[\frac{1}{\mu m} \right]$	Combination	$\kappa/MCF \left[\frac{1}{\mu m} \right]$
D1O1	46.44	D5O1	48.07
D1O2	48.18	D5O2	49.87
D2O	47.42	D6O1	47.80
D2O2	49.20	D6O2	49.59
D3O1	97.93	D7O1	47.54
D3O2	101.59	D7O2	49.32
D4O1	100.16	D8O1	47.09
D4O2	103.90	D8O2	48.85

To account for the effects of initial orientation and martensite distribution style, we divide κ parameter by the MCF factor and to measure the heterogeneity in the strain distribution we calculated the CV factor for each of the simulation cases (values provided in Table 3-10). The scattered plot of κ/MCF versus CV is displayed in Figure 3-37.

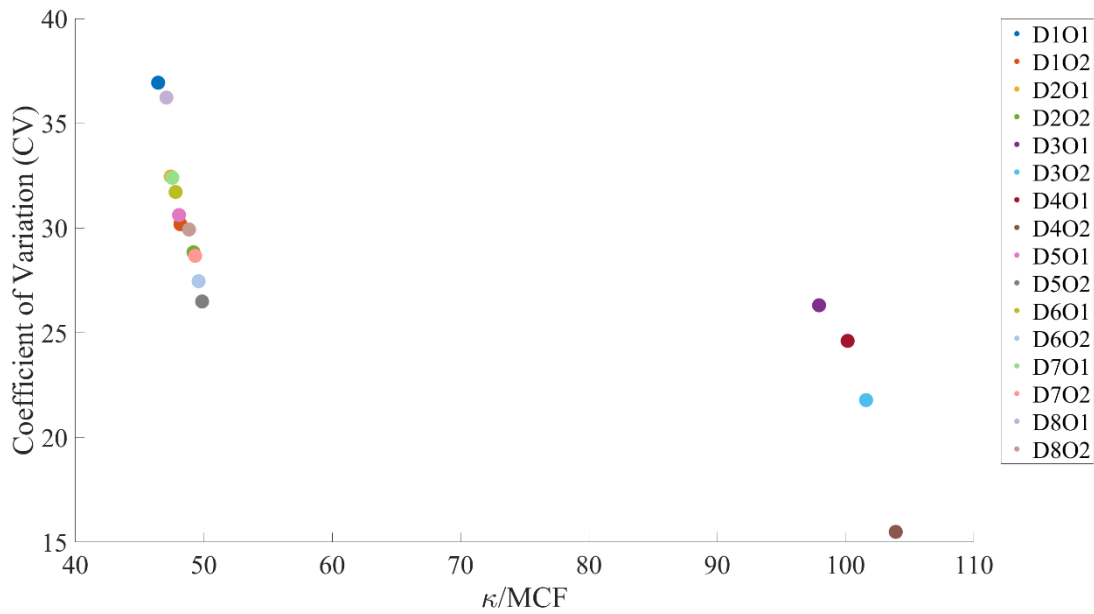


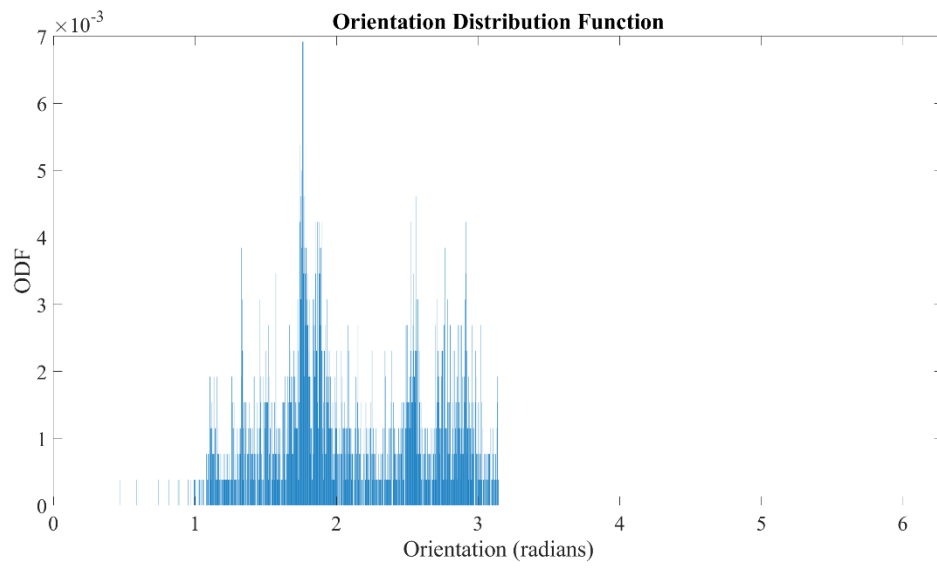
Figure 3-37 strain versus $MCF \times \kappa$

As can be clearly observed from the scatter plot, the product κ/MCF is an effective factor for distinguishing between simulation cases while considering their inhomogeneity potential. A lower value of κ/MCF signifies higher inhomogeneity. In the plot, the data points for D3O1, D3O2, D4O1, and D4O2 are positioned lower than other cases because they possess a more compact martensite distribution. This means that the CV factor for those cases are lower and they have shown less inhomogeneity in the lastic deformation behavior. The lower κ/MCF factor corresponds to fully scattered martensite cases, which are more prone to localization and exhibit more dispersed deformation hotspots and eventually their CV factor is relatively higher.

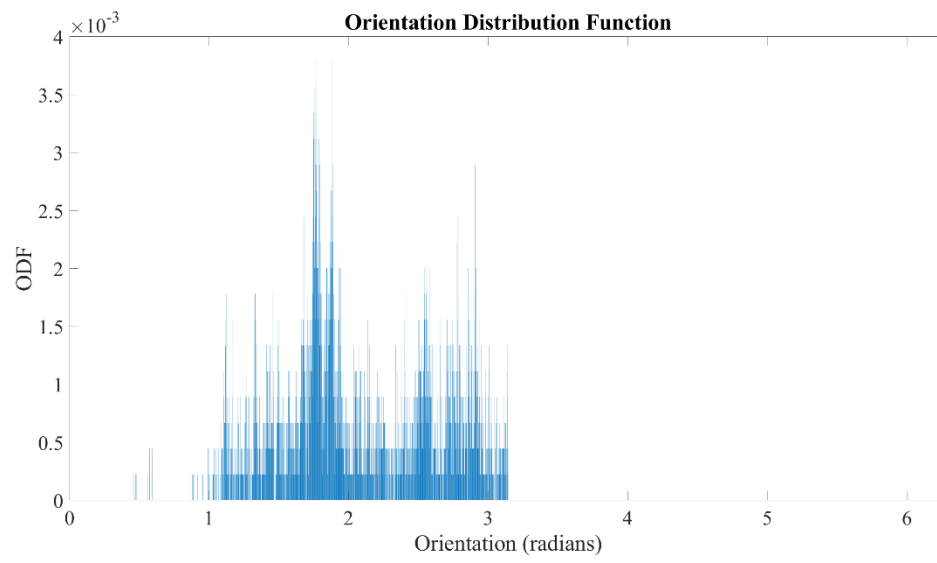
The product κ/MCF is referred to as well as serves as a determinant of homogeneity potentiality.

3.2.4 Element size and loading direction combined

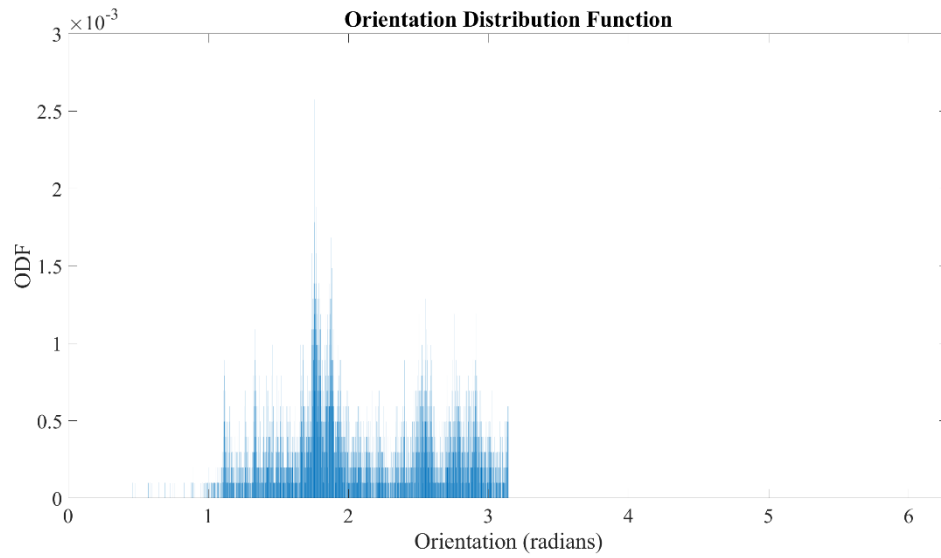
Up to this point, all calculations performed on the RVE model which was directly achieved from post processing our EBSD file with the $0.49\mu\text{m} \times 0.49\mu\text{m}$ cell size, which consists of 2601 cells. In order to address the potential influence of the chosen RVE cell size on simulation outcomes and to examine the impact of grain morphology on the fundamental physics of the simulation, we have extended our simulations to include both higher and lower grain size values. The selected RVE area remained constant throughout all simulations for grain sizes other than $0.49\mu\text{m} \times 0.49\mu\text{m}$. The new morphologies were tailored to maintain an equivalent orientation distribution compared to the main RVE by ensuring that their ODF deviation errors relative to that of the main case stayed below 5%. The smaller grain size cases that were tested are $0.37\mu\text{m} \times 0.37\mu\text{m}$ and $0.247\mu\text{m} \times 0.25\mu\text{m}$. This means that there were 4489 and 10100 elements forming the models with $0.37\mu\text{m} \times 0.37\mu\text{m}$ and $0.247\mu\text{m} \times 0.25\mu\text{m}$ cell sizes, respectively. The ODF of the main case and each of the lower cell size cases are shown in Figure 3-38.



(a)



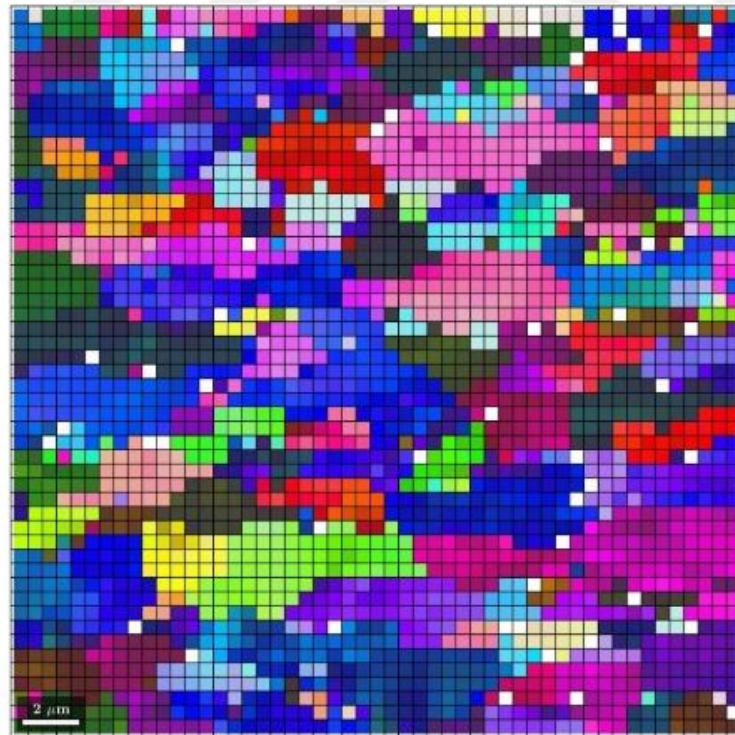
(b)



(c)

Figure 3-38 ODF of the applied RVEs, (a) $0.49\mu\text{m} \times 0.49\mu\text{m}$, (b) $0.37\mu\text{m} \times 0.37\mu\text{m}$, and (c) $0.247\mu\text{m} \times 0.25\mu\text{m}$

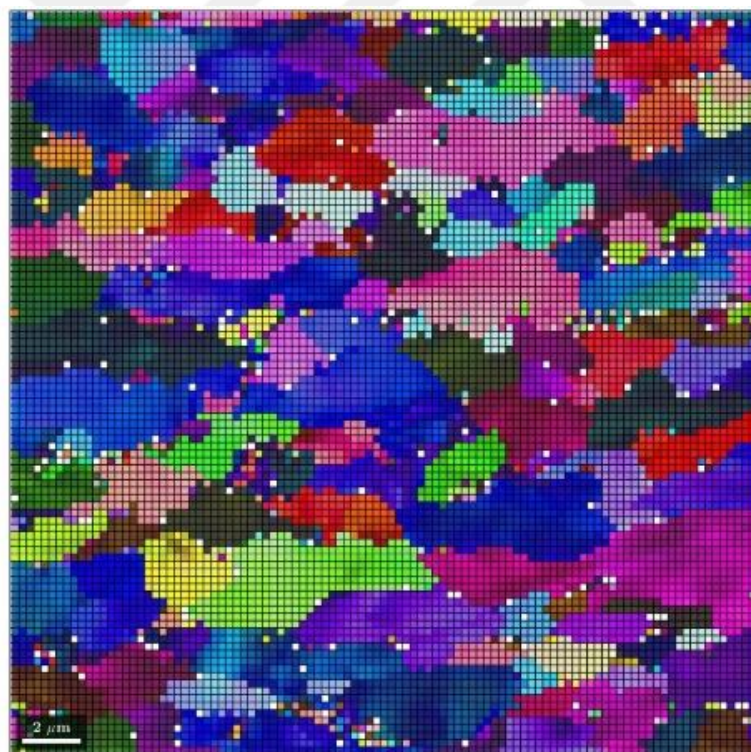
When comparing the ODFs of the RVEs illustrated in Figure 3-38, it can be observed that the distribution of the ODFs is consistent for different sizes. The morphology of the model employing three levels of discretization is depicted in Figure 3-39.



(a)



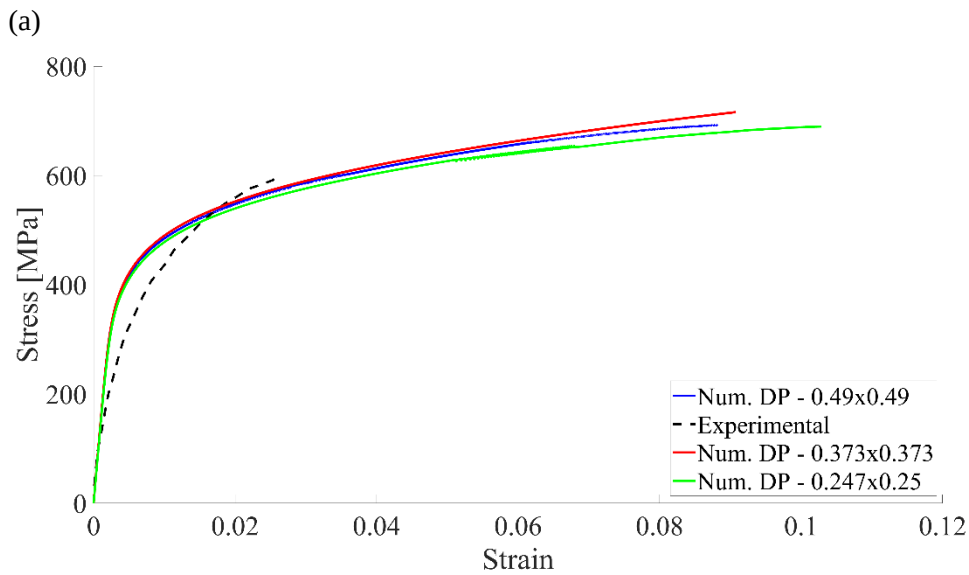
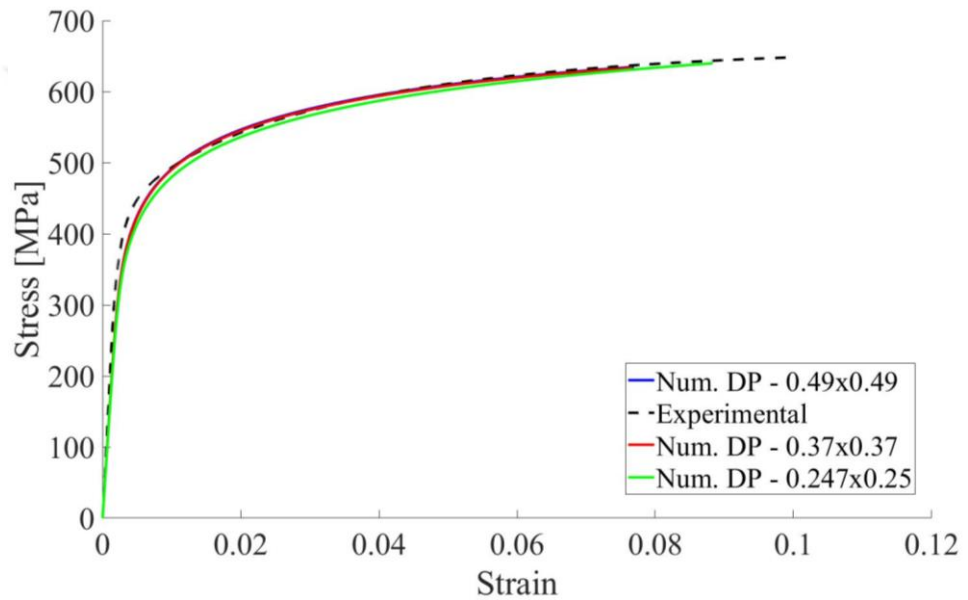
(b)



(c)

Figure 3-39 Three discretization level of the numerical models used in the simulations, (a) $0.49\mu\text{m} \times 0.49\mu\text{m}$ grain size, (b) $0.37\mu\text{m} \times 0.37\mu\text{m}$ grain size, (c) $0.247\mu\text{m} \times 0.25\mu\text{m}$ grain size discretization. The white squares represent the coordinates for the martensite phase ($\sim 4.4\%$ in all grids).

The fraction of martensite regions compared to the total number of elements in the models (RVE) for all three discretization cases closely featured the actual martensite-to-DP ratio observed in the complete EBSD dataset. Stress-strain curves under tensile, and compression loading conditions for the three discretization levels, specifically the RVE grain sizes of $0.49\mu\text{m} \times 0.49\mu\text{m}$, $0.37\mu\text{m} \times 0.37\mu\text{m}$, and $0.247\mu\text{m} \times 0.25\mu\text{m}$, are presented in Figure 3-40.



(b)

Figure 3-40. Stress-strain graphs for the three discretization levels in HYPEL-CPJ2, (a) tensile case, (b) compression case, experimental result is taken from Figure 2 reported by Ghaei et al. [154].

As demonstrated by Figure 3-40, it is clear that the level of grain discretization does not exert a substantial influence on the mechanical behavior of the DP (HYPELA-CPJ2) models. The impact of grain refinement on the material's strain hardening behavior was insignificant in all three loading cases, including tensile and compression, for the DP models. It's worth mentioning that reducing the grain size, which incurs significant computational expenses, does not yield a significant divergence in terms of capturing the experimental response.

With considering the strain distribution plots in the $0.49\mu\text{m} \times 0.49\mu\text{m}$, $0.37\mu\text{m} \times 0.37\mu\text{m}$, and $0.247\mu\text{m} \times 0.25\mu\text{m}$ simulation cases it can be observed that the decrease in the element size has slightly increased the maximum and minimum stress and strain gaps, indicating a slight escalation in the concentration of deformation. The strain distribution plots are displayed in Figure 3-41.

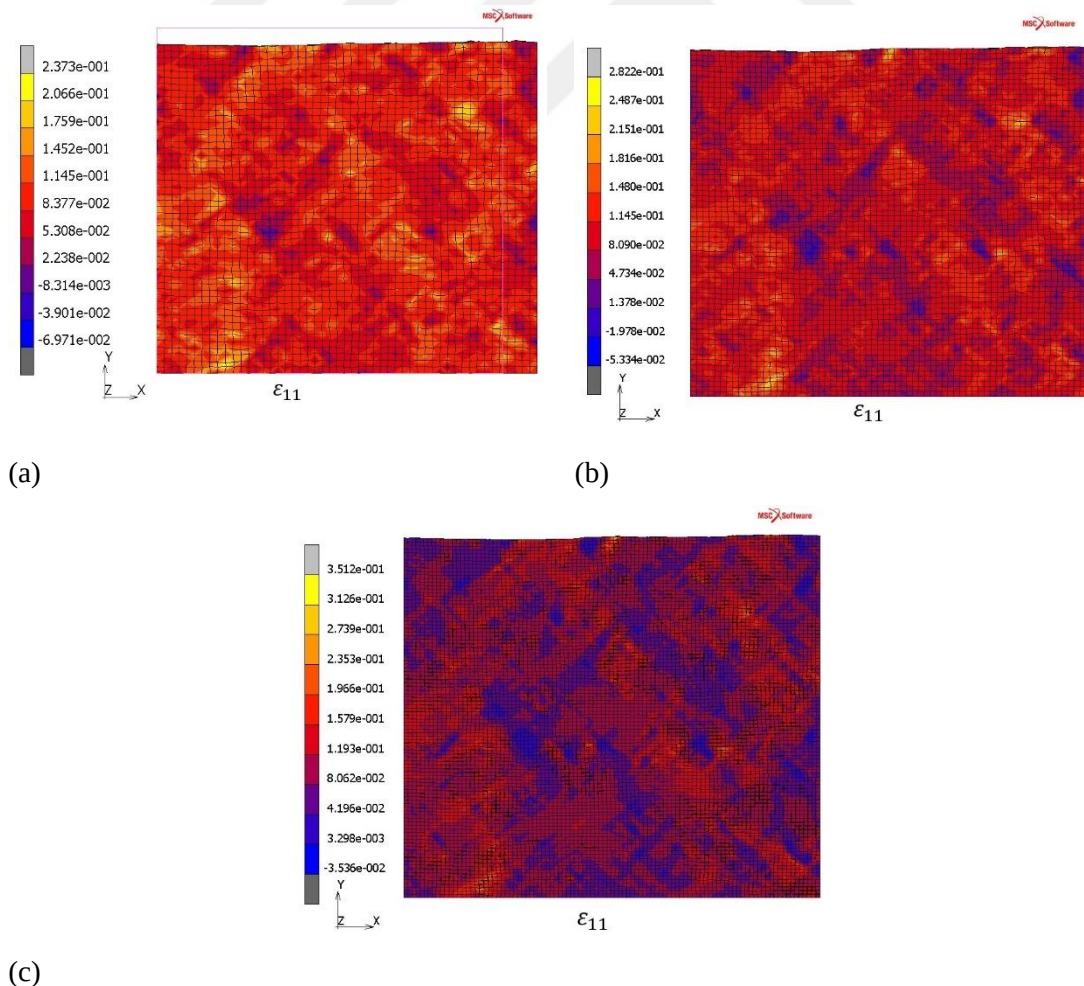


Figure 3-41 strain and stress distribution plots in the simulated RVEs, (a) strain in $0.49\mu\text{m} \times 0.49\mu\text{m}$ case, (b) strain in $0.37\mu\text{m} \times 0.37\mu\text{m}$ case, (c) strain in $0.247\mu\text{m} \times 0.25\mu\text{m}$ case.

With a closer look at the strain and stress distribution plots, it can be realized that similar heterogeneous distribution patterns exist in the RVEs with different element sizes. For a clearer understanding of the reader in Figure 3-41 the concentration zones are marked and juxtaposed. As seen, the heterogeneity patterns exhibit a similar composition, but with smaller elements, they become more detailed and, in some areas, broken down into several hotspots.

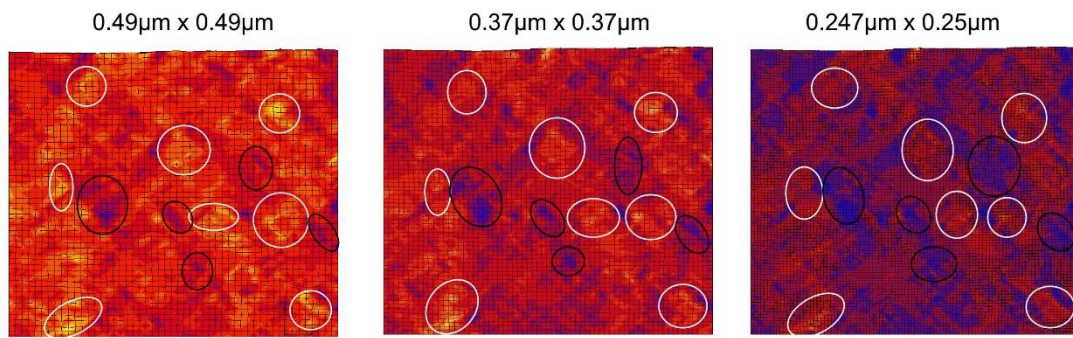
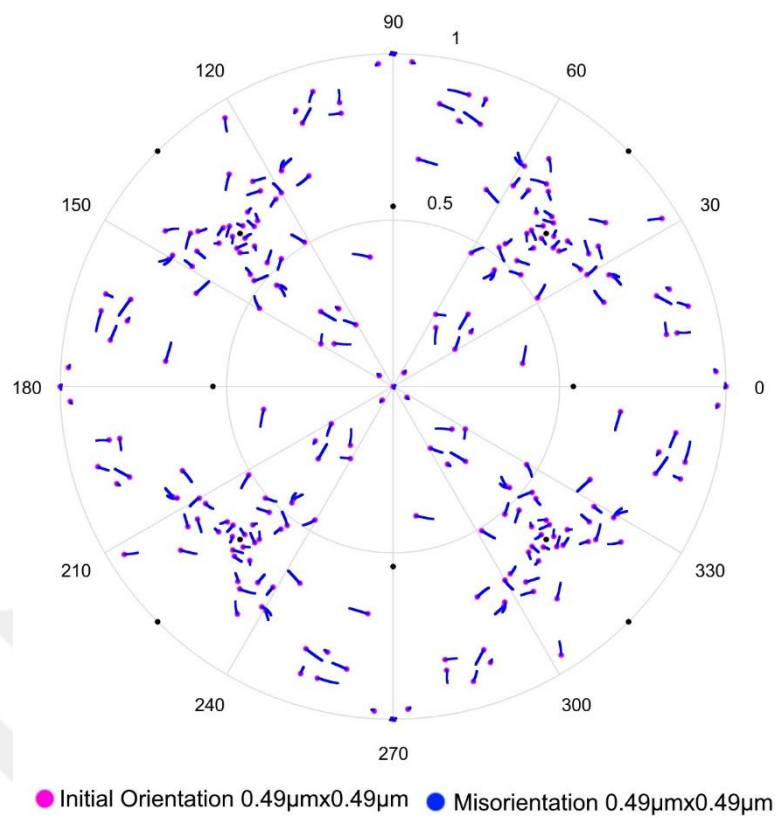
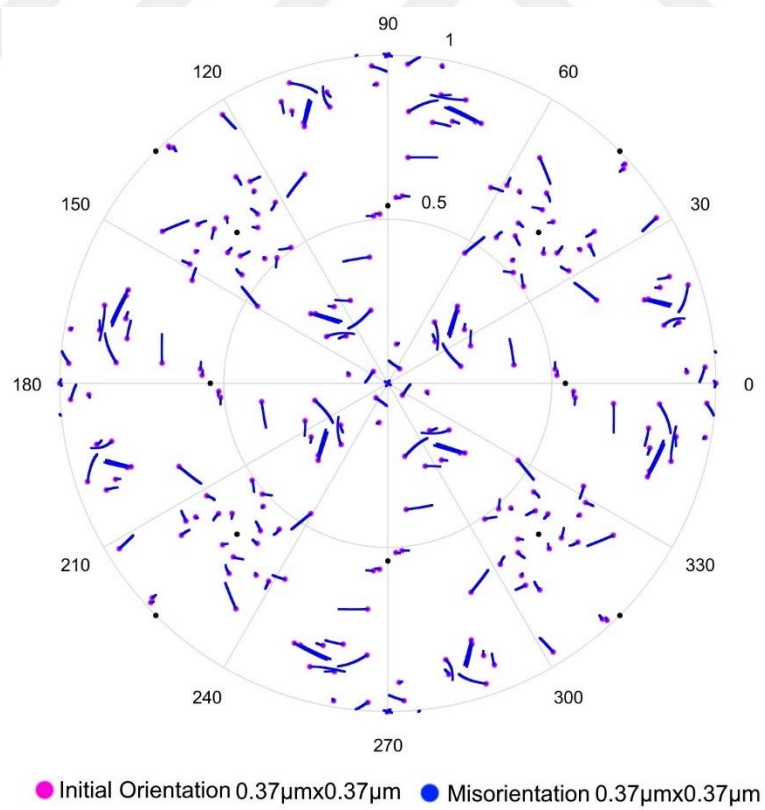


Figure 3-42 localized and concentrated responses are marked, white marks encircle the maximums and black ones encircle the minimums in strain distributions.

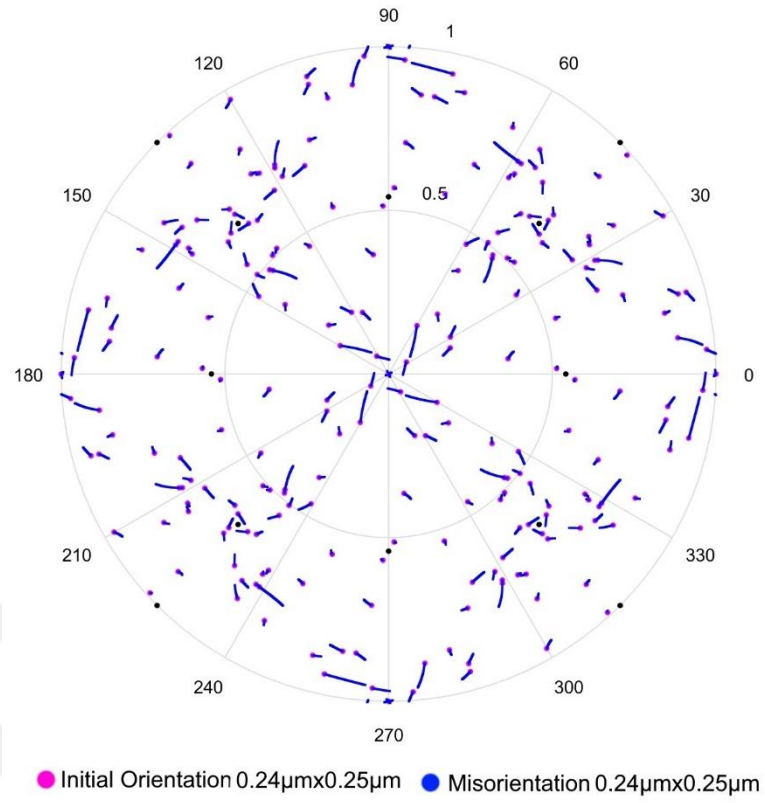
Similar to the increase of maximum and minimum deformation results gaps with the decrease of the elements sizes, the comparison of the misorientation pole figures align with this result and indicate more discrepancies in the misorientations as element refinement increases. The misorientation pole figures of the RVEs are provided in Figure 3-43.



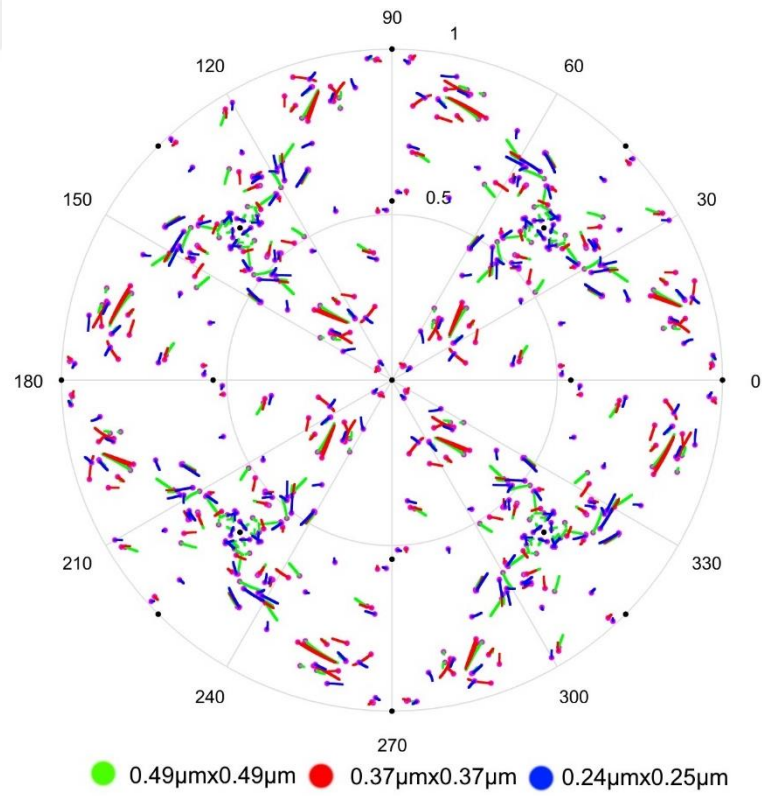
(a)



(b)



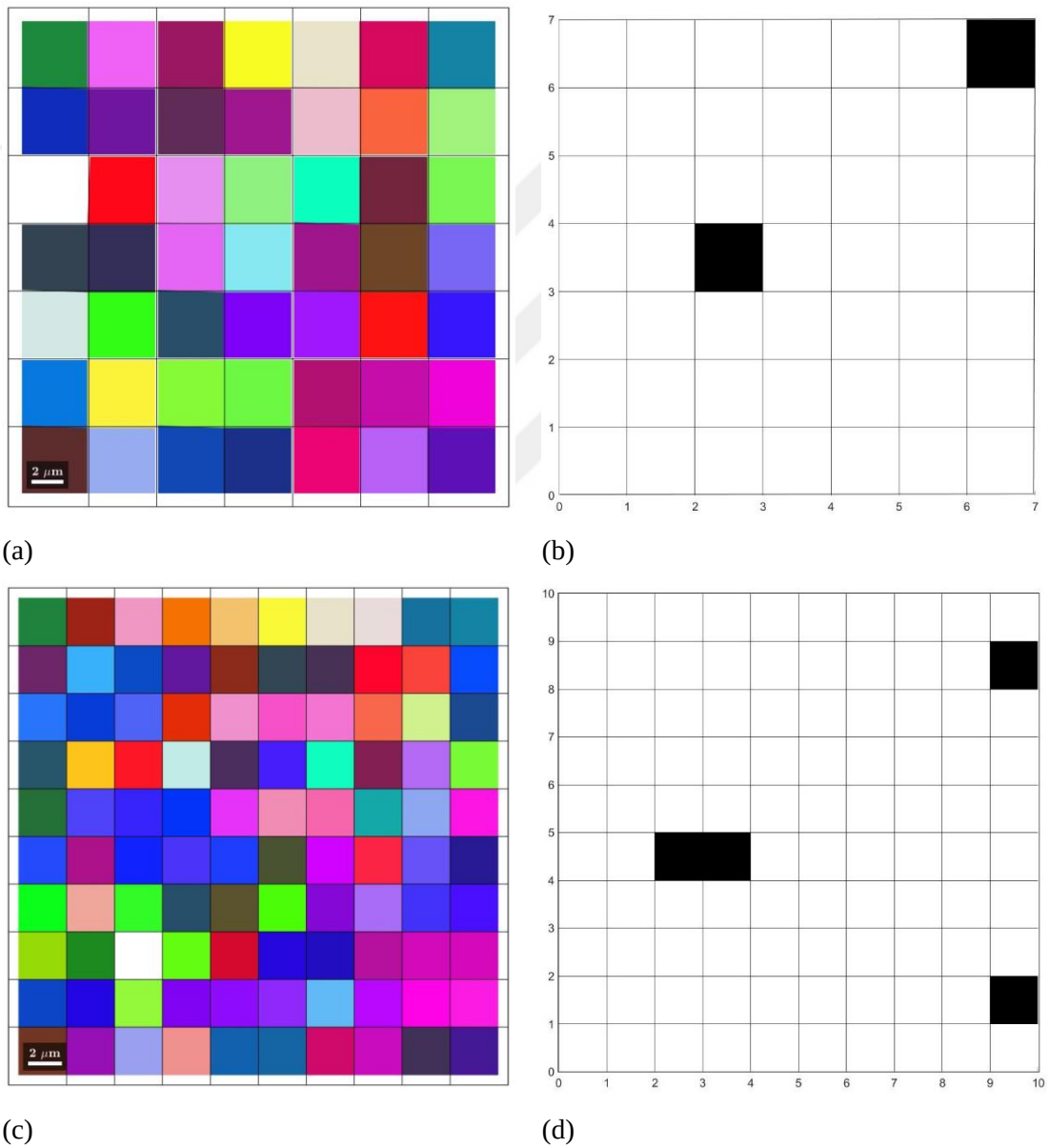
(c)

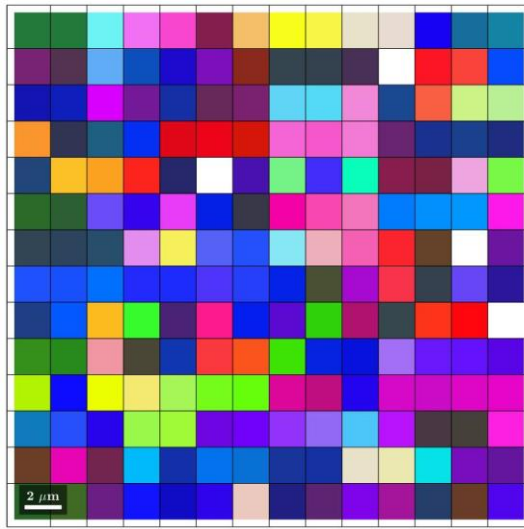


(d)

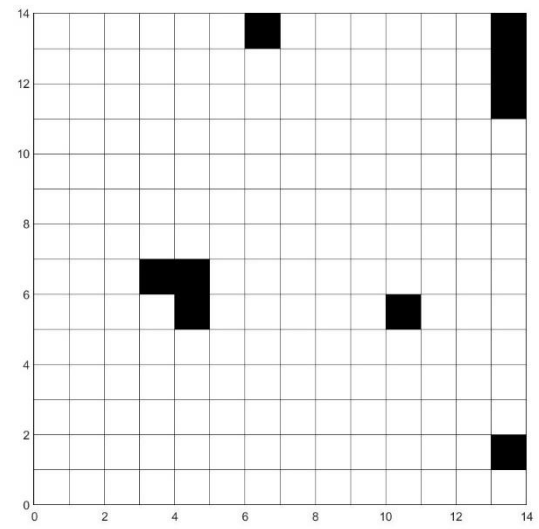
Figure 3-43 Misorientation pole figures of the simulated RVEs, (a) 0.49 μm $\times$ 0.49 μm case, (b) 0.37 μm $\times$ 0.37 μm case (c) 0.24 μm $\times$ 0.25 μm case, and (d) superimposed.

The simulation involving different morphologies has been expanded to include larger element sizes. This time, four distinct morphologies were derived from the same RVE area while ensuring that the ODF deviation error values remained below 15%. Simulations with larger RVE sizes were performed using combinations of $3.57\mu\text{m} \times 3.57\mu\text{m}$, $2.50\mu\text{m} \times 2.50\mu\text{m}$, and $1.78\mu\text{m} \times 1.78\mu\text{m}$. The martensite volume fraction in all cases was maintained within the range of 4% to 5%, consistent with our main RVE case of $0.49\mu\text{m} \times 0.49\mu\text{m}$. The morphologies of martensite distributions of the large-sized RVEs are provided in Figure 3-44.





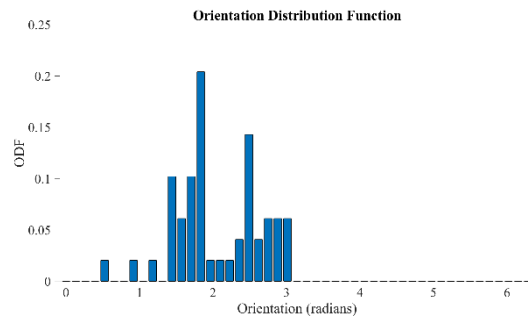
(e)



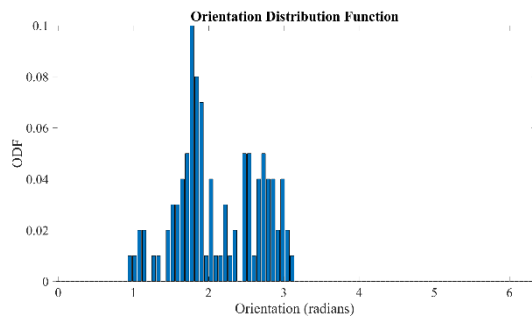
(f)

Figure 3-44 large-sized RVE morphology and martensite distribution illustration, (a) $3.57\mu\text{m} \times 3.57\mu\text{m}$ (7×7) case morphology, (b) 7×7 martensite distribution with 4.1% volume fraction (c) $2.50\mu\text{m} \times 2.50\mu\text{m}$ (10×10) case morphology, (d) 10×10 martensite distribution with 4% volume fraction, (e) $1.78\mu\text{m} \times 1.78\mu\text{m}$ (14×14) case morphology, and (f) 14×14 martensite distribution with 4.6% volume fraction.

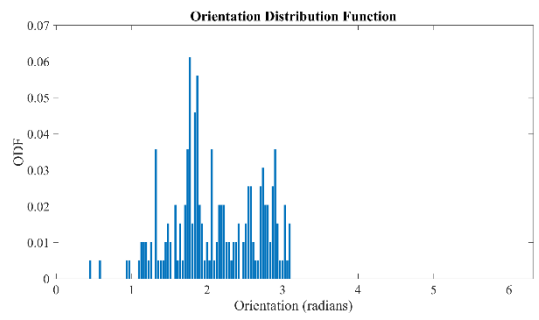
The ODF of each of the larger-sized RVEs is depicted in Figure 3-45. Upon comparing these ODFs with the main $0.49\mu\text{m} \times 0.49\mu\text{m}$ ODF case, it becomes evident that the modeling platform has effectively maintained the associativity between the simulation cases.



(a)



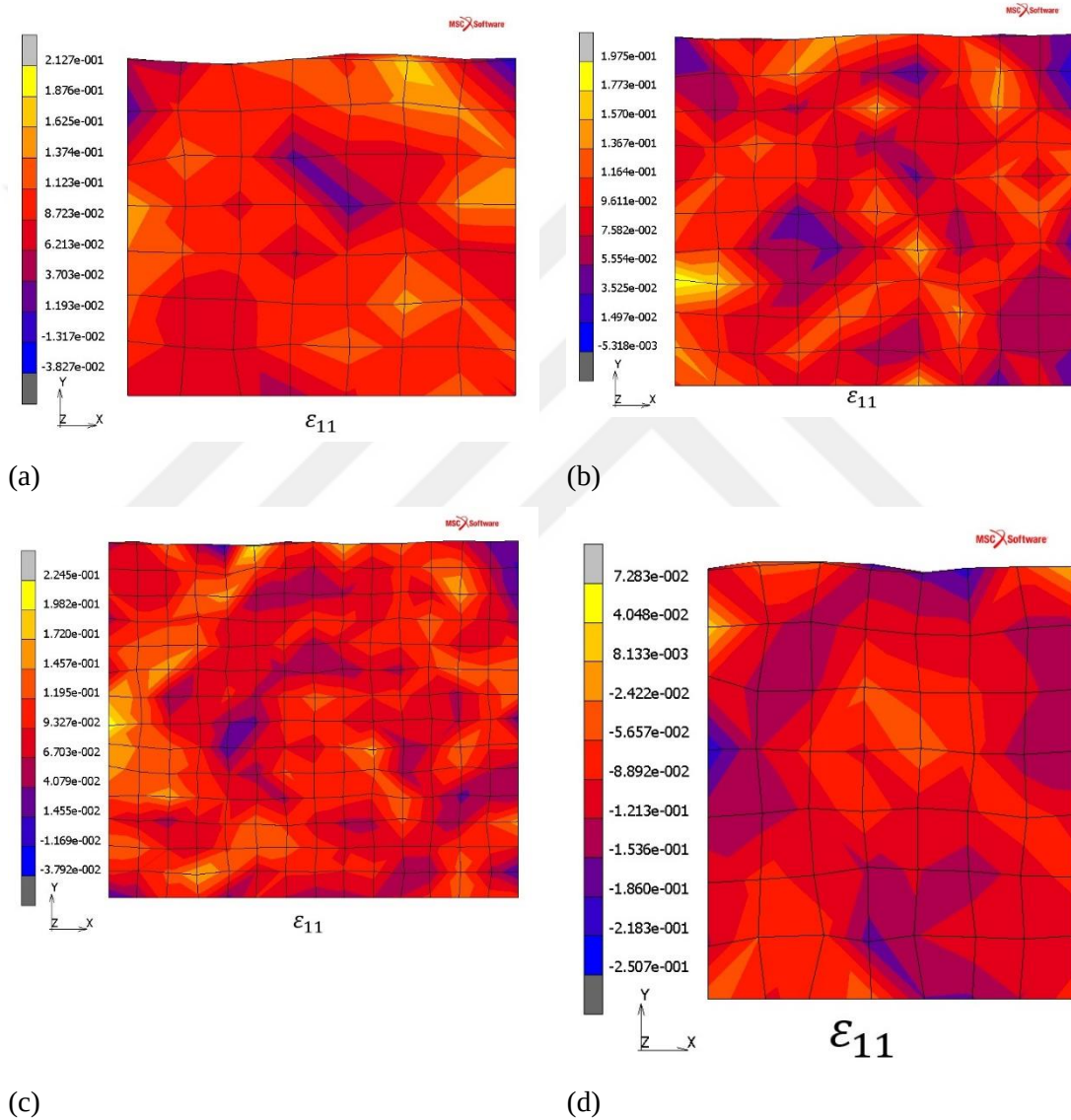
(b)



(c)

Figure 3-45 ODFs of the larg-sized RVEs, (a) 7×7 case, (b) 10×10 case, and (c) 14×14 case.

The resultant strain distribution of the RVEs is provided in Figure 3-46 for both tensile and compression cases. Considering the changes in the composition of inhomogeneity under both tensile and compression loading conditions as the element size decreases, it becomes apparent that the distributions become more detailed, and the pattern in larger RVEs becomes more fragmented.



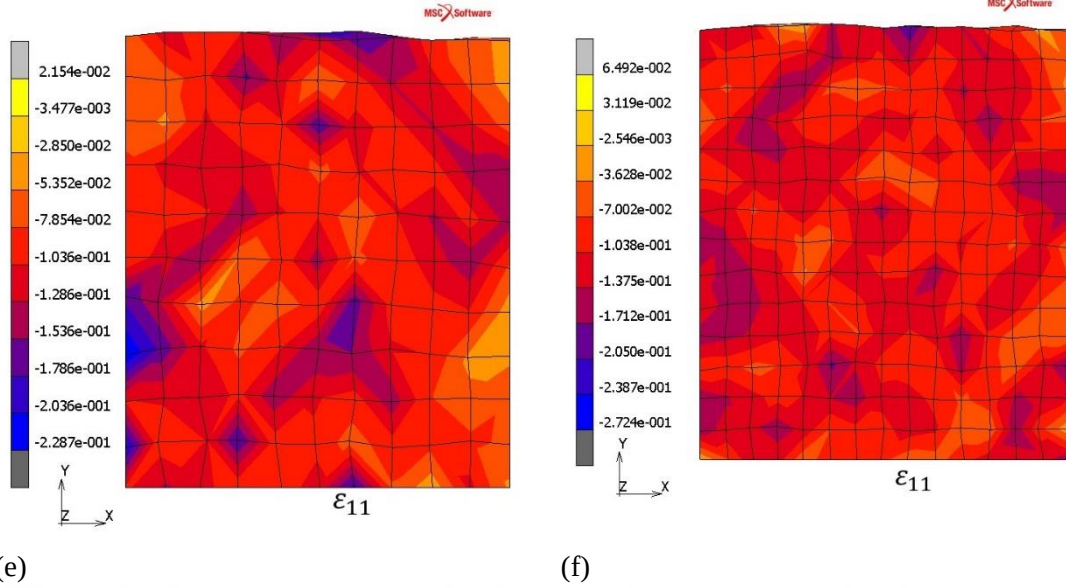
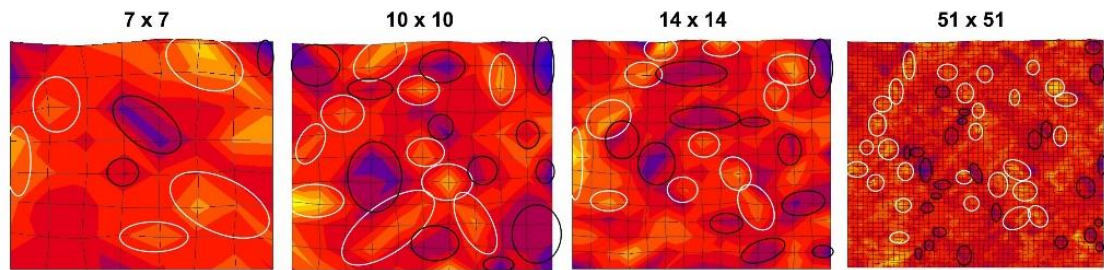
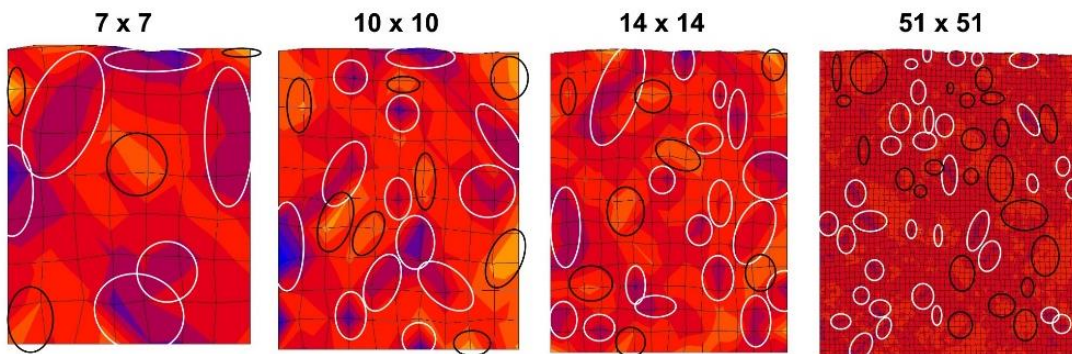


Figure 3-46 resultant strain distribution of the simulations with the developed large-sized RVEs, (a) tensile 7×7 , (b) tensile 10×10 , (c) tensile 14×14 , (d) compression 7×7 , (e) compression 10×10 , and (f) compression 14×14 .

To elucidate this in Figure 3-47 the gradual evolution and fragmentation of the inhomogeneity composition is displayed. This evolving trend demonstrates that our RVEs consistently represent the same area, accurately capturing the ODF and phase morphology.



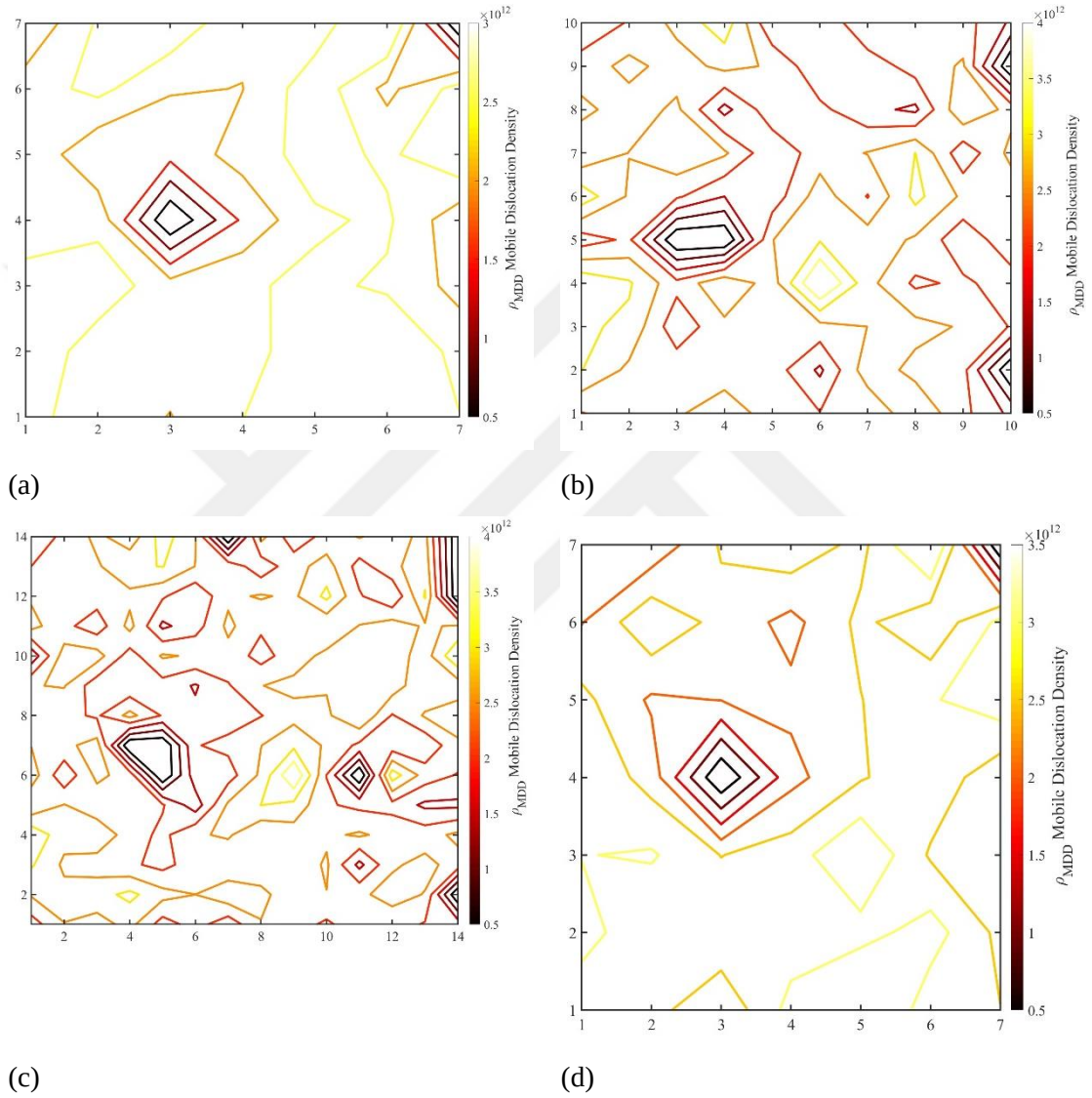
(a)



(b)

Figure 3-47 deformation hotspots distribution evolution with the decrease of elements size, maximum points are shown in using white ellipses and minimum zones are marked by black ones (a) tensile case, and (b) compression case.

The distribution pattern of deformation hotspots in both tensile and compression cases aligns with the contour plots extracted from the post-processed results output files. The contour plots are illustrated in Figure 3-48.



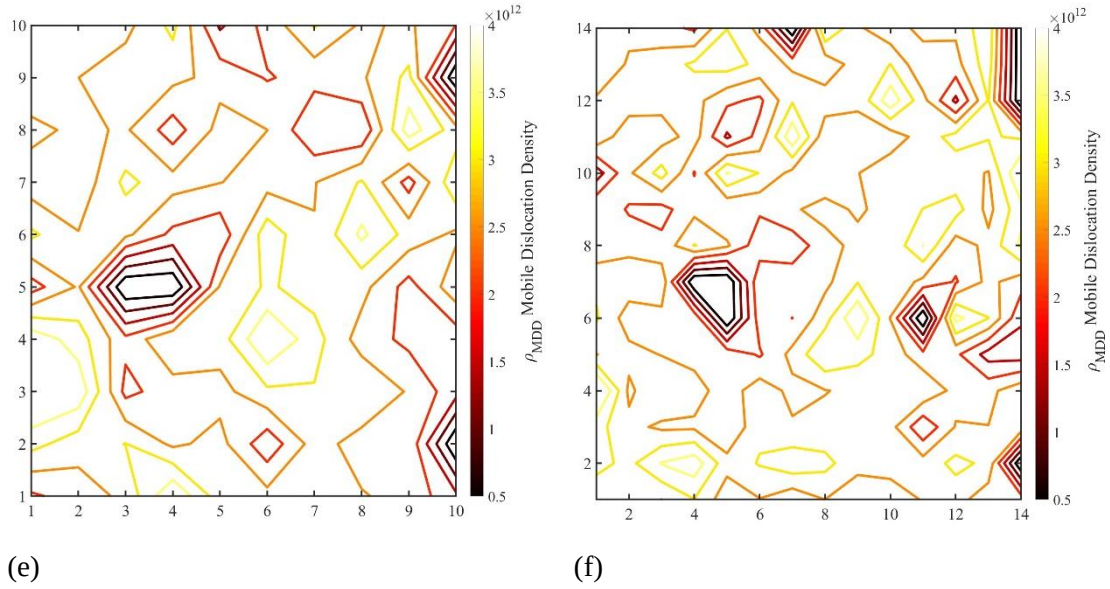


Figure 3-48 MDD distribution contour plots of the large size RVEs, (a) tensile 7×7 , (b) tensile 10×10 , (c) tensile 14×14 , (d) compression 7×7 , (e) compression 10×10 , (f) compression 14×14 .

To establish the relationship between MDD values, misorientation, and RVE size, the scatter plot of double-axis data points is presented in Figure 3-49 using the obtained results. The data points of compression are situated at higher levels in the plots compared to the tensile cases. This observation indicates the influence of non-Schmid stress contribution to the growth of deformation through retexturing and subsequent enhancement of dislocation density. There is an upward trend in dislocation density and misorientation with the refinement of elements (decrease in element size). This is attributed to the increased dispersion of the martensite regions and consequently increased ferrite-martensite interfaces within the RVEs.

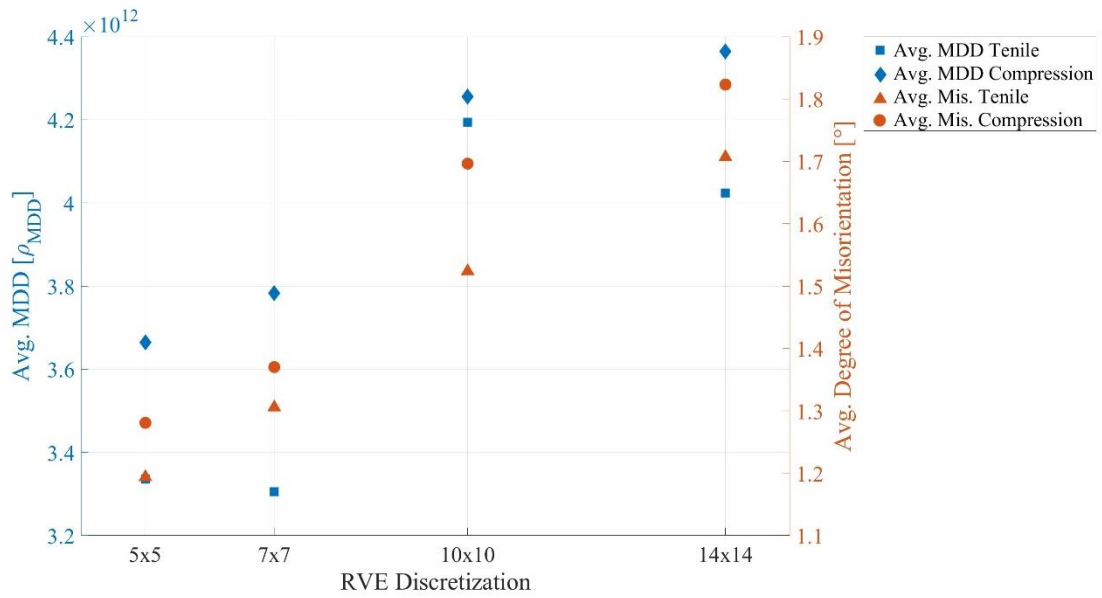


Figure 3-49 double axis scatter plot average MDD, average degree of misorientation and RVE size.

By plotting the maximum MDD value spotted in the tensile and compression deformation cases, it can be clearly seen that how deformation direction poses different restrictive conditions on the elements and eventually a different growth rate of MDD as deformation progresses. The corresponding plot is shown in Figure 3-50.

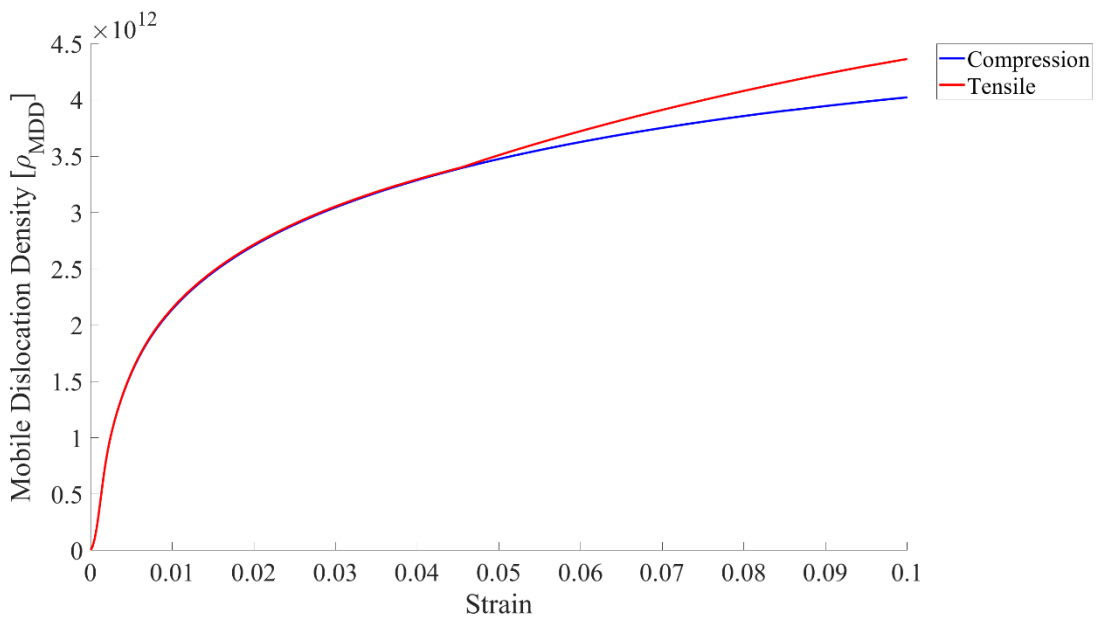


Figure 3-50 maximum MDD evolution in tensile and compression cases.

To compare the real case results in different discretization level their corresponding MCF, κ and CV factors are calculated and provided in Table 3-11.

Table 3-11 MCF, κ and CV factors for each of the element size in the real simulation cases.

Discretization case	MCF [1/ μm]	κ	κ/MCF [1/ μm]	CV (Tensile)	CV (Compression)
7 $\times$ 7 Model	5.00	130.63	26.12	36.50	28.21
10 $\times$ 10 Model	5.80	245.00	42.17	30.54	26.62
14 $\times$ 14 Model	8.29	424.36	58.15	28.12	21.07
51 $\times$ 51 Model	27.89	6450	231.26	25.58	12.38
67 $\times$ 67 Model	37.25	11428	306.79	20.86	12.22
101 $\times$ 100 Model	55.95	25048	447.68	16.12	9.82

In Figure 3-51 the scattered plot of the potential of inhomogeneity, κ/MCF , versus coefficient of variation, CV, is provided for all discretization levels. As observed, decreasing the element size leads to an increase in inhomogeneity, but this increase begins to saturate at a critical level, specifically in the 51 $\times$ 51 discretization case. This observation indicates that our developed model construction methodology enables us to effectively select the most efficient element size for simulations, optimizing them for both time and computational cost. Consequently, the combination of κ/MCF and the CV factor proves to be a powerful tool for evaluating the efficiency of a simulation case in comparison to others.

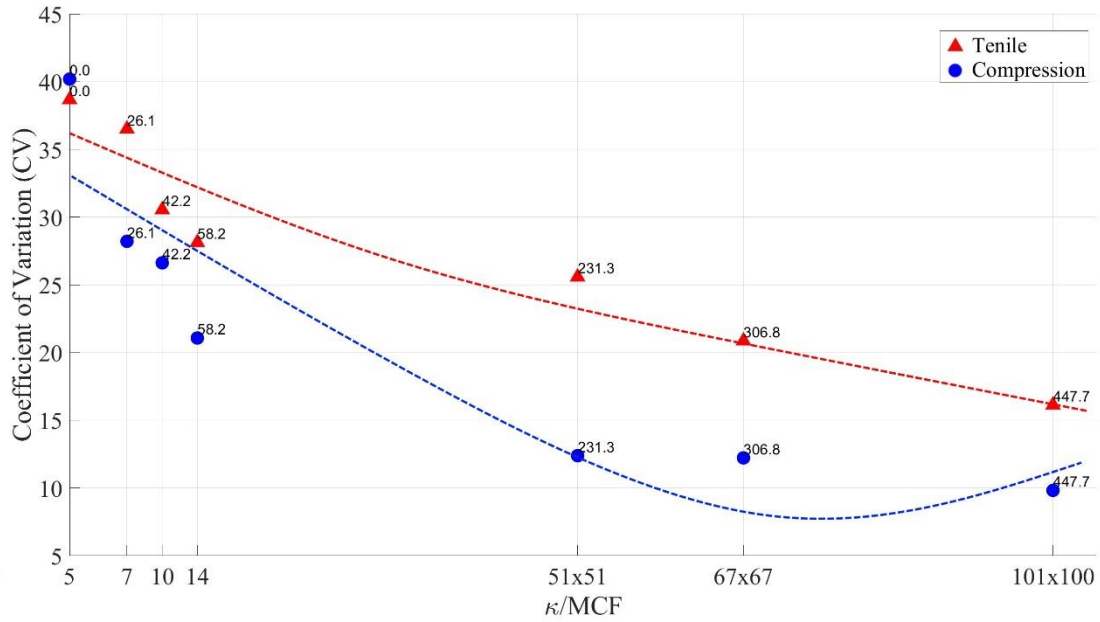
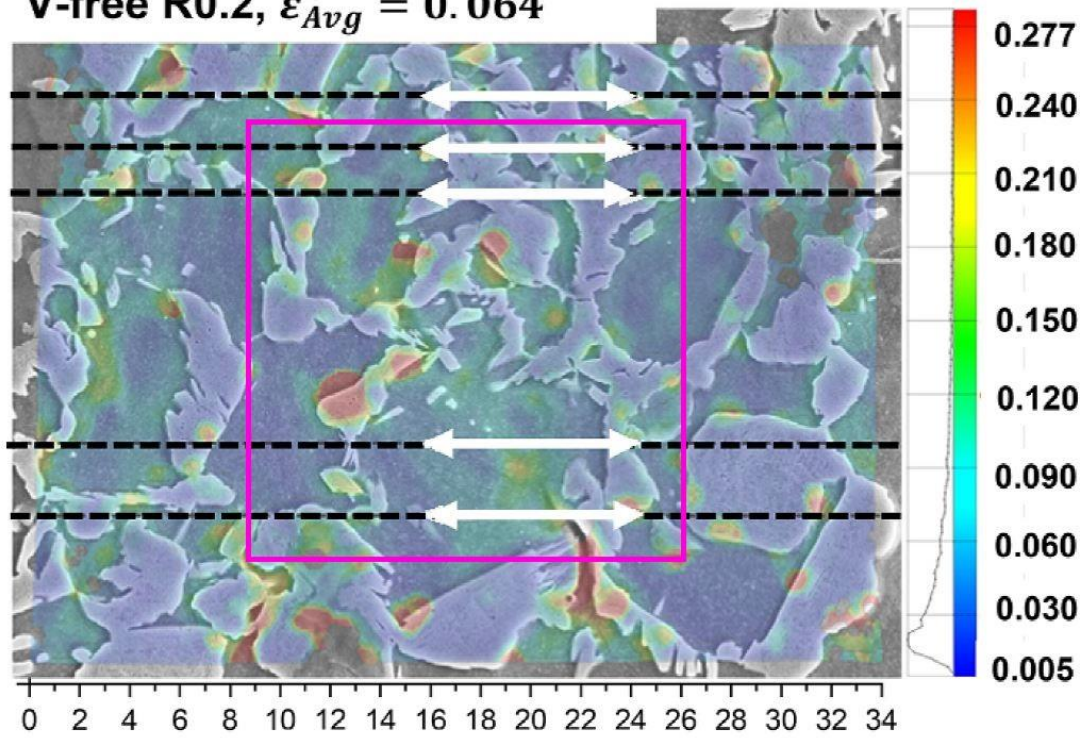


Figure 3-51 κ/MCF versus CV for main realistic models with different element sizes

3.2.5 Strain distribution assessment

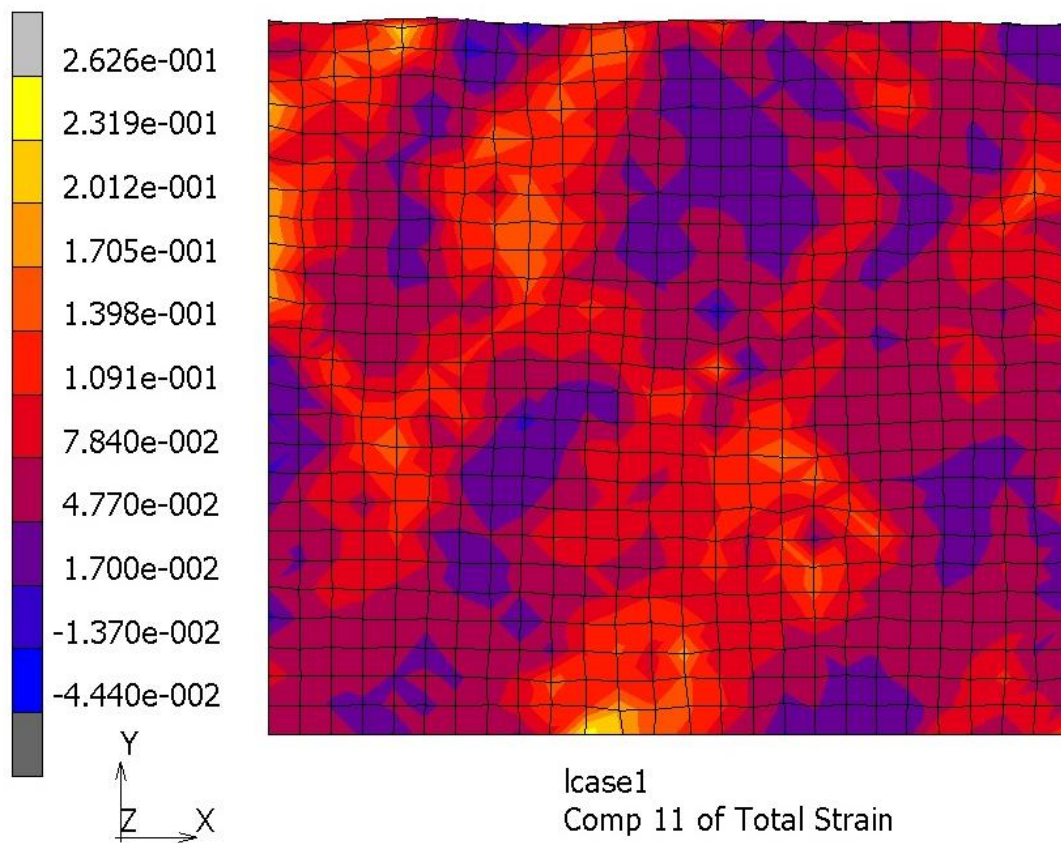
To validate the model's precision in replicating observed experimental patterns, in addition to the stress-strain curves examination we also visualized the contour bands depicting the evolution of the total strain component on the deformed model. This visualization was then compared with the reference experimental data, as depicted in Figure 3-52. Upon analyzing the strain fields obtained from experiments and simulations, it becomes clear that the simulations effectively replicate the overall strain distribution observed in both microstructures. For instance, in both microstructures, the primary plastic strain is concentrated within the ferritic regions, which are positioned predominantly in the middle and lower regions of the investigated area in both simulations and experiments. Upon thorough examination, it becomes apparent that the simulations do not precisely reproduce specific sharply localized strain bands observed in the experimental measurements. Instances of such discrepancies are noticeable in Figure 3-52 (a), where concentrated high-strain areas (indicated in red) are situated at the top.

V-free R0.2, $\epsilon_{Avg} = 0.064$



(a)

MSC Software



lcase1
Comp 11 of Total Strain

(b)

Figure 3-52 Contrasting the experimental strain distribution from reference [153] with the numerical results: (a) displays the experimental distribution of average strain, with a highlighted purple box marking the selected area, while (b) exhibits the corresponding total strain distribution within the deformed model.

In the simulation graph, martensite regions are associated with their respective deformed elements and overlaid onto the strain distribution to effectively illustrate how this phase influences strain localization. The RVE model achieves enhanced consistency with the experimental data by aligning the orientation distribution of ferrite phase grains.

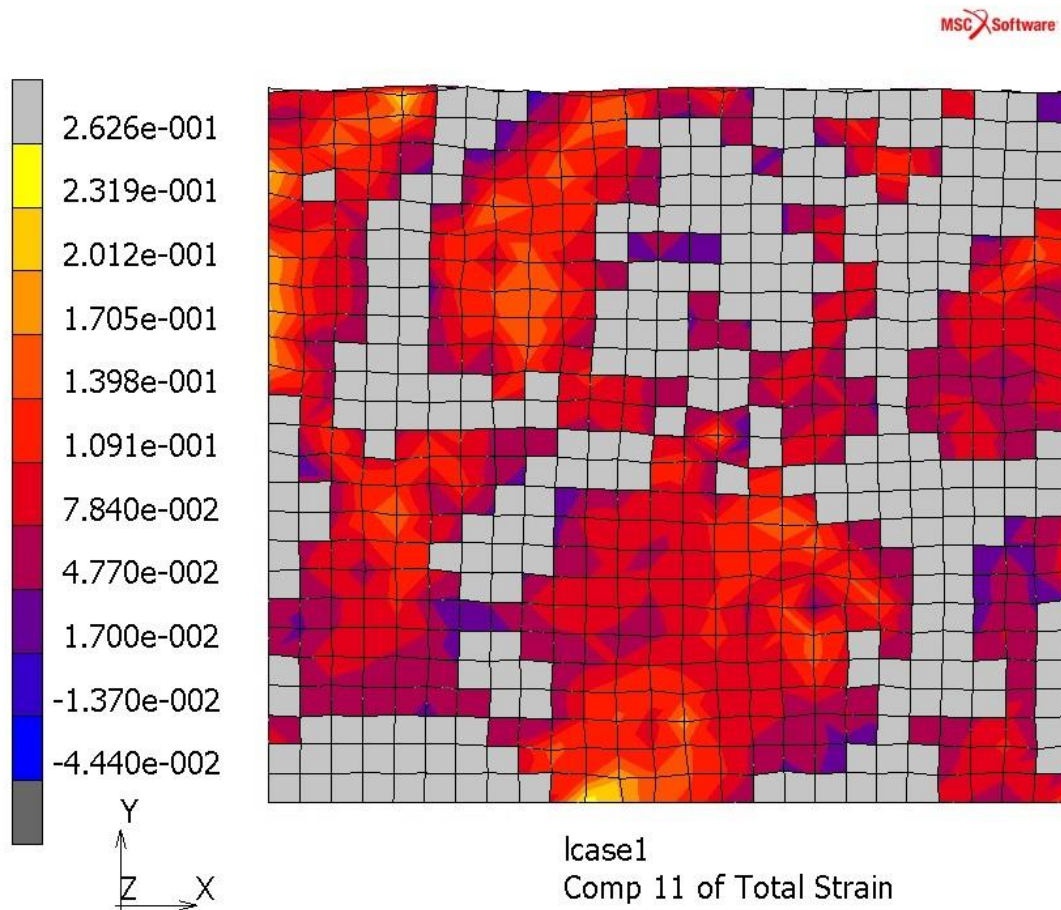


Figure 3-53 superimposed martensite elements on the strain distribution achieved from the simulation to compare with the distribution provided in [153].

Furthermore another experimental study is selected and the strain distribution of our numerical results are compared to the reported experiment results. For this purpose the research study of Base et al. [163] is selected. First the phases distribution is captured from the experimental case illustration that is shown in Figure 3-53.

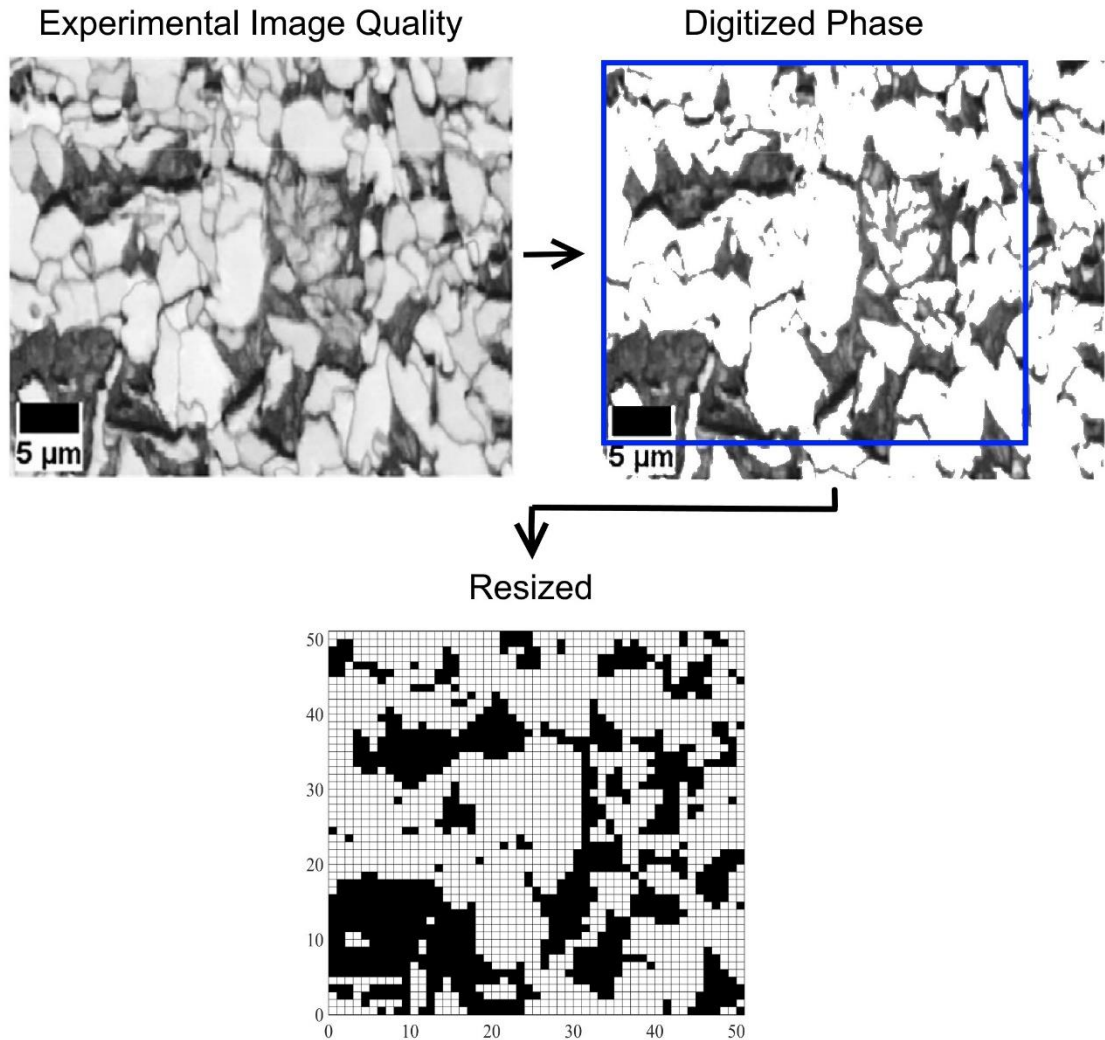
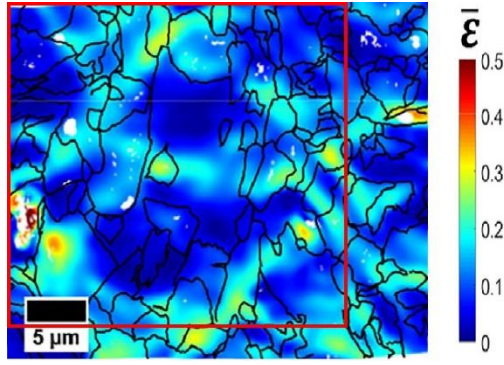
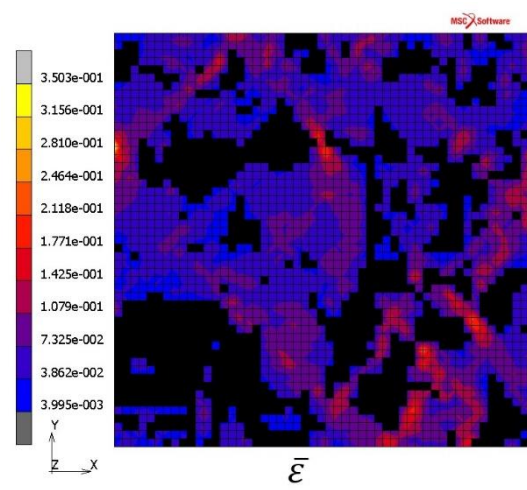


Figure 3-54 Processed phase distribution of [163] experimental investigation.

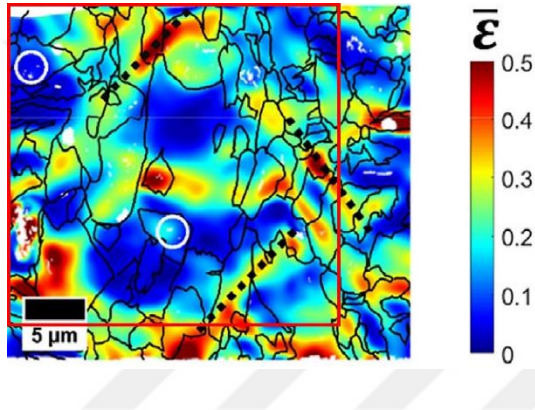
The strain distribution figures for 10% and 15% of the applied strain values are displayed in Figure 3-55. As evident, the model has accurately identified the localization points for the realistic case. The slight variation in plot intensity is attributed to the absence of information regarding the initial orientation distribution in the experimental case study.



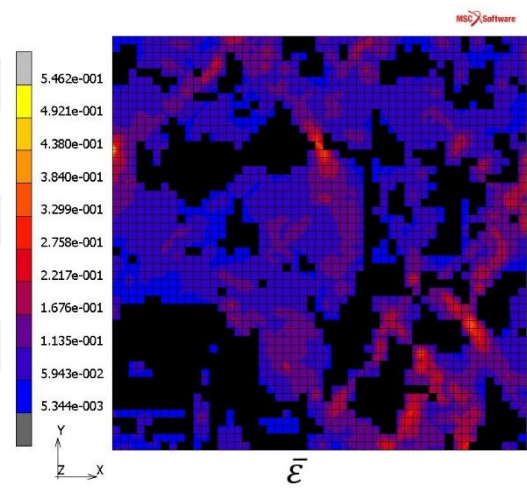
(a)



(b)



(c)



(d)

Figure 3-55 strain localization and inhomogeneity experimental validation with respect to Basu et al. [163] study, in both experimental distribution results from the literature the red box shows the area which our simulation model covers (a) experiemental reference strain distribution in 10% elongation, (b) our numerical result in undeformed state under 10% elongation, (c) experiemental reference strain distribution in 15% elongation, (d) our numerical result in undeformed state under 15% elongation.

4 Conclusion

This thesis has explored how single crystal and dual-phase steel respond to plastic deformation under different loading and effective conditions by studying their physically constitutive behaviors. The developed physics-based model using a strain hardening formulation method that applies dislocations interaction and density is intended to capture the hardening evolution in the BCC material. Initially the developed model was employed to simulate α -iron deformation behavior and to catch its experimentally observed dependence to the major effective parameters in three different geometries including MP, SC and ETG. In the calculation of mobile dislocation density, dislocation interactions, such as self-interactions, co-planar interactions, glissile interactions, Hirth lock, and Lomer-Cottrell lock mechanisms, were accounted for using a multiplicative matrix. The dislocation interaction factors were based on experimental studies from the literature [164]. The equations for the crystal plasticity simulation scheme were resolved through the utilization of the CPFEM technique. The calculations were performed using custom MATLAB scripts, along with a combination of MSC MARC a commercial finite element analysis software, and a FORTRAN subroutine. In the crystal plasticity (constitutive) model, it was considered that crystal deformation occurs exclusively through slip mechanism, and specifically by the main $\{110\} \langle 111 \rangle$ slip system evolution.

The simulation scheme was enhanced to be able to feature phase-duality behavior in the model. The developed crystal plasticity subroutine, which was limited to ferrite phase simulation, was combined with an isotropic plasticity (J2 plasticity) scheme to have the martensite phase incorporated in the simulation. This approach is based on a 3D single-layer model and is employed to investigate the evolution of microstructure and its impact on the overall mechanical response of DP600 steel under external loads. To achieve this objective, the RVE of the material was constructed using an EBSD map obtained through experimentation on the alloy. Two distinct numerical simulation methods were created to examine the impact of the martensite phase. First, a simulation model was developed to analyze the deformation of an SP ferrite material known as HYPELA-CP. This was achieved by implementing crystal plasticity and incorporating dislocation density-driven hardening formulations. In the second model, an enhancement was introduced to simultaneously simulate the behavior of both ferrite and martensite phases. This was

accomplished using a hybrid approach that combined crystal plasticity and J2 plasticity formulations, respectively. The models incorporated tensile, compression, and shear boundary conditions and were discretized into grains of three different sizes. Various RVE grain sizes were tested to confirm that the mesh choice did not significantly impact the simulations, thus illustrating the effectiveness of the approach, particularly when employing larger cell sizes.

Our developed model features the following advantages and properties:

- **Exceptional Stress/Strain Localization Capacity:** Our developed single-layer 3D model showcases a remarkable capacity for stress and strain localization. This unique feature adds a new dimension to our study, as it addresses the challenges associated with stress concentration in a novel manner.
- **Substantial Enhancement in Computation Run Time Efficiency:** The numerical processing pipeline we selected resulted in a significant improvement in computation run time efficiency. This enhancement underscores the practical applicability of our model and its potential for real-world engineering scenarios.
- **Correlation Between Stress Localization and Slip Systems Schmid Factor Orientational Stiffness:** A groundbreaking contribution of our research lies in establishing a strong correlation between stress localization and the orientational stiffness of slip systems Schmid factors in the vicinity of martensite. This observation provides an original insight into the numerical concepts explored in our research.
- **Capturing Background Physical Evolution of Dislocations:** Our modeling platform reliably captures the physical evolution of dislocations, further advancing our understanding of the material's behavior at a fundamental level.
- **Importance of Geometrical Deployment of Martensite Regions:** Through distribution plots, we highlight the significant influence of the geometrical deployment of martensite regions on stress distribution. This insight contributes to the broader understanding of stress flow mechanisms.

Specific emphasis was placed on the interaction of the martensite phase with the surrounding ferrite matrix, as numerical results revealed steep strain gradients forming

during loading. The martensite phase is expected to play an important role in the deformation response, but the mechanism(s) through which this occurs is provided by the results we present. Texture evolution and variations in local dislocation density occur, leading to changes in spatial plasticity behavior, particularly in proximity to martensite regions. The influence of these transformations in the DP material on the overall stress-strain response can be attributed to the substantial increase in dislocation densities near martensite regions, as our simulations have demonstrated. This occurs because martensite regions exhibit lower deformability, and plastic deformation initiates in adjacent ferrite grains, where the deformation intensity reaches its peak, indicating severely limited plasticity. Notably, the ferrite regions bordering the martensite phase undergo a more significant reorientation during deformation, leading to an increase in the rate of strain hardening. It's worth noting that the numerical cell size does play a role in terms of grain misorientation evolution, primarily due to the averaging of neighboring grain orientations in the simulation code. This aspect should be treated with caution when comparing with actual experimental data.

Furthermore, we demonstrate that the strain hardening response varies depending on the type of load applied, such as tension, compression, or shear. This variation is attributed in part to the cubic mechanical strength characteristics of the ferrite matrix and the differing degrees of constraint imposed by the ferrite grains adjacent to the martensite. Importantly, our numerical model effectively replicates the observed distribution of strain heterogeneity in the experiments. In summary, the model we've created can act as a robust, physics-driven tool for predicting the plastic behavior of DP steels and offering valuable understanding of the deformation mechanisms, particularly when the microstructural features are well-defined. Simulations prove to be highly efficient in terms of time, as they do not exhibit any sensitivity to cell size in the outcomes. With suitable adjustments, this approach can be expanded to encompass other alloy systems as well. The approach presented here is generic and can be applied to other alloy systems with a BCC structure by adjusting their specific physical parameters.

In summary the developed modeling methodology features the following unique properties:

- It is flexible.
- All physical parameters involved in the simulation can be tracked.
- It is founded based on the physics of the problem.

- It is sensitive to all thermos-mechanical parameters affecting the case.
- The model can be created directly.
- It is computationally efficient.
- It is a realistic modeling approach.
- The developed inhomogeneity tools are reliable measures to evaluate a system in terms of their inhomogeneity potential.
- Two or more phases and other crystalline alloys can be simulated using the developed methodology.
- It can be combined with secondary schemes to evaluate other plasticity induced responses including failure, damage and crack behavior.



5 References

- [1] C. Hamelin, B. Diak, A. Pilkey, Multiscale modelling of the induced plastic anisotropy in bcc metals, *International Journal of Plasticity* 27(8) (2011) 1185-1202.
- [2] A. Patra, D.L. McDowell, Crystal plasticity-based constitutive modelling of irradiated bcc structures, *Philosophical Magazine* 92(7) (2012) 861-887.
- [3] M. Knezevic, I.J. Beyerlein, M.L. Lovato, C.N. Tomé, A.W. Richards, R.J. McCabe, A strain-rate and temperature dependent constitutive model for BCC metals incorporating non-Schmid effects: Application to tantalum–tungsten alloys, *International Journal of Plasticity* 62 (2014) 93-104.
- [4] S. Narayanan, D.L. McDowell, T. Zhu, Crystal plasticity model for BCC iron atomistically informed by kinetics of correlated kinkpair nucleation on screw dislocation, *Journal of the Mechanics and Physics of Solids* 65 (2014) 54-68.
- [5] G. Gottstein, *Physical foundations of materials science*, Springer 2004.
- [6] A. Keh, Work hardening and deformation sub-structure in iron single crystals deformed in tension at 298 K, *Philosophical Magazine* 12(115) (1965) 9-30.
- [7] F. Guiu, P. Pratt, The effect of orientation on the yielding and flow of molybdenum single crystals, *physica status solidi (b)* 15(2) (1966) 539-552.
- [8] S. Takeuchi, K. Maeda, Slip in high purity tantalum between 0.7 and 40 K, *Acta metallurgica* 25(12) (1977) 1485-1490.
- [9] J. Nagakawa, M. Meshii, The deformation of niobium single crystals at temperatures between 77 and 4.2 K, *Philosophical Magazine A* 44(5) (1981) 1165-1191.
- [10] W. Wasserbäch, Anomalous slip in high-purity niobium and tantalum single crystals, *Physica status solidi (a)* 147(2) (1995) 417-446.
- [11] A. Seeger, W. Wasserbäch, Anomalous slip—a feature of high-purity body-centred cubic metals, *physica status solidi (a)* 189(1) (2002) 27-50.
- [12] E. Kuramoto, Y. Aono, K. Kitajima, Thermally activated slip deformation of high purity iron single crystals between 4.2 K and 300 K, *Scripta Metallurgica* 13(11) (1979) 1039-1042.
- [13] D. Caillard, Kinetics of dislocations in pure Fe. Part I. In situ straining experiments at room temperature, *Acta Materialia* 58(9) (2010) 3493-3503.
- [14] M. Tang, L. Kubin, G. Canova, Dislocation mobility and the mechanical response of bcc single crystals: a mesoscopic approach, *Acta Materialia* 46(9) (1998) 3221-3235.

- [15] Y. Aono, E. Kuramoto, K. Kitajima, Fundamental plastic behaviors in high-purity BCC metals (Nb, Mo and Fe), *Strength of Metals and Alloys (ICSMA 6)*, Elsevier 1982, pp. 9-14.
- [16] A. Argon, *Strengthening mechanisms in crystal plasticity*, OUP Oxford 2007.
- [17] A. Ma, F. Roters, D. Raabe, A dislocation density based constitutive law for BCC materials in crystal plasticity FEM, *Computational Materials Science* 39(1) (2007) 91-95.
- [18] C.R. Weinberger, C.C. Battaile, T.E. Buchheit, E.A. Holm, Incorporating atomistic data of lattice friction into BCC crystal plasticity models, *International Journal of Plasticity* 37 (2012) 16-30.
- [19] V. Vitek, M. Mrovec, R. Gröger, J. Bassani, V. Racherla, L. Yin, Effects of non-glide stresses on the plastic flow of single and polycrystals of molybdenum, *Materials Science and Engineering: A* 387 (2004) 138-142.
- [20] E. Schmid, W. Boas, *Plasticity of Crystals* (translated from the 1935 edition of *Kristallplastizität*): FA Hughes and Co, London, UK (1950).
- [21] M. Duesbery, On non-glide stresses and their influence on the screw dislocation core in body-centred cubic metals I. The Peierls stress, *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 392(1802) (1984) 145-173.
- [22] J. Christian, Some surprising features of the plastic deformation of body-centered cubic metals and alloys, *Metallurgical transactions A* 14(7) (1983) 1237-1256.
- [23] A.S. Khan, R. Liang, Behaviors of three BCC metal over a wide range of strain rates and temperatures: experiments and modeling, *International Journal of Plasticity* 15(10) (1999) 1089-1109.
- [24] G.Z. Voyiadjis, F.H. Abed, A coupled temperature and strain rate dependent yield function for dynamic deformations of bcc metals, *International Journal of Plasticity* 22(8) (2006) 1398-1431.
- [25] M. Duesbery, R. Foxall, A detailed study of the deformation of high purity niobium single crystals, *Philosophical Magazine* 20(166) (1969) 719-751.
- [26] C. Bolton, G. Taylor, Anomalous slip in high-purity niobium single crystals deformed at 77 K in tension, *Philosophical Magazine* 26(6) (1972) 1359-1376.
- [27] P. Jeffcoat, B. Mordike, K. Rogausch, Anomalous slip in Mo-5 at.% Nb and Mo-5 at.% Re alloy single crystals, *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics* 34(4) (1976) 583-592.
- [28] H. Matsui, H. Kimura, Anomalous {110} slip in high-purity molybdenum single crystals and its comparison with that in V (a) metals, *Materials Science and Engineering* 24(2) (1976) 247-256.

- [29] V. Vitek, V. Paidar, Non-planar dislocation cores: a ubiquitous phenomenon affecting mechanical properties of crystalline materials, *Dislocations in solids* 14 (2008) 439-514.
- [30] L. Kubin, The low temperature mechanical properties of BCC metals and their alloys, *Reviews on the deformation behavior of materials* 4(3) (1982) 181-275.
- [31] M.a.-S. Duesbery, V. Vitek, Plastic anisotropy in bcc transition metals, *Acta Materialia* 46(5) (1998) 1481-1492.
- [32] V. Vitek, Structure of dislocation cores in metallic materials and its impact on their plastic behaviour, *Progress in materials science* 36 (1992) 1-27.
- [33] K. Ito, V. Vitek, Atomistic study of non-Schmid effects in the plastic yielding of bcc metals, *Philosophical Magazine A* 81(5) (2001) 1387-1407.
- [34] Z. Wang, I. Beyerlein, An atomistically-informed dislocation dynamics model for the plastic anisotropy and tension–compression asymmetry of BCC metals, *International Journal of Plasticity* 27(10) (2011) 1471-1484.
- [35] R. Gröger, V. Vitek, Impact of non-Schmid stress components present in the yield criterion for bcc metals on the activity of $\{110\}<111>$ slip systems, *Computational Materials Science* 159 (2019) 297-305.
- [36] H. Cho, C.A. Bronkhorst, H.M. Mourad, J.R. Mayeur, D. Luscher, Anomalous plasticity of body-centered-cubic crystals with non-Schmid effect, *International Journal of Solids and Structures* 139 (2018) 138-149.
- [37] S. Mahmoody, Micromechanical modeling of dual-phase steel using a rate-dependent crystal plasticity model, (2007).
- [38] N. Saeidi, F. Ashrafizadeh, B. Niroumand, Development of a new ultrafine grained dual phase steel and examination of the effect of grain size on tensile deformation behavior, *Materials Science and Engineering: A* 599 (2014) 145-149.
- [39] M.D. Tumuluru, Resistance spot welding of coated high-strength dual-phase steels, *Welding Journal* 85(8) (2006) 31-37.
- [40] A.N. Vasilakos, J. Ohlert, K. Giasla, G.N. Haidemenopoulos, W. Bleck, Low-alloy TRIP steels: a correlation between mechanical properties and the retained austenite stability, *steel research* 73(6-7) (2002) 249-252.
- [41] P. Wanjara, M. Hone, N. Nickoletopoulos, A. Tendler, S. Yue, J. Nemes, Dual-phase steels for cold-heading applications, *Wire Journal International* 34(9) (2001) 104-107.
- [42] C.C. Tasan, M. Diehl, D. Yan, M. Bechtold, F. Roters, L. Schemmann, C. Zheng, N. Peranio, D. Ponge, M. Koyama, An overview of dual-phase steels: advances in microstructure-oriented processing and micromechanically guided design, *Annual Review of Materials Research* 45 (2015) 391-431.

- [43] H. Ghassemi-Armaki, R. Maaß, S. Bhat, S. Sriram, J. Greer, K. Kumar, Deformation response of ferrite and martensite in a dual-phase steel, *Acta Materialia* 62 (2014) 197-211.
- [44] P. Chen, H. Ghassemi-Armaki, S. Kumar, A. Bower, S. Bhat, S. Sadagopan, Microscale-calibrated modeling of the deformation response of dual-phase steels, *Acta Materialia* 65 (2014) 133-149.
- [45] C. Tian, D. Ponge, L. Christiansen, C. Kirchlechner, On the mechanical heterogeneity in dual phase steel grades: Activation of slip systems and deformation of martensite in DP800, *Acta Materialia* 183 (2020) 274-284.
- [46] N. Habibi, N. Vajragupta, S. Münstermann, Deformation and Damage Assessments of Two DP1000 Steels Using a Micromechanical Modelling Method, *Crystals* 11(7) (2021) 805.
- [47] I.A. Riyad, W.G. Feather, E. Vasilev, R.A. Lebensohn, B.A. McWilliams, A.L. Pilchak, M. Knezevic, Modeling the role of local crystallographic correlations in microstructures of Ti-6Al-4V using a correlated structure visco-plastic self-consistent polycrystal plasticity formulation, *Acta Materialia* 203 (2021) 116502.
- [48] I.A. Riyad, B.A. McWilliams, B. Pramanik, M. Knezevic, Correlated structure viscoplastic self-consistent polycrystal plasticity: Application to modeling strain rate sensitive deformation of Ti-6Al-4 V, *International Journal of Plasticity* 163 (2023) 103571.
- [49] M. Zecevic, M. Knezevic, Latent hardening within the elasto-plastic self-consistent polycrystal homogenization to enable the prediction of anisotropy of AA6022-T4 sheets, *International Journal of Plasticity* 105 (2018) 141-163.
- [50] P. Gudmundson, C.F. Dahlberg, Isotropic strain gradient plasticity model based on self-energies of dislocations and the Taylor model for plastic dissipation, *International Journal of Plasticity* 121 (2019) 1-20.
- [51] M. Knezevic, S.R. Kalidindi, Crystal plasticity modeling of microstructure evolution and mechanical fields during processing of metals using spectral databases, *JOM* 69(5) (2017) 830-838.
- [52] A. Eghtesad, M. Knezevic, A full-field crystal plasticity model including the effects of precipitates: Application to monotonic, load reversal, and low-cycle fatigue behavior of Inconel 718, *Materials Science and Engineering: A* 803 (2021) 140478.
- [53] A. Eghtesad, M. Knezevic, Modeling cyclic plasticity of additively manufactured alloy Mar-M-509 using a high-performance spectral-based micromechanical model, *Applications in Engineering Science* 7 (2021) 100065.
- [54] Y.L. Li, C.P. Kohar, W. Muhammad, K. Inal, Precipitation kinetics and crystal plasticity modeling of artificially aged AA6061, *International Journal of Plasticity* 152 (2022) 103241.

- [55] M. Ardeljan, M. Knezevic, Explicit modeling of double twinning in AZ31 using crystal plasticity finite elements for predicting the mechanical fields for twin variant selection and fracture analyses, *Acta Materialia* 157 (2018) 339-354.
- [56] T.J. Barrett, D.J. Savage, M. Ardeljan, M. Knezevic, An automated procedure for geometry creation and finite element mesh generation: Application to explicit grain structure models and machining distortion, *Computational materials science* 141 (2018) 269-281.
- [57] A. Arsenlis, D.M. Parks, R. Becker, V.V. Bulatov, On the evolution of crystallographic dislocation density in non-homogeneously deforming crystals, *Journal of the Mechanics and Physics of Solids* 52(6) (2004) 1213-1246.
- [58] O.C. Zienkiewicz, R.L. Taylor, *The finite element method for solid and structural mechanics*, Elsevier 2005.
- [59] F. Roters, P. Eisenlohr, L. Hantcherli, D.D. Tjahjanto, T.R. Bieler, D. Raabe, Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: Theory, experiments, applications, *Acta materialia* 58(4) (2010) 1152-1211.
- [60] M. Sachtleber, Z. Zhao, D. Raabe, Experimental investigation of plastic grain interaction, *Materials Science and Engineering: A* 336(1-2) (2002) 81-87.
- [61] S. Daroju, T. Kuwabara, M. Knezevic, Experimental characterization and crystal plasticity modeling of dual-phase steels subjected to strain path reversals, *Mechanics of Materials* 168 (2022) 104293.
- [62] E. Bittencourt, Interpretation of the size effects in micropillar compression by a strain gradient crystal plasticity theory, *International Journal of Plasticity* 116 (2019) 280-296.
- [63] A. Arsenlis, M. Tang, Simulations on the growth of dislocation density during Stage 0 deformation in BCC metals, *Modelling and Simulation in Materials Science and Engineering* 11(2) (2003) 251.
- [64] A. Patra, D.L. McDowell, Continuum modeling of localized deformation in irradiated bcc materials, *Journal of Nuclear Materials* 432(1-3) (2013) 414-427.
- [65] H. Li, X. Sun, H. Yang, A three-dimensional cellular automata-crystal plasticity finite element model for predicting the multiscale interaction among heterogeneous deformation, DRX microstructural evolution and mechanical responses in titanium alloys, *International Journal of Plasticity* 87 (2016) 154-180.
- [66] L. Stainier, A.M. Cuitiño, M. Ortiz, A micromechanical model of hardening, rate sensitivity and thermal softening in bcc single crystals, *Journal of the Mechanics and Physics of Solids* 50(7) (2002) 1511-1545.
- [67] Y.J. Lee, G. Subhash, G. Ravichandran, Constitutive modeling of textured body-centered-cubic (bcc) polycrystals, *International Journal of Plasticity* 15(6) (1999) 625-645.

- [68] M. Lee, H. Lim, B. Adams, J. Hirth, R. Wagoner, A dislocation density-based single crystal constitutive equation, *International Journal of Plasticity* 26(7) (2010) 925-938.
- [69] R.A. Austin, D.L. McDowell, A dislocation-based constitutive model for viscoplastic deformation of fcc metals at very high strain rates, *International Journal of Plasticity* 27(1) (2011) 1-24.
- [70] M. Ardeljan, I.J. Beyerlein, M. Knezevic, A dislocation density based crystal plasticity finite element model: Application to a two-phase polycrystalline HCP/BCC composites, *Journal of the Mechanics and Physics of Solids* 66 (2014) 16-31.
- [71] A. Ma, F. Roters, A constitutive model for fcc single crystals based on dislocation densities and its application to uniaxial compression of aluminium single crystals, *Acta materialia* 52(12) (2004) 3603-3612.
- [72] E. Demir, A physically based constitutive model for FCC single crystals with a single state variable per slip system, *Modelling and Simulation in Materials Science and Engineering* 25(1) (2016) 015009.
- [73] E. Demir, I. Gutierrez-Urrutia, Investigation of strain hardening near grain boundaries of an aluminum oligocrystal: experiments and crystal based finite element method, *International Journal of Plasticity* 136 (2021) 102898.
- [74] Q. Qin, J.L. Bassani, Non-Schmid yield behavior in single crystals, *Journal of the Mechanics and Physics of Solids* 40(4) (1992) 813-833.
- [75] Q. Qin, J.L. Bassani, Non-associated plastic flow in single crystals, *Journal of the Mechanics and Physics of Solids* 40(4) (1992) 835-862.
- [76] P. Steinmann, E. Kuhl, E. Stein, Aspects of non-associated single crystal plasticity: influence of non-Schmid effects and localization analysis, *International journal of solids and structures* 35(33) (1998) 4437-4456.
- [77] S. Nemat-Nasser, T. Okinaka, L. Ni, A physically-based constitutive model for bcc crystals with application to polycrystalline tantalum, *Journal of the Mechanics and Physics of Solids* 46(6) (1998) 1009-1038.
- [78] T. Yalcinkaya, W. Brekelmans, M. Geers, BCC single crystal plasticity modeling and its experimental identification, *Modelling and Simulation in Materials Science and Engineering* 16(8) (2008) 085007.
- [79] A. Patra, T. Zhu, D.L. McDowell, Constitutive equations for modeling non-Schmid effects in single crystal bcc-Fe at low and ambient temperatures, *International Journal of Plasticity* 59 (2014) 1-14.
- [80] D.J. Savage, I.J. Beyerlein, M. Knezevic, Coupled texture and non-Schmid effects on yield surfaces of body-centered cubic polycrystals predicted by a crystal plasticity finite element approach, *International Journal of Solids and Structures* 109 (2017) 22-32.
- [81] A.M. Cantara, M. Zecevic, A. Eghtesad, C.M. Poulin, M. Knezevic, Predicting elastic anisotropy of dual-phase steels based on crystal mechanics and microstructure, *International Journal of Mechanical Sciences* 151 (2019) 639-649.

- [82] K. Sedighiani, K. Traka, F. Roters, J. Sietsma, D. Raabe, M. Diehl, Crystal plasticity simulation of in-grain microstructural evolution during large deformation of IF-steel, *Acta Materialia* 237 (2022) 118167.
- [83] Z. Cai, B. Meng, M. Wan, X. Wu, M. Fu, A modified yield function for modeling of the evolving yielding behavior and micro-mechanism in biaxial deformation of sheet metals, *International Journal of Plasticity* 129 (2020) 102707.
- [84] T. Yalçinkaya, S.O. Çakmak, C. Tekoğlu, A crystal plasticity based finite element framework for RVE calculations of two-phase materials: Void nucleation in dual-phase steels, *Finite Elements in Analysis and Design* 187 (2021) 103510.
- [85] T. Yalçinkaya, B. Tatli, I.E. Ünsal, I.U. Aydiner, Crack Initiation and Propagation in Dual-phase Steels Through Crystal Plasticity and Cohesive Zone Frameworks, *Procedia Structural Integrity* 42 (2022) 1651-1659.
- [86] A. Chatterjee, M.B. Sk, A. Ghosh, R. Mitra, D. Chakrabarti, Combining crystal plasticity and electron microscopy to elucidate texture dependent micro-mechanisms of tensile deformation in lath martensitic steel, *International Journal of Plasticity* 153 (2022) 103251.
- [87] F. Qayyum, A.A. Chaudhry, S. Guk, M. Schmidtchen, R. Kawalla, U. Prahl, Effect of 3D representative volume element (RVE) thickness on stress and strain partitioning in crystal plasticity simulations of multi-phase materials, *Crystals* 10(10) (2020) 944.
- [88] A. Govik, R. Rentmeester, L. Nilsson, A study of the unloading behaviour of dual phase steel, *Materials Science and Engineering: A* 602 (2014) 119-126.
- [89] S.K. Paul, A. Kumar, Micromechanics based modeling to predict flow behavior and plastic strain localization of dual phase steels, *Computational materials science* 63 (2012) 66-74.
- [90] A. Ramazani, K. Mukherjee, H. Quade, U. Prahl, W. Bleck, Correlation between 2D and 3D flow curve modelling of DP steels using a microstructure-based RVE approach, *Materials Science and Engineering: A* 560 (2013) 129-139.
- [91] X. Sun, K.S. Choi, W.N. Liu, M.A. Khaleel, Predicting failure modes and ductility of dual phase steels using plastic strain localization, *International Journal of Plasticity* 25(10) (2009) 1888-1909.
- [92] X. Sun, K.S. Choi, A. Soulami, W.N. Liu, M.A. Khaleel, On key factors influencing ductile fractures of dual phase (DP) steels, *Materials Science and Engineering: A* 526(1-2) (2009) 140-149.
- [93] J. Kadkhodapour, A. Butz, S. Ziaei-Rad, S. Schmauder, A micro mechanical study on failure initiation of dual phase steels under tension using single crystal plasticity model, *International Journal of Plasticity* 27(7) (2011) 1103-1125.
- [94] J.H. Kim, D. Kim, F. Barlat, M.-G. Lee, Crystal plasticity approach for predicting the Bauschinger effect in dual-phase steels, *Materials Science and Engineering: A* 539 (2012) 259-270.

- [95] C.C. Tasan, J.P. Hoefnagels, M. Diehl, D. Yan, F. Roters, D. Raabe, Strain localization and damage in dual phase steels investigated by coupled in-situ deformation experiments and crystal plasticity simulations, *International Journal of Plasticity* 63 (2014) 198-210.
- [96] W. Woo, V. Em, E.-Y. Kim, S. Han, Y. Han, S.-H. Choi, Stress-strain relationship between ferrite and martensite in a dual-phase steel studied by in situ neutron diffraction and crystal plasticity theories, *Acta materialia* 60(20) (2012) 6972-6981.
- [97] A. Biswas, S.R. Kalidindi, A. Hartmaier, A hybrid approach for the efficient computation of polycrystalline yield loci with the accuracy of the crystal plasticity finite element method, *Modelling and Simulation in Materials Science and Engineering* 30(2) (2022) 025015.
- [98] C. Bonatti, B. Berisha, D. Mohr, From CP-FFT to CP-RNN: Recurrent neural network surrogate model of crystal plasticity, *International Journal of Plasticity* 158 (2022) 103430.
- [99] A. Castillo, S.R. Kalidindi, A Bayesian framework for the estimation of the single crystal elastic parameters from spherical indentation stress-strain measurements, *Frontiers in Materials* 6 (2019) 136.
- [100] A. Venkatraman, D.L. McDowell, S.R. Kalidindi, Bayesian analysis of parametric uncertainties and model form probabilities for two different crystal plasticity models of lamellar grains in $\alpha + \beta$ Titanium alloys, *International Journal of Plasticity* 154 (2022) 103289.
- [101] Y. Hollenweger, D.M. Kochmann, An efficient temperature-dependent crystal plasticity framework for pure magnesium with emphasis on the competition between slip and twinning, *International Journal of Plasticity* 159 (2022) 103448.
- [102] S.K. Sahoo, S.S. Dhinwal, V.Q. Vu, L.S. Toth, A new macroscopic strain hardening function based on microscale crystal plasticity and its application in polycrystal modeling, *Materials Science and Engineering: A* 823 (2021) 141634.
- [103] Y. Zhang, H. Chen, Y.-F. Jia, D.-F. Li, G.-J. Yuan, X.-C. Zhang, S.-T. Tu, A modified kinematic hardening model considering hetero-deformation induced hardening for bimodal structure based on crystal plasticity, *International Journal of Mechanical Sciences* 191 (2021) 106068.
- [104] J. Kadkhodapour, A. Butz, S.Z. Rad, Mechanisms of void formation during tensile testing in a commercial, dual-phase steel, *Acta Materialia* 59(7) (2011) 2575-2588.
- [105] D.-K. Kim, E.-Y. Kim, J. Han, W. Woo, S.-H. Choi, Effect of microstructural factors on void formation by ferrite/martensite interface decohesion in DP980 steel under uniaxial tension, *International Journal of Plasticity* 94 (2017) 3-23.
- [106] A. Alaie, S.Z. Rad, J. Kadkhodapour, M. Jafari, M.A. Asadabad, S. Schmauder, Effect of microstructure pattern on the strain localization in DP600 steels analyzed using combined in-situ experimental test and numerical simulation, *Materials Science and Engineering: A* 638 (2015) 251-261.

- [107] Q. Li, H. Zhang, F. Chen, D. Xu, D. Sui, Z. Cui, Study on the plastic anisotropy of advanced high strength steel sheet: Experiments and microstructure-based crystal plasticity modeling, *International Journal of Mechanical Sciences* 176 (2020) 105569.
- [108] M. Zecevic, Y.P. Korkolis, T. Kuwabara, M. Knezevic, Dual-phase steel sheets under cyclic tension–compression to large strains: experiments and crystal plasticity modeling, *Journal of the Mechanics and Physics of Solids* 96 (2016) 65-87.
- [109] T. Hussein, M. Umar, F. Qayyum, S. Guk, U. Prah, Micromechanical Effect of Martensite Attributes on Forming Limits of Dual-Phase Steels Investigated by Crystal Plasticity-Based Numerical Simulations, *Crystals* 12(2) (2022) 155.
- [110] M. Jafari, N. Saeidi, S. Ziaei-Rad, M. Jamshidian, H. Kim, Micromechanical analysis of orientation dependency on deformation behavior in DP steels by dislocation density-based crystal plasticity simulation, *Mechanics of Materials* 134 (2019) 132-142.
- [111] C. Jia, G. Shen, W. Chen, B. Hu, C. Zheng, D. Li, Mesoscopic analysis of deformation heterogeneity and recrystallization microstructures of a dual-phase steel using a coupled simulation approach, *Acta Metallurgica Sinica (English Letters)* 34 (2021) 777-788.
- [112] H. Ghadbeigi, C. Pinna, S. Celotto, Failure mechanisms in DP600 steel: Initiation, evolution and fracture, *Materials Science and Engineering: A* 588 (2013) 420-431.
- [113] Q. Han, Y. Kang, P.D. Hodgson, N. Stanford, Quantitative measurement of strain partitioning and slip systems in a dual-phase steel, *Scripta Materialia* 69(1) (2013) 13-16.
- [114] S.-H. Joo, J.K. Lee, J.-M. Koo, S. Lee, D.-W. Suh, H.S. Kim, Method for measuring nanoscale local strain in a dual phase steel using digital image correlation with nanodot patterns, *Scripta Materialia* 68(5) (2013) 245-248.
- [115] C. Tasan, J. Hoefnagels, M. Geers, Microstructural banding effects clarified through micrographic digital image correlation, *Scripta Materialia* 62(11) (2010) 835-838.
- [116] M. Kapp, T. Hebesberger, O. Kolednik, A micro-level strain analysis of a high-strength dual-phase steel, *International journal of materials research* 102(6) (2011) 687-691.
- [117] J. Marteau, H. Haddadi, S. Bouvier, Investigation of strain heterogeneities between grains in ferritic and ferritic-martensitic steels, *Experimental Mechanics* 53(3) (2013) 427-439.
- [118] J. Galán-López, J. Hidalgo, Use of the Correlation between Grain Size and Crystallographic Orientation in Crystal Plasticity Simulations: Application to AISI 420 Stainless Steel, *Crystals* 10(9) (2020) 819.
- [119] S. Sinha, J.A. Szpunar, N.K. Kumar, N. Gurao, Tensile deformation of 316L austenitic stainless steel using in-situ electron backscatter diffraction and crystal plasticity simulations, *Materials Science and Engineering: A* 637 (2015) 48-55.

- [120] C. Zhang, L.-W. Zhang, W.-F. Shen, Y.-N. Xia, Y.-T. Yan, 3d crystal plasticity finite element modeling of the tensile deformation of polycrystalline ferritic stainless steel, *Acta Metallurgica Sinica (English Letters)* 30(1) (2017) 79-88.
- [121] A. Sardar, Predicting the Flow and Failure Properties of Dual Phase Steel Using Finite Element and Phenomenological Models, University of Guelph, 2019.
- [122] T. Mori, K. Tanaka, Average stress in matrix and average elastic energy of materials with misfitting inclusions, *Acta metallurgica* 21(5) (1973) 571-574.
- [123] J. Ha, J. Lee, J.H. Kim, M.-G. Lee, F. Barlat, Investigation of plastic strain rate under strain path changes in dual-phase steel using microstructure-based modeling, *International Journal of Plasticity* 93 (2017) 89-111.
- [124] L. Mishnaevsky Jr, N. Lippmann, S. Schmauder, Computational design of multiphase materials at the mesolevel, ASME International Mechanical Engineering Congress and Exposition, November, 2001, pp. 11-16.
- [125] L. Mishnaevsky, N. Lippmann, S. Schmauder, Computational modeling of crack propagation in real microstructures of steels and virtual testing of artificially designed materials, *International journal of fracture* 120 (2003) 581-600.
- [126] D. Tjahjanto, P. Eisenlohr, F. Roters, A novel grain cluster-based homogenization scheme, *Modelling and Simulation in Materials Science and Engineering* 18(1) (2009) 015006.
- [127] F. Roters, P. Eisenlohr, C. Kords, D.D. Tjahjanto, M. Diehl, D. Raabe, DAMASK: the Düsseldorf Advanced MATERIAL Simulation Kit for studying crystal plasticity using an FE based or a spectral numerical solver, *Procedia Iutam* 3 (2012) 3-10.
- [128] J. Schröder, D. Balzani, D. Brands, Approximation of random microstructures by periodic statistically similar representative volume elements based on lineal-path functions, *Archive of Applied Mechanics* 81 (2011) 975-997.
- [129] V. Uthaisangsuk, U. Prah, W. Bleck, Stretch-flangeability characterisation of multiphase steel using a microstructure based failure modelling, *Computational Materials Science* 45(3) (2009) 617-623.
- [130] Y. Prawoto, M. Fanone, S. Shahedi, M. Ismail, W.W. Nik, Computational approach using Johnson–Cook model on dual phase steel, *Computational Materials Science* 54 (2012) 48-55.
- [131] E.J. Seo, L. Cho, Y. Estrin, B.C. De Cooman, Microstructure-mechanical properties relationships for quenching and partitioning (Q&P) processed steel, *Acta Materialia* 113 (2016) 124-139.
- [132] E.H. Lee, Elastic-plastic deformation at finite strains, (1969).
- [133] G. Po, Y. Cui, D. Rivera, D. Cereceda, T.D. Swinburne, J. Marian, N. Ghoniem, A phenomenological dislocation mobility law for bcc metals, *Acta Materialia* 119 (2016) 123-135.

- [134] A. Maniatty, P. Dawson, Y.S. Lee, A time integration algorithm for elasto-viscoplastic cubic crystals applied to modelling polycrystalline deformation, *International Journal for Numerical Methods in Engineering* 35(8) (1992) 1565-1588.
- [135] C.A. Bronkhorst, S. Kalidindi, L. Anand, Polycrystalline plasticity and the evolution of crystallographic texture in FCC metals, *Philosophical Transactions of the Royal Society of London. Series A: Physical and Engineering Sciences* 341(1662) (1992) 443-477.
- [136] E. Demir, F. Roters, D. Raabe, Bending of single crystal microcantilever beams of cube orientation: Finite element model and experiments, *Journal of the Mechanics and Physics of Solids* 58(10) (2010) 1599-1612.
- [137] F. Roters, D. Raabe, G. Gottstein, Work hardening in heterogeneous alloys—a microstructural approach based on three internal state variables, *Acta materialia* 48(17) (2000) 4181-4189.
- [138] U. Kocks, H. Mecking, Physics and phenomenology of strain hardening: the FCC case, *Progress in materials science* 48(3) (2003) 171-273.
- [139] H. Frost, M. Ashby, *Deformation-mechanism maps: The plasticity and creep of metals and ceramics*(Book), Oxford, Pergamon Press, 1982, 175 p (1982).
- [140] M. Geers, M. Cottura, B. Appolaire, E.P. Busso, S. Forest, A. Villani, Coupled glide-climb diffusion-enhanced crystal plasticity, *Journal of the Mechanics and Physics of Solids* 70 (2014) 136-153.
- [141] E. Nes, Modelling of work hardening and stress saturation in FCC metals, *Progress in materials science* 41(3) (1997) 129-193.
- [142] P. Franciosi, Glide mechanisms in bcc crystals: an investigation of the case of α -iron through multislip and latent hardening tests, *Acta Metallurgica* 31(9) (1983) 1331-1342.
- [143] F. Maresca, V. Kouznetsova, M. Geers, Reduced crystal plasticity for materials with constrained slip activity, *Mechanics of Materials* 92 (2016) 198-210.
- [144] J.J. Adams, D. Agosta, R. Leisure, H. Ledbetter, Elastic constants of monocrystal iron from 3 to 500 K, *Journal of applied physics* 100(11) (2006) 113530.
- [145] R. Johnson, D. Oh, Analytic embedded atom method model for bcc metals, *Journal of Materials Research* 4(5) (1989) 1195-1201.
- [146] A. Das, Revisiting stacking fault energy of steels, *Metallurgical and materials transactions A* 47 (2016) 748-768.
- [147] S. Turteltaub, A. Suiker, Transformation-induced plasticity in ferrous alloys, *Journal of the Mechanics and Physics of Solids* 53(8) (2005) 1747-1788.
- [148] Q. Furnémont, M. Kempf, P. Jacques, M. Göken, F. Delannay, On the measurement of the nanohardness of the constitutive phases of TRIP-assisted multiphase steels, *Materials Science and Engineering: A* 328(1-2) (2002) 26-32.

- [149] S.A.H. Motaman, F. Roters, C. Haase, Anisotropic polycrystal plasticity due to microstructural heterogeneity: a multi-scale experimental and numerical study on additively manufactured metallic materials, *Acta Materialia* 185 (2020) 340-369.
- [150] S.A.H. Motaman, C. Haase, The microstructural effects on the mechanical response of polycrystals: A comparative experimental-numerical study on conventionally and additively manufactured metallic materials, *International Journal of Plasticity* 140 (2021) 102941.
- [151] M.H. Joudivand Sarand, E. Demir, A physically based model for bcc materials including non-Schmid effects and its application to single crystals of α -iron at different model scales, *Modelling and Simulation in Materials Science and Engineering* 29(5) (2021) 055016.
- [152] S.O. Cakmak, T. Yalçinkaya, Morphology and grain orientation dependent localization and necking in dual-phase steels, *Procedia Structural Integrity* 21 (2019) 224-232.
- [153] C. Pelligra, J. Samei, J. Kang, D.S. Wilkinson, The effect of vanadium on microstrain partitioning and localized damage during deformation of unnotched and notched DP1300 steels, *International Journal of Plasticity* 158 (2022) 103435.
- [154] A. Ghaei, D.E. Green, S. Thuillier, F. Morestin, On the use of cyclic shear, bending and uniaxial tension–compression tests to reproduce the cyclic response of sheet metals, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 229(3) (2015) 453-462.
- [155] W. Spitzig, A. Keh, Orientation and temperature dependence of slip in iron single crystals, *Metallurgical Transactions* 1 (1970) 2751-2757.
- [156] S. Zwiese, J. Diehl, Temperature and strain rate dependence of the macro yield stress of high purity iron single crystals, *Strength of Metals and Alloys*, Elsevier 1979, pp. 59-64.
- [157] H. Lim, L. Hale, J. Zimmerman, C. Battaile, C. Weinberger, A multi-scale model of dislocation plasticity in α -Fe: Incorporating temperature, strain rate and non-Schmid effects, *International Journal of Plasticity* 73 (2015) 100-118.
- [158] W. Liu, J. Lian, N. Aravas, S. Münstermann, A strategy for synthetic microstructure generation and crystal plasticity parameter calibration of fine-grain-structured dual-phase steel, *International Journal of Plasticity* 126 (2020) 102614.
- [159] K. Park, M. Nishiyama, N. Nakada, T. Tsuchiyama, S. Takaki, Effect of the martensite distribution on the strain hardening and ductile fracture behaviors in dual-phase steel, *Materials Science and Engineering: A* 604 (2014) 135-141.
- [160] F. Briffod, H. Hu, T. Shiraiwa, M. Enoki, Effect of in-lath slip strength on the strain partitioning in a dual-phase steel investigated by high-resolution digital image correlation and crystal plasticity simulations, *Materials Science and Engineering: A* 862 (2023) 144413.

- [161] Y. Tomota, I. TAMURA, Mechanical behavior of steels consisting of two ductile phases, Transactions of the Iron and Steel Institute of Japan 22(9) (1982) 665-677.
- [162] G. Avramovic-Cingara, Y. Ososkov, M. Jain, D. Wilkinson, Effect of martensite distribution on damage behaviour in DP600 dual phase steels, Materials Science and Engineering: A 516(1-2) (2009) 7-16.
- [163] S. Basu, A. Patra, B.N. Jaya, S. Ganguly, M. Dutta, I. Samajdar, Study of microstructure-property correlations in dual phase steels for achieving enhanced strength and reduced strain partitioning, Materialia 25 (2022) 101522.
- [164] K. Sedighiani, K. Traka, F. Roters, D. Raabe, J. Sietsma, M. Diehl, Determination and analysis of the constitutive parameters of temperature-dependent dislocation-density-based crystal plasticity models, Mechanics of Materials 164 (2022) 104117.



6 Appendices

6.1 HYPELA2-CP Fortran subroutine

```
c Mohammad Hasan Joudivand
c Jan. 09, 2020
c
c   This subroutine is written for bcc materials
c   The material properties and element properties are the SAME over the mesh
for every elements
c   Include the global variables and subroutines used
include "globalvars.f"
include "globalsubs.f"
c   Include the subroutine that contains the calculations
include "calculations.f"
SUBROUTINE HYPELA2(D,G,E,DE,S,T,DT,NGENS,N,NN,KCUS,MATUS,
2  NDI,NSHEAR,DISP,DISPT,COORD,FFN,FROTN,STRECHN,EIGVN,FFN1,
3  FROTN1,STRECHN1,EIGVN1,NCRD,ITEL,NDEG,NDM,NNODE,
4  JTYPE,LCLASS,IFR,IFU)
  use globalsubs, only : initialize_all
  use calculations, only : calculate
  use globalvars, only : El_MisTr
  IMPLICIT REAL *8 (A-H, O-Z)
c   real(8) dummyS(6),dummyD(6,6)
c   INCLUDE LIBRARIES and FORTRAN FILES necessary:
c   Include the library 'creeps' to get the time increment (TIMINC)
include "creeps"
c   Include the library 'concom' to get the increment number (INC)
include "concom"
  DIMENSION E(*),DE(*),T(*),DT(*),G(*),D(NGENS,NGENS),S(NGENS)
  DIMENSION N(2),COORD(NCRD,NNODE),DISP(NDEG,NNODE),
2  DISPT(NDEG,NNODE),FFN(ITEL,ITEL),FROTN(ITEL,ITEL),
3  STRECHN(ITEL),EIGVN(ITEL,ITEL),FFN1(ITEL,ITEL),
4  FROTN1(ITEL,ITEL),STRECHN1(ITEL),EIGVN1(ITEL,ITEL),
5  Fe(ITEL,ITEL)
  DIMENSION MATUS(2),KCUS(2)

c   Input:
c   E is the total elastic mechanical strain.
c   DE is the increment of mechanical strain.
c   t = n
c   t = n+1
c   T is the state variables (comes in at t = n; must be updated to have state
variables at t = n +1).
c   DT is the increment of state variables.
c   NGENS is the size of the stress-strain law.
c   N is the element number.
c   NN is the integration point number.
c   KCUS(1) is your layer number (always 1 for continuum elements).
c   KCUS(2) is the internal layer number (always 1 for continuum element).
c   MATUS(1) is the user material identifier.
c   MATUS(2) is the internal material identifier.
c   NDI is the number of direct components.
c   NSHEAR is the number of shear components.
c   DISP is the incremental displacements.
c   DISPT is the displacements at t = n (at assembly lov1 = 4) and the
```

```

c      displacements at t = n +1 (at stress recovery lvl = 6).
c      COORD is the coordinates.
c      NCRD Is the number of coordinates.
c      NDEG is the number of degrees of freedom.
c      ITEL is the dimension of F and R; 2 for plane-stress and 3 for the rest of
the cases.
c      NNODE is the number of nodes per element.
c      JTYPE is the element type.
c      LCLASS is the element class.
c      IFR is set to 1 if R has been calculated.
c      IFU is set to 1 if STRECH has been calculated.
c      At t = n (or the beginning of the increment):
c      FFN is the deformation gradient.
c      FROTN is the rotation tensor.
c      STRECHN is the square of principal stretch ratios, lambda (i).
c      EIGVN (I,J) I principal direction components for J eigenvalues.
c      At t = n +1 (or the current time step):
c      FFN1 is the deformation gradient.
c      FROTN1 is the rotation tensor.
c      STRECHN1 is the square of principal stretch ratios, lambda (i).
c      EIGVN1(I,J) is the I principal direction components for J eigenvalues.
c
c      Enter to initialization subroutine until all the necessary initializations
are complete
      if (inc==0) then
c          call initialize_all(N(2),NN,T,dummyS,dummyD)
          call initialize_all(N(1),NN,S,D)

      else
c          After all initializations are complete enter to calculation loop
c          call calculate(FFN,FFN1,timinc,ncycle,inc,N(2),
c          &                NN,dummyS,dummyD)
          call
calculate(FFN,FFN1,timinc,ncycle,inc,int(subinc),N(1),NN,S,D,Fe)
      endif
c      S=dummyS(1:4)
c      D=dummyD(1:4,1:4)
c
c      write(6,*) 'ITEL'
c      write(6,*) ITEL
c      DO I = 1,SIZE(El_MisTr)
          if (N(1)==El_MisTr(1,I)) then
              if (NN==1) then
                  write(6,*) 'inc_flag'
                  write(6,*) inc
                  write(6,*) 'Element_number_flag'
                  write(6,*) N(1)
                  write(6,*) 'IP_number_flag'
                  write(6,*) NN
                  write(6,*) 'F_elastic_flag'
                  write(6,*) Fe
              endif
          endif
      end do
c      write(6,*) 'FFN1'
c      write(6,*) FFN1
c      write(6,*) 'jacobi'
c      write(6,*) D
c      write(6,*) 'stress'
c      write(6,*) S

```

```

c      write(6,*) 'ncycle'
c      write(6,*) ncycle
c      write(6,*) 'incsub'
c      write(6,*) incsub
c      RETURN
c      END

c
c      Only at the beginning increment
c      SUBROUTINE UBGINC(INC,INCSUB)
c      IMPLICIT REAL *8 (A-H, O-Z)
c      include 'spacevec'
c      include 'dimen'
c10    format (i4,3f10.3)
c      write(6,*) ''
c      write(6,*) ''
c      write(6,*) '*****'
c      write(6,*) 'Only at the beginning of the increment!'
c      Some important numbers to be outputted
c      write(6,*) 'Number of elements: ', numel
c      write(6,*) 'Number of nodes: ', numnp
c      write(6,*) 'Number of dof per node: ', ndeg
c      write(6,*) 'Number of coordinate directions: ', ncrd
c      Nodal coordinates
c      write(6,*) 'Nodal coordinates: # - X - Y - Z'
c      do i=1,numnp
c          write(6,10) i,(xord_d((i-1)*ncrd+j),j=1,ncrd)
c      enddo
c      write(6,*) '*****'
c      RETURN
c      END
c
c      SUBROUTINE PLOTV(V,S,SP,ETOT,EPLAS,ECREEP,T,M,NN,KCUS,NDI,
c      &NSHEAR,JPLTCD)
c      use globalvars, only : global_mis,global_gammadot_av,
c      &global_rhoSSD_av
c      IMPLICIT REAL*8 (A-H, O-Z)
c      DIMENSION S(*),SP(*),ETOT(*),EPLAS(*),ECREEP(*)
c      integer M,NN

c
c
c      First user defined variable, average shear rate (1/s)
c      if (jpltcd==1) then
c          v=global_gammadot_av(M,NN)
c          return
c      endif

c      First user defined variable, average SSD density (1/m2)
c      if (jpltcd==2) then
c          v=global_rhoSSD_av(M,NN)
c          return
c      endif

c
c      Second user defined variable, misorientation (deg)
c      if (jpltcd==2) then
c          v=global_mis(M,NN)
c          return
c      endif

c
c      RETURN
c      END

```

```

c Mohammad Hasan Joudivand
c Jan. 09, 2020
c
c      module calculations
c      contains
c
c      This subroutine calculates the two main variables: Stress and Consistent
c      tangent
c      subroutine calculate(F_t,F,dt,ncycle,inc,subinc,el_no,ip_no,
c      &stress,jacob,Fe)
c      use globalvars, only: global_Fp,global_Fp_t,
c      &global_Fe,global_Fe_t,global_rhoSSD,
c      &global_rhoSSD_t,I3,I6,inc_old,subinc_old,global_gammadot_av,
c      &global_rhoSSD_av,global_jacob,innoitmax,ounoitmax,global_ori,
c      &global_mis,global_stress,global_jacob,global_jacob_t,njaco
c      use globalsubs, only: convert3x3to6,invert3x3,skew,misorientation,
c      &polar
c      implicit none
c      Inputs
c      integer ncycle,el_no,ip_no,inc,subinc,i,j,is
c      real(8) F_t(3,3),F(3,3),dt
c      Outputs
c      real(8) stress(6),jacob(6,6)
c      Variables used within this subroutine
c      real(8) Fp_t(3,3),Fe_t(3,3),rhoSSD_t(24)
c      real(8) Fp(3,3),rhoSSD(24),Cauchy(3,3)
c      real(8) invFp(3,3),PK2_vec(6),gammadot(24),dgammadot_dtau(24)
c      real(8) Lp(3,3),Fe(3,3),rhoM(24),detFp
c      integer innoit,ounoit,notconv
c      real(8) gammadot_av,rhoSSD_av
c
c      If the FE increment is updated (FE accepts the solution
c      and moves to a different increment then do the assignment)
c      if (inc_old.ne.inc) then
c          write(6,*) 'Updated the state variables'
c          inc_old=inc
c          global_Fp_t=global_Fp
c          global_Fe_t=global_Fe
c          global_rhoSSD_t=global_rhoSSD
c          global_jacob_t=global_jacob
c          subinc_old=0
c      end if
c
c
c      If there is a sub increment
c      new subincrement? update starting with subinc 2
c      if(subinc.gt.subinc_old) then
c          write(6,*) 'Updated the state variables'
c          write(6,*) 'Sub-incrementation!!!'
c          global_Fp_t=global_Fp
c          global_Fe_t=global_Fe
c          global_rhoSSD_t=global_rhoSSD
c          global_jacob_t=global_jacob
c          subinc_old=subinc
c      end if

```

```

Fp_t=global_Fp_t(el_no,ip_no,,:)
Fe_t=global_Fe_t(el_no,ip_no,,:)

rhoSSD_t=global_rhoSSD_t(el_no,ip_no,:)

call SC_main(F,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,dt,
&invFp,PK2_vec,gammadot,dgammadot_dtau,Lp,rhoSSD,
&rhoM,innoit,ounoit,Fe,Fp,stress,gammadot_av,rhoSSD_av)

C
C

C      write(6,*) 'innoit', innoit
C      write(6,*) 'ounoit', ounoit

C      Store the important results
      global_Fp(el_no,ip_no,,:)=Fp
      global_Fe(el_no,ip_no,,:)=Fe

      global_rhoSSD(el_no,ip_no,:)=rhoSSD

      global_rhoSSD_av(el_no,ip_no)=rhoSSD_av
      global_gammadot_av(el_no,ip_no)=gammadot_av

      global_stress(el_no,ip_no,:)=stress

C
C

      if (innoit.ge.innoitmax) then
        write(6,*) 'Inner loop during stress calculation
&      did not converge!'
        write(6,*) 'el_no'
        write(6,*) el_no
        write(6,*) 'ip_no'
        write(6,*) ip_no
        write(6,*) 'inner no it'
        write(6,*) innoit
        write(6,*) 'outer no it'
        write(6,*) ounoit
        write(6,*) 'F'
        write(6,*) F
        write(6,*) 'F_t'
        write(6,*) F_t
        write(6,*) 'Fp'
        write(6,*) Fp
        write(6,*) 'Fp_t'
        write(6,*) Fp_t
        write(6,*) 'rhoSSD_t'
        write(6,*) rhoSSD_t
        write(6,*) 'rhoSSD'

```

```

        write(6,*) rhoSSD
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
    call quit(9101)
endif
c
    if (ounoit.ge.ounoitmax) then
        write(6,*) 'Outer loop during stress calculation
&    did not converge!'
        write(6,*) 'el_no'
        write(6,*) el_no
        write(6,*) 'ip_no'
        write(6,*) ip_no
        write(6,*) 'inner no it'
        write(6,*) innoit
        write(6,*) 'outer no it'
        write(6,*) ounoit
        write(6,*) 'F'
        write(6,*) F
        write(6,*) 'F_t'
        write(6,*) F_t
        write(6,*) 'Fp'
        write(6,*) Fp
        write(6,*) 'Fp_t'
        write(6,*) Fp_t
        write(6,*) 'rhoSSD_t'
        write(6,*) rhoSSD_t
        write(6,*) 'rhoSSD'
        write(6,*) rhoSSD
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
    call quit(9102)
endif

```

```

c      For a number of increments do the jacobian calculation
c      Note, jacobian is needed at the first calculation
    if (modulo(inc,njaco).eq.0) then
c      Calculate the material tangent (using perturbation)
        call SC_jacobian(F,F_t,Fe_t,Fp_t,rhoSSD_t,
&dt,el_no,ip_no,stress,jacob,notconv)
c      When the jacobian calculation did not converge, assign the old
jacobian!
        if (notconv.eq.1) then
            write(6,*) 'el_no', el_no
            write(6,*) 'ip_no', ip_no
            write(6,*) 'Jacobian has not converged!!!'
c            Jacobian
            jacob=global_jacob_t(el_no,ip_no,:,:)
            return
        endif
    else

```

```

c          Note this also works when inc=1 since it is elasticity matrix
          jacob=global_jacob_t(el_no,ip_no,:,:)
endif

c  Assign the value of jacobian if there is no convergence
global_jacob(el_no,ip_no,:,:)=jacob

c
c  Calculate the outputs
cc  Misorientation with respect to the initial orientation
cc  Initial orientation
c  g1=global_ori(el_no,ip_no,:,:)
cc  Polar decompose the elastic part of the deformation gradient
c  call polar(Fe_T,R,U,sing)
cc  If singularity does no exist
cc  otherwise old solution will remain.
c  if (sing.eq.0)      then
cc      Calcualte misorientation using the subroutine
c      call misorientation(g1,R,1,ax,ang,dg)
cc      Store the misorientation value
c      global_mis(el_no,ip_no)=ang
c  endif

c  write(6,*) 'el_no'
c  write(6,*) el_no
c  write(6,*) 'ip_no'
c  write(6,*) ip_no
c  write(6,*) 'stress'
c  write(6,*) stress
c  write(6,*) 'jacob'
c  write(6,*) jacob

return
end subroutine

c
c
c

c
c  This subroutine calculates consistent tangent
c  The effect of non-local deformations at the former time step is included
only!
      subroutine SC_jacobian(F,F_t,Fe_t,Fp_t,
&rhoSSD_t,dt,el_no,ip_no,Cauchy_vec,
&jacob,notconv)
      use globalvars, only : deps,innoitmax,ounoitmax,global_jacob
      use globalsubs, only : convert6to3x3
      implicit none
c  Inputs
real(8) F_t(3,3),F(3,3),Fe_t(3,3),dt
real(8) Fp_t(3,3),rhoSSD_t(24),Cauchy(3,3)
c  Outputs
real(8) jacob(6,6)
integer notconv,el_no,ip_no
c  Variables used within this subroutine

```

```

real(8) Cauchy_per(3,3)
real(8) F_per(3,3),dFrel_vec(6),dFrel(3,3),Cauchy_per_vec(6)
real(8) Cauchy_vec(6),PK2_per_vec(6)
real(8) invFp_per(3,3),dummy2(24),dummy3(24),dummy4(3,3)
real(8) dummy5(24),dummy6(24),dummy7(24)
real(8) dummy8(3,3),dummy9(3,3),dummy10,dummy11
integer i,j,innoit,ounoit
c      Assign the convergent behavior
notconv=0
jacob=0.0

cc      Calculate inverse of the deformation gradient
c      call invert3x3(F,invF,detF)
c
cc      Calculate velocity gradient
c      L = (I3 - matmul(F_t,invF))/dt
c
c      dum = I3 - L*dt_per
c      call invert3x3(dum,dum,dummy10)
c
cc      Calculate the FUTURE defomation gradient
c      F_per_new = matmul(dum,F)
c
c      do i=1,3
c          write(6,*) (F_per_new(i,j),j=1,3)
c      enddo

do i=1,6

c          Component-wise pertubation
dFrel_vec=0.0

c      dFrel_vec(i)=deps
      if (i.le.3) then
          dFrel_vec(i)=deps
c          Note it is not deps/2 since during conversion only one component is
considered
      else
          dFrel_vec(i)=deps/2.0
      endif

c          Convert the vector to a matrix
      call convert6to3x3(dFrel_vec,dFrel)

      F_per=F+matmul(dFrel,F_t)

!      write(6,*)'F_per'
!do k=1,3
!      write(6,*) (F_per(k,j),j=1,3)
!enddo
!
!write(6,*)'F'
!do k=1,3
!      write(6,*) (F(k,j),j=1,3)
!enddo

c          Call the calculation procedure
      call SC_main(F_per,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,

```

```

&dt,invFp_per,PK2_per_vec,dummy2,dummy3,dummy4,dummy5,
&dummy7,innoit,ounoit,dummy8,dummy9,Cauchy_per_vec,dummy10,dummy11)
C
!
! write(6,*) 'Cauchy_per_vec'
! do j=1,6
!     write(6,*) Cauchy_per_vec(j)
! enddo
!
! write(6,*) 'Cauchy_vec'
! do j=1,6
!     write(6,*) Cauchy_vec(j)
! enddo
!c
C
!
! if ((innoit.ge.innoitmax).or.(ounoit.ge.ounoitmax)) then
!     notconv=1
!     return
! endif
C
!c
! Calculate the elastic part of the deformation
! Fe_per=matmul(F_per,invFp_per)
!
! Calculate Cauchy stress
! call cauchystress(PK2_per_vec,Fe_per,Cauchy_per,Cauchy_per_vec)
!c
!
! Assignment of jacobian components
!
! jacob(:,i)=(Cauchy_per_vec-Cauchy_vec)/deps
!
!
! enddo
C
!c
! Make it symmetric
! jacob=(transpose(jacob)+jacob)/2.0
!
!
!
!c
! jacob=zeta66
!
! return
! end subroutine
C
C
C
C main subroutine that calculates the stress and correct Fp through
C Dawson-Maniatty intergration method
C This contains all the sub-routines called during the calculation
C INPUTS: F(3,3), Fe_t(3,3), Fp_t(3,3), tauc_t(24), dt
C OUPUTS: invFp(3,3), PK2_vec(6), gammadot(24), dgammadot_dtau(24),
C         Lp(3,3), tauc(24), initno, outno
C USES: scale, innertol, outertol, innoitmax, ounoitmax

```

```

subroutine SC_main(F,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,dt,invFp,PK2_vec,
&gammadot,dgammadot_dtau,Lp,rhoSSD,rhoM,innoit,ounoit,Fe,Fp,stress,
&gammadot_av,rhoSSD_av)
  use globalvars, only : ounoitmax, innoitmax, innertol, outertol,
&factor,rhoSSD0
  use globalsubs, only : invert3x3
  implicit none
c   Input variable declarations
  real(8) F(3,3),Fe_t(3,3),Fp_t(3,3)
  real(8) rhoSSD_t(24),dt
  integer el_no,ip_no
c   Output variable declarations
  real(8) invFp(3,3),PK2_vec(6),gammadot(24),dgammadot_dtau(24)
  real(8) Lp(3,3),rhoSSD(24),rhoM(24)
  integer ounoit,innoit
c   Variables used within this subroutine
  integer i,j,is
  real(8) detFp,invFe_t(3,3),Fp(3,3),C(3,3),Ce(3,3),taueff(24)
  real(8) normR,R_vec(9),inres,inres0,Fp_temp(3,3),detFe,sigma(24)
  real(8) dFp(3,3),rhoSSD_dot(24)
  real(8) R_rhoSSD(24),outres
  real(8) taupass(24),Vol(24),vel_con(24),mob_con(24),rho_f(24)
  real(8) Fe(3,3),stress(6),Cauchy(3,3),gammadot_av,rhoSSD_av
c
c
c   Initial fully-plastic guess
  call invert3x3(Fe_t,invFe_t,detFe)
  Fp=matmul(invFe_t,F)
  Fp_temp=Fp_t
c   Overall Right-Cauchy-Green tensor
  C=matmul(transpose(F),F)
c   Assign initial variables
  rhoSSD=rhoSSD_t
c   write(6,*) 'rhoSSD', rhoSSD

c
c   Outer loop starts HERE!
  do ounoit=1,ounoitmax
c       Assign a high value for the residual (Necessary for scale-back
algorithm)
      inres=1.0d+10
c
c       Calculate parameters that depend on the state variables at the
outer level of iterations
      call statedeptconsts(rhoSSD,taupass,Vol,vel_con, mob_con,rho_f)

c
c       write(6,*) 'rho_f', rho_f
c       Inner loop starts HERE!
      do innoit=1,innoitmax
c           Inverse of the plastic deformation gradient
          call invert3x3(Fp,invFp,detFp)
c
c           Constitutive calculations
          write(6,*) ',innoit',innoit
c
c           write(6,*) 'PK2_vec', PK2_vec
          call
constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,ip_no,
&Ce,PK2_vec,taueff,sigma,gammadot,dgammadot_dtau,Lp,rhoM)
c

```

```

C          write(3,*) 'tau(1)',tau(1)
C          write(3,*) 'C'
C          do i=1,3
C              write(3,*) (C(i,j),j=1,3)
C          enddo
C          write(3,*) 'invFp'
C          do i=1,3
C              write(3,*) (C(i,j),j=1,3)
C          enddo
C          write(6,*) 'taupass',taupass
C      write(6,*) 'tau',tau
C          write(6,*) 'gammadot',gammadot
C          write(3,*) 'rhoM(1)',rhoM(1)
C          write(3,*) 'velcon',velcon
C          write(3,*) 'Vol',Vol
C          write(3,*) 'dgammadot_dtau(1)',dgammadot_dtau(1)
C          Residual calculation
C          call residual(invFp,Fp_t,Lp,detFp,dt,R_vec,normR)
C          write(3,*) 'R_vec'
C          write(3,*) R_vec
C          write(6,*) 'normR',normR
C          Assign old value of residual
C          inres0=inres
C          inres=normR
C
C          If the residual is smaller than the tolerance exit the loop
C          if (normR.lt.innertol) then
C              write(3,*) innoit
C              write(3,*) 'converged'
C              exit
C          If the tolerance is increasing scale back the initial guess
C          elseif ((normR.gt.innertol).and.(inres.gt.inres0)) then
C              write(3,*) innoit
C              write(3,*) 'scaled-back the initial guess'
C              write(3,*) 'Fp',Fp
C              write(3,*) 'Fp_temp',Fp_temp
C              Fp=((1.0-factor)*Fp)+(factor*Fp_temp)
C              inres=1.0d+10
C          else otherwise calculate the tangent for Newton-Raphson
C      scheme
C          else
C              write(3,*) innoit
C              call tangent(R_vec,Ce,invFp,detFp,
C                  & Fp_t,dgammadot_dtau,el_no,ip_no,dt,dFp)
C
C              write(3,*) 'invFp'
C              do i=1,3
C                  write(3,*) (invFp(i,j),j=1,3)
C              enddo
C              write(3,*) 'dFp'
C              do i=1,3
C                  write(6,*) (dFp(i,j),j=1,3)
C              enddo
C
C              write(3,*) 'iterate with tangent'
C              Store the current Fp as old Fp since it may be used
C      in scaling-back algorithm
C              Fp_temp=Fp
C              The new deformation gradient after NR iteration
C              Fp=Fp+dFp
C              write(3,*) 'dFp'

```

```

c                                     write(3,*) dFp
c                                     endif
c                                     enddo
c                                     write(3,*) 'innoit: ',innoit
c                                     END of INNER iteration loop
c                                     Calculate amount of slip system hardening
c                                     call hardening(gammadot,taueff,sigma,rhoSSD,rhoM,taupass,
& rho_f,rhoSSD_dot)

c                                     if (ip_no.eq.8) then
c                                     write(6,*) 'rhoSSD_dot(1)',rhoSSD_dot(1)
c                                     write(6,*) 'gammadot(1)',gammadot(1)
c                                     endif

c                                     Residual increments of slip resistance
c                                     R_rhoSSD=dabs(rhoSSD-rhoSSD_t-(rhoSSD_dot*dt))/minval(rhoSSD)

c                                     Relative maximum change in slip system resistivity
c                                     outres=maxval(R_rhoSSD)

c                                     write(6,*) '*****'
c                                     write(6,*) 'R_rhoSSD',R_rhoSSD
c                                     write(6,*) 'outres', outres
c                                     write(6,*) '*****'

c                                     Check if the change of hardening is within the tolerances
c                                     if(outres.lt.outertol)then
c                                     exit
c                                     endif

c                                     If not accept the result and continue the implicit iteration
c                                     rhoSSD=rhoSSD_t+(rhoSSD_dot*dt)

c                                     enddo
c                                     END of OUTER iteration loop
c                                     write(3,*) 'ounoit: ',ounoit
c
c
c
c
c                                     call statedeptconsts(rhoSSD,taupass,Vol,vel_con, mob_con,rho_f)

c                                     call constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,ip_no,
& Ce,PK2_vec,taueff,sigma,gammadot,dgammadot_dtau,Lp,rhoM)

c                                     Calculate the elastic part of the deformation
c                                     Fe=matmul(F,invFp)

c                                     Calculate Cauchy stress
c                                     call cauchystress(PK2_vec,Fe,Cauchy,stress)

c                                     Average of the shear rates
c                                     gammadot_av=0.0
c                                     do is=1,24

```

```

        gammadot_av=gammadot_av+dabs(gammadot(is))
    enddo
    gammadot_av=gammadot_av/24.0

c    Average of the SSD densities
    rhoSSD_av=0.0
    do is=1,24
        rhoSSD_av=rhoSSD_av+rhoSSD(is)
    enddo
    rhoSSD_av = rhoSSD_av/24.0

    return
end subroutine

c
c
c
c
c
c    This subroutine includes the constitutive calculations
c    INPUTS: C(3,3), invFp(3,3), tauc(24)
c    OUTPUTS: Ce(3,3), PK2_vec(6), tau(24), gammadot(24), dgammadot_dtau(24),
Lp(3,3)
c    USES: Schmid(24,3,3), Schmid_vec(24,6), zeta66(6,6), gammadot0, mm,
I3(3,3)
    subroutine constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,
&ip_no,Ce,PK2_vec,taueff,sigma,gammadot,dgammadot_dtau,Lp,rhoM)
    use globalvars, only : I3,zeta66,Schmid,KBT,
&Schmid_vec,Climb_vec,Burger,B0,rhoSSD0,NonSchmid1_vec,
&NonSchmid2_vec,NonSchmid3_vec,a1,a2,a3, BCF
    use globalsubs, only : convert3x3to6
    implicit none
c    Input variable declarations
    real(8) C(3,3),invFp(3,3)
    integer el_no,ip_no
    real(8) taupass(24),vel_con(24),mob_con(24),Vol(24)
c    Output variable declarations
    real(8) Ce(3,3),PK2_vec(6),taueff(24),sigma(24),rhoM(24)
    real(8) gammadot(24),dgammadot_dtau(24),Lp(3,3)
c    Variables used within the code
    real(8) E(3,3),E_vec(6),PK2(3,3)
    real(8) vel,dvel_dtau,velth,dvelth_dtau,veldr,dveldr_dtau
    real(8) drhoM_dtau,tau(24)
    real(8) tau_ns(24),tau_ns1(24),tau_ns2(24),tau_ns3(24)
    integer xx,i

c
c
c
c    open(unit=3,file="output.txt")

c    Elastic-Right Cauchy-Green tensor
    Ce=matmul(transpose(invFp),matmul(C,invFp))

c
c    Green-Lagrange strain tensor
    E=(Ce-I3)/2.0d+0

c
c    Vectorized strain (Notation as Reid, Def. Geo.)
    call convert3x3to6(E,E_vec)

c
c

```

```

c      Vectorized 2nd Piola-Kirchoff stress tensor through Elastic constitutive
law
      PK2_vec=matmul(zeta66,E_vec)

c
c      Calculation of plastic part of the velocity gradient
      Lp=0.0d+0
      taueff=0.0d+0
      tau=0.0d+0
      tau_ns1=0.0d+0
      tau_ns2=0.0d+0
      tau_ns3=0.0d+0

      do xx=1,24

c          Calculate resolved shear stress
          do i=1,6
              tau(xx)=tau(xx)+BCF*Schmid_vec(xx,i)*PK2_vec(i)
              tau_ns1(xx)=tau_ns1(xx)+a1
              & *BCF*NonSchmid1_vec(xx,i)*PK2_vec(i)
              tau_ns2(xx)=tau_ns2(xx)+a2
              & *BCF*NonSchmid2_vec(xx,i)*PK2_vec(i)
              tau_ns3(xx)=tau_ns3(xx)+a3
              & *BCF*NonSchmid3_vec(xx,i)*PK2_vec(i)
          enddo

          tau_ns(xx)=tau_ns1(xx)+tau_ns2(xx)+tau_ns3(xx)

c          Calculate climb stress
          sigma(xx)=0.0d+0
          do i=1,6
              sigma(xx)=sigma(xx)+BCF*(Climb_vec(xx,i)*PK2_vec(i))
          enddo

c          Effective stress
          taueff(xx)=tau(xx)+tau_ns(xx)-taupass(xx)

c          If the applied stress is greater than the passing stress
          if ((taueff(xx).gt.0d+0).and.(tau(xx).gt.0d+0)) then

c
c              write(6,*)'Hello Im here!'
c
c              write(6,*)'tau_ns',tau_ns(xx)
c              write(6,*)'tau',tau(xx)

c              Thermally activated slip velocity
              velth=vel_con(xx)*dexp(taueff(xx)*Vol(xx)/KBT)

c              write(6,*)'velth',velth

c              and its derivative wrto shear stress
dvelth_dtau=vel_con(xx)*dexp(taueff(xx)*Vol(xx)/KBT)*Vol(xx)/KBT

c              Mobile dislocation
              rhoM(xx)=mob_con(xx)*taueff(xx)**2.0d+0
c              Derivative of mobile dislocation density wrto shear stress
              drhoM_dtau=2.0d+0*mob_con(xx)*taueff(xx)

c              Drag velocity
              velldr = taueff(xx)*Burger/B0

```



```

c      Lp from calculated shear rates
      R1=Lp
c      Lp from fully-implicit integration using plastic part of the deformation
gradient
      R2=-(I3-matmul(Fp_t,invFp))/dt
c      Residual from the determinant
      R3=lambdap*(detFp-1.0d+0)*I3
c      Total residual
      R=R1+R2+R3

c
c
c      Vectorize residual
      call convert3x3to9(R,R_vec)
c      Norm of the residual
      call normvec(R_vec,9,normR)
c
      return
      end subroutine

c
c
c
c
c      This subroutine calculates the tangent and the increment in plastic part
of the deformation gradient
c      for the Newton-Raphson scheme
c      INPUTS:          R_vec(9), Ce(3,3), invFp(3,3), detFp, Fp_t(3,3),
dgamma_dot_dtau(24), dt
c      OUTPUTS:         dFp(3,3)
c      USES:            Schmid(24,3,3), zeta3333(3,3,3,3), lambda_p, I3(3,3)
      subroutine tangent(R_vec,Ce,invFp,detFp,Fp_t,
&                        dgamma_dot_dtau,el_no,ip_no,dt,dFp)
&
      use globalvars, only: lambdap,I3,Schmid,zeta3333,
&NonSchmid1, NonSchmid2, NonSchmid3, a1, a2, a3,BCF
      use globalsubs, only: convert3x3x3x3to9x9,invertnxn,convert9to3x3
      implicit none
c      Input variable declarations
      real(8) R_vec(9),Ce(3,3),invFp(3,3),detFp,Fp_t(3,3)
      real(8) dgamma_dot_dtau(24), dt
      integer el_no,ip_no
c      Output variable declaration
      real(8) dFp(3,3)
c      Variables used within this subroutine
      real(8) D1(3,3,3,3),D2(3,3,3,3),D3(3,3,3,3),D(3,3,3,3),det
      real(8) sum1a(24,3,3),sum1b(24,3,3),const_D1
      real(8) sum2,D9x9(9,9),invD9x9(9,9),dFp_vec(9)
      integer i,j,k,l,m,n,p,r,s,xx

c
c
c
c
c      Calculation of 1st component of the 4th order tangent, D1
      sum1a=0.0d+0
      sum1b=0.0d+0
      do k=1,3
         do l=1,3
            do xx=1,24
               do m=1,3
                  do n=1,3
                     const_D1=BCF*Schmid(xx,m,n)+
&(a1*BCF*NonSchmid1(xx,m,n))+(a2*BCF*NonSchmid2(xx,m,n))+
&(a3*BCF*NonSchmid3(xx,m,n))

```

```

do p=1,3
do r=1,3
do s=1,3
C      Calculation of D1a
sum1a(xx,k,l)=sum1a(xx,k,l)+((-1.0/2.0)*dgammadot_dtau(xx)*
&((invFp(1,m)*Ce(k,p))+(Ce(m,k)*invFp(1,p)))*
&zeta3333(p,n,r,s)*(Ce(r,s)-I3(r,s))*const_D1)
C      Calculation of D1b
sum1b(xx,k,l)=sum1b(xx,k,l)+((-1.0/2.0)*dgammadot_dtau(xx)*
&Ce(m,p)*zeta3333(p,n,r,s)*((invFp(1,r)*Ce(k,s))+
&(Ce(r,k)*invFp(1,s)))*const_D1)
enddo
enddo
enddo
enddo
enddo
enddo
enddo
enddo
enddo
enddo
C      D1=0.0d+0
do i=1,3
do j=1,3
do k=1,3
do l=1,3
do xx=1,24
D1(i,j,k,l)=D1(i,j,k,l)+
&(BCF*Schmid(xx,i,j)*(sum1a(xx,k,l)+sum1b(xx,k,l)))
enddo
enddo
enddo
enddo
enddo
C
CC
C      Calculation of 2nd and 3rd components of the 4th order tangent, D2 and D3
do i=1,3
do j=1,3
do k=1,3
do l=1,3
sum2=0.0
do m=1,3
C      Calculation of D2
sum2=sum2-
&(Fp_t(i,m)*invFp(m,k)*invFp(1,j)/dt)
enddo
D2(i,j,k,l)=sum2
C      Calculation of D3
D3(i,j,k,l)=I3(i,j)*lambdap*detFp*invFp(1,k)
enddo
enddo
enddo
enddo
D=D1+D2+D3
C
C      Convert the jacobian to 9x9 matrix
call convert3x3x3x3to9x9(D,D9x9)
C
C      Invert the Jacobian
call invertnxn(D9x9,9,invD9x9,det)
C

```

```

C
C      XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
C      A check must be placed here to check if the jacobian is invertible or not
C      XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
C
C      Increment for the deformation gradient
C      dFp_vec=matmul(-invD9x9,R_vec)
C
C      Convert the vector to 2nd order tensor
C      call convert9to3x3(dFp_vec,dFp)
C
C
C      return
C      end subroutine
C
C
C      This subroutine calculates the state variables before going into the inner
loop
C      INPUTS:          gammadot(24), tauc(24)
C      OUTPUTS:         dtauc(24)
C      USES:            h0, ss, a, hardmat
C      subroutine statedeptconsts(rhoSSD,taupass,Vol,vel_con,mob_con,rho_f)
C      use globalvars, only: G,Burger,c1,c2,c3,m1,m2,intmat,proj
C      use globalsubs, only : matmulvec
C      implicit none
C      Input variable declarations
C      real(8) rhoSSD(24)
C      Output variable declarations
C      real(8) taupass(24),Vol(24),vel_con(24),mob_con(24),rho_f(24)
C      Variables used within this subroutine
C      real(8) lambda
C      integer xx
C
C
C      Pass the dislocation density from the interaction matrix
C      rho_f = matmul(proj,rhoSSD)
C
C      write(6,*) 'rho_f',rho_f
C
C
C      do xx=1,24
C
C          Long-range stress
C          taupass(xx)=c1*G*Burger*dsqrt(rhoSSD(xx))
C
C          Jump distance
C          lambda=c2/dsqrt(rhoSSD(xx))
C
C          Activation volume
C          Vol(xx)=c3*(Burger**2.0)*lambda
C          Vol(xx)=c3*Burger**3.0
C
C          Thermally activated velocity multiplier
C          vel_con(xx)=m1*lambda
C
C          Mobile dislocation density multiplier
C          if (rho_f(xx).lt.0) then
C              write(6,*) '- forest'
C          endif
C
C          mob_con(xx)=m2*rhoSSD(xx)/rho_f(xx)
C      enddo
C
C      return
C      end subroutine

```

```

C
C
C
C
C
C      This subroutine calculates the amount of hardening for a given shear rate
C      INPUTS:          gammadot(24), tauc(24)
C      OUTPUTS:         dtauc(24)
C      USES:            h0, ss, a, hardmat
C      subroutine hardening(gammadot,taueff,sigma,rhoSSD,rhoM,
&taupass,rho_f,rhoSSDdot)
C      use globalvars, only: c4,m3,m4,m5,m6,Burger,Dv,acDc,Qcs,KBT,intmat
C      implicit none
C      Input variable declarations
C      real(8) gammadot(24),rhoSSD(24),rhoM(24)
C      real(8) taueff(24),sigma(24),taupass(24),rho_f(24)
C      Output variable declarations
C      real(8) rhoSSDdot(24)
C      Variables used within this subroutine
C      real(8) rhoSSDdot1(24),rhoSSDdot2(24),rhoSSDdot3(24)
C      real(8) rhoSSDdot4(24),g,P,Deff
C      integer xx
C
C      rhoSSDdot = 0.0d+0
C      rhoSSDdot1 = 0.0d+0
C      rhoSSDdot2 = 0.0d+0
C      rhoSSDdot3 = 0.0d+0
C      rhoSSDdot4 = 0.0d+0
C
C      do xx=1,24
C
C      Evolution rate of EDGE dislocaitons
C      Hardening due to mean free path
C      rhoSSDdot1(xx)=c4*dsqrt(rhoSSD(xx))/Burger*gammadot(xx)
C
C      Annihilation by climb
C      Deff = Dv + rhoSSD(xx)*acDc
C      rhoSSDdot3(xx)=-m4*Deff*(rhoSSD(xx)**2.0)*dabs(sigma(xx))*
&gammadot(xx)
C
C      if (taueff(xx).gt.0d+0) then
C
C
C      Immobilization of mobile dislocations by dipole formation
C      rhoSSDdot2(xx)=m3*rhoM(xx)/taueff(xx)*gammadot(xx)
C
C
C      write(6,*) 'rhoSSDdot1', rhoSSDdot1
C      write(6,*) 'rhoSSDdot2', rhoSSDdot2
C      write(6,*) 'rhoSSDdot3', rhoSSDdot3
C

```

```

c          Evolution rate of SCREW dislocaitons
c          Annihilation by cross-slip
c          Activation energy for cross-slip
c          g = m6*taueff(xx)
c          Probability of cross-slip
c          P = dexp(-Qcs/KBT)*dsinh(g/KBT)
c          rhoSSDdot4(xx)=-m5*dsqrt(rhoSSD(xx))*P*gammadot(xx)
c          write(6,*) 'rhoSSDdot4', rhoSSDdot4

c
c

c          turn hardening off
c          rhoSSD_dot(xx) = 0.0d+0

endif

enddo

c
c          Total rate of change of edge dislocation density
c          rhoSSDdot = rhoSSDdot1 + rhoSSDdot2 + rhoSSDdot3 + rhoSSDdot4
c          rhoSSDdot = matmul(intmat,rhoSSDdot)

c          Divide the slip rate into half due to sign of the system (both +/- slip
c          considered)
c          rhoSSDdot = 0.5 * rhoSSDdot

return
end subroutine

c
c
c          This subroutine calculates the Cauchy stress
c          INPUTS:      PK2_vec(6), invFp(3,3)
c          OUTPUTS: Cauchy(3,3), Cauchy_ve(6)
c          subroutine cauchystress(PK2_vec,Fe,Cauchy,Cauchy_vec)
c          use globalsubs, only: convert6to3x3,determinant,convert3x3to6
c          implicit none
c          Input variable declarations
c          real(8) PK2_vec(6), Fe(3,3)
c          Input variable declarations
c          real(8) Cauchy(3,3),Cauchy_vec(6)
c          Variables used within this subroutine
c          real(8) detFe,PK2(3,3)

c          2nd PK stress
c          call convert6to3x3(PK2_vec,PK2)
c          Determinant of the elastic part of the overall deformation
c          call determinant(Fe,detFe)
c          Cauchy stress
c          Cauchy=matmul(matmul(Fe,PK2),transpose(Fe))/detFe
c          Vectorize Cauchy stress
c          call convert3x3to6(Cauchy,Cauchy_vec)

return

```

```

        end subroutine
c
c
c
c
c
c
c
        end module

```

6.2 HYPELA2-CPJ2 Fortran subroutine

```

c Mohammad Hasan Joudivand
c Jan. 09, 2020
c
        module calculations
        contains
c
c      This subroutine calculates the two main variables: Stress and Consistent
c      tangent
        subroutine calculate(F_t,F,dt,ncycle,inc,subinc,el_no,ip_no,
&stress,jacob,Fe)
        use globalvars, only: global_Fp,global_Fp_t,
&global_Fe,global_Fe_t,global_rhoSSD,
&global_rhoSSD_t,I3,I6,inc_old,subinc_old,global_gammadot_av,
&global_rhoSSD_av,global_jacob,innoitmax,ounoitmax,global_ori,
&global_mis,global_stress,global_jacob,global_jacob_t,njaco,
&global_sigc_t,global_sigc,phase_id
        use globalsubs, only: convert3x3to6,invert3x3,skew,misorientation,
&polar
        implicit none
c      Inputs
        integer ncycle,el_no,ip_no,inc,subinc,i,j,is
        real(8) F_t(3,3),F(3,3),dt
c      Outputs
        real(8) stress(6),jacob(6,6)
c      Variables used within this subroutine
        real(8) Fp_t(3,3),Fe_t(3,3),rhoSSD_t(24)
        real(8) Fp(3,3),rhoSSD(24),Cauchy(3,3)
        real(8) invFp(3,3),PK2_vec(6),gammadot(24),dgammadot_dtau(24)
        real(8) Lp(3,3),Fe(3,3),rhoM(24),detFp
        integer innoit,ounoit,notconv
        real(8) gammadot_av,rhoSSD_av
        integer iloop, oloop
        real(8) sigc_t, sigc, epsdot

c      If the FE increment is updated (FE accepts the solution
c      and moves to a different increment then do the assignment)
        if (inc_old.ne.inc) then
            write(6,*) 'Updated the state variables'
            inc_old=inc
            global_Fp_t=global_Fp
            global_Fe_t=global_Fe
            global_rhoSSD_t=global_rhoSSD
            global_jacob_t=global_jacob

```

```

        global_sigc_t = global_sigc
        subinc_old=0
    end if
c
c
c    If there is a sub increment
c    new subincrement? update starting with subinc 2
    if(subinc.gt.subinc_old) then
        write(6,*) 'Updated the state variables'
        write(6,*) 'Sub-incrementation!!!'
        global_Fp_t=global_Fp
        global_Fe_t=global_Fe
        global_rhoSSD_t=global_rhoSSD
        global_jacob_t=global_jacob
        global_sigc_t = global_sigc
        subinc_old=subinc
    end if

c    Martensite calculations
    if (phase_id(el_no).eq.2.0) then

c
        write(6,*) 'MARTENSITE CALCULATIONS'

        Fp_t=global_Fp_t(el_no,ip_no,,:)
        Fe_t=global_Fe_t(el_no,ip_no,,:)
        sigc_t=global_sigc_t(el_no,ip_no)

c        Calculate stress and flow stress using J2 plasticity
        call MAR_main(dt,F,Fp_t,Fe_t,sigc_t,
& Fp,Fe,epsdot,sigc,stress,iloop,oloop)

        if (iloop.ge.innoitmax) then
            write(6,*) 'Inner loop during stress calculation
& did not converge!'
            write(6,*) 'el_no'
            write(6,*) el_no
            write(6,*) 'ip_no'
            write(6,*) ip_no
            write(6,*) 'inner no it'
            write(6,*) iloop
            write(6,*) 'outer no it'
            write(6,*) oloop
            write(6,*) 'F'
            write(6,*) F
            write(6,*) 'F_t'
            write(6,*) F_t
            write(6,*) 'Fp_t'
            write(6,*) Fp_t
            write(6,*) 'sigc_t'
            write(6,*) sigc_t
            write(6,*) 'sigc'
            write(6,*) sigc
            write(6,*) 'stress'

```

```

        write(6,*) stress
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
        call quit(9101)
    endif
c
    if (oloop.ge.ounoitmax) then
        write(6,*) 'Outer loop during stress calculation
& did not converge!'
        write(6,*) 'el_no'
        write(6,*) el_no
        write(6,*) 'ip_no'
        write(6,*) ip_no
        write(6,*) 'inner no it'
        write(6,*) iloop
        write(6,*) 'outer no it'
        write(6,*) oloop
        write(6,*) 'F'
        write(6,*) F
        write(6,*) 'F_t'
        write(6,*) F_t
        write(6,*) 'Fp_t'
        write(6,*) Fp_t
        write(6,*) 'sigc_t'
        write(6,*) sigc_t
        write(6,*) 'sigc'
        write(6,*) sigc
        write(6,*) 'stress'
        write(6,*) stress
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
        call quit(9102)
    endif

c      Update the state variables
        global_Fp(el_no,ip_no,:)=Fp
        global_Fe(el_no,ip_no,:)=Fe
        global_sigc(el_no,ip_no)=sigc

c      JACOBIAN CALCULATION FOR MARTENSITE
c      For a number of increments do the jacobian calculation
c      Note, jacobian is needed at the first calculation
c      if (modulo(inc,njaco).eq.0) then
c          Calculate the material tangent (using perturbation)

                call MAR_jacobian(dt,F_t,F,Fe_t,Fp_t,sigc_t,
& stress,jacob,notconv)

c          jacob=zeta66

c          When the jacobian calculation did not converge, assign the old
jacobian!
                if (notconv.eq.1) then
c                    Jacobian
                    jacob=global_jacob_t(el_no,ip_no,:,:)
                    return
                endif

```

```

else
c      Note this also works when inc=1 since it is elasticity matrix
      jacob=global_jacob_t(el_no,ip_no,,:)

endif

c      Store the important variables

      global_sigc(el_no,ip_no) = sigc

      global_gammadot_av(el_no,ip_no) = epsdot*3.1
      global_rhoSSD_av(el_no,ip_no) = 0.0

      global_mis(el_no,ip_no) = 0.0

c      Single crystal calculations
      else

c      write(6,*) 'SINGLE CRYSTAL CALCULATIONS'

      Fp_t=global_Fp_t(el_no,ip_no,,:)
      Fe_t=global_Fe_t(el_no,ip_no,,:)
      rhoSSD_t=global_rhoSSD_t(el_no,ip_no,:)

      call SC_main(F,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,dt,
&invFp,PK2_vec,gammadot,dgammadot_dtau,Lp,rhoSSD,
&rhoM,innoit,ounoit,Fe,Fp,stress,gammadot_av,rhoSSD_av)

c
c
      if (innoit.ge.innoitmax) then
&      write(6,*) 'Inner loop during stress calculation
&      did not converge!'
      write(6,*) 'el_no'
      write(6,*) el_no
      write(6,*) 'ip_no'
      write(6,*) ip_no
      write(6,*) 'inner no it'
      write(6,*) innoit
      write(6,*) 'outer no it'
      write(6,*) ounoit
      write(6,*) 'F'
      write(6,*) F
      write(6,*) 'F_t'

```

```

        write(6,*) F_t
        write(6,*) 'Fp'
        write(6,*) Fp
        write(6,*) 'Fp_t'
        write(6,*) Fp_t
        write(6,*) 'rhoSSD_t'
        write(6,*) rhoSSD_t
        write(6,*) 'rhoSSD'
        write(6,*) rhoSSD
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
      call quit(9101)
endif

c
      if (ounoit.ge.ounoitmax) then
        write(6,*) 'Outer loop during stress calculation
&      did not converge!'
        write(6,*) 'el_no'
        write(6,*) el_no
        write(6,*) 'ip_no'
        write(6,*) ip_no
        write(6,*) 'inner no it'
        write(6,*) innoit
        write(6,*) 'outer no it'
        write(6,*) ounoit
        write(6,*) 'F'
        write(6,*) F
        write(6,*) 'F_t'
        write(6,*) F_t
        write(6,*) 'Fp'
        write(6,*) Fp
        write(6,*) 'Fp_t'
        write(6,*) Fp_t
        write(6,*) 'rhoSSD_t'
        write(6,*) rhoSSD_t
        write(6,*) 'rhoSSD'
        write(6,*) rhoSSD
        write(6,*) 'dt'
        write(6,*) dt
c      QUIT MARC
      call quit(9102)
endif

c      Update the state variables
      global_Fp(el_no,ip_no,:)=Fp
      global_Fe(el_no,ip_no,:)=Fe
      global_rhoSSD(el_no,ip_no,:)=rhoSSD

c      write(6,*) 'innoit', innoit
c      write(6,*) 'ounoit', ounoit

      global_rhoSSD_av(el_no,ip_no)=rhoSSD_av

      global_gammadot_av(el_no,ip_no)=gammadot_av

      global_mis(el_no,ip_no) = 0.0

```

```

c      For a number of increments do the jacobian calculation
c      Note, jacobian is needed at the first calculation
c      if (modulo(inc,njaco).eq.0) then
c          Calculate the material tangent (using perturbation)
c          call SC_jacobian(F,F_t,Fe_t,Fp_t,rhoSSD_t,
&dt,el_no,ip_no,stress,jacob,notconv)
c          When the jacobian calculation did not converge, assign the old
jacobian!
c          if (notconv.eq.1) then
c              write(6,*) 'el_no', el_no
c              write(6,*) 'ip_no', ip_no
c              write(6,*) 'Jacobian has not converged!!!'
c              Jacobian
c              jacob=global_jacob_t(el_no,ip_no,:,:)
c              return
c          endif
c      else
c          Note this also works when inc=1 since it is elasticity matrix
c          jacob=global_jacob_t(el_no,ip_no,:,:)
c      endif

```

```

c      Store the important results
c      global_stress(el_no,ip_no,:)=stress

c      Assign the value of jacobian even if there is no convergence
c      global_jacob(el_no,ip_no,:)=jacob

c
c
c
c      Calculate the outputs
cc      Misorientation with respect to the initial orientation
cc      Initial orientation
c      g1=global_ori(el_no,ip_no,:,:)
cc      Polar decompose the elastic part of the deformation gradient
c      call polar(Fe_T,R,U,sing)
cc      If singularity does no exist
cc      otherwise old solution will remain.
c      if (sing.eq.0) then
cc          Calcualte misorientation using the subroutine
c          call misorientation(g1,R,1,ax,ang,dg)
cc          Store the misorientation value
c          global_mis(el_no,ip_no)=ang
c      endif

c      write(6,*) 'el_no'

```

```

c      write(6,*) el_no
c      write(6,*) 'ip_no'
c      write(6,*) ip_no
c      write(6,*) 'stress'
c      write(6,*) stress
c      write(6,*) 'jacob'
c      write(6,*) jacob

      return
end subroutine

c
c
c

c
c
c      Main routine for jacobian calculation for Martensite
c      This subroutine calculates consistent tangent
c      subroutine MAR_jacobian(dt,F_t,F,Fe_t,Fp_t,sigc_t,
&stress,jacob,notconv)
      use globalvars, only : deps,innoitmax,ounoitmax
      use globalsubs, only : convert3x3to6,convert6to3x3
      implicit none
c      Inputs
real(8) F_t(3,3),F(3,3),dt
real(8) Fp_t(3,3),Fe_t(3,3),sigc_t,stress(6)
c      Outputs
real(8) jacob(6,6)
integer notconv
c      Variables used within this subroutine
real(8) Fp(3,3),Fe(3,3),epsdot,sigc
real(8) Fper(3,3),dFrel_vec(6),dFrel(3,3),Cauchy_per_vec(6)
integer i,iloop,oloop

c
c      Assign the convergent behavior
notconv=0

cc      Cauchy stress
c      call convert6to3x3(Cauchy_t_vec,Cauchy_t)
c
c      Increment 6 components of relative deformation gradient
do i=1,6
c          Component-wise perturbation
dFrel_vec=0.0
dFrel_vec(i)=deps

c          if (i.lt.4) dFrel_vec(i)=deps
cc          Note it is not deps/2 since during conversion only one component is
considered
c          if (i.ge.4) dFrel_vec(i)=deps/2.0
c          Convert the vector to a matrix
call convert6to3x3(dFrel_vec,dFrel)
Fper=F+matmul(dFrel,F_t)
c          Call the calculation procedure
call MAR_main(dt,Fper,Fp_t,Fe_t,sigc_t,
& Fp,Fe,epsdot,sigc,Cauchy_per_vec,iloop,oloop)

      jacob(1:6,i)=(Cauchy_per_vec-stress)/deps
enddo

```

```

c      Make it symmetric
      jacob=(transpose(jacob)+jacob)/2.0

      return
    end subroutine

c
c

c
c      Main routine for J2-plasticity
      subroutine MAR_main(dt,F,Fp_t,Fe_t,sigc_t,
& Fp,Fe,epsdot,sigc,stress,iloop,oloop)
      use globalsubs, only: invert3x3, convert3x3to6, trace, normmat,
& convert6to3x3
      use globalvars, only: epsdot0_mart,mm_mart,h0_mart,ss_mart,a_mart,
& zeta66_mart,innertol,outertol,innoitmax,ounoitmax,deps,factor,I3,
& G_mart
      implicit none
c      Inputs
      real(8) F(3,3),dt
      real(8) Fp_t(3,3),Fe_t(3,3),sigc_t
c      Outputs
      real(8) Fp(3,3),Fe(3,3),epsdot,sigc,stress(6)
      integer iloop,oloop
c      Vairables used
      real(8) invFp_t(3,3),GLS(3,3),GLSvec(6),det
      real(8) GLStr(3,3),GLStrvec(6)
      real(8) Cauchybarvec(6),Cauchybar(3,3),Cauchybardev(3,3)
      real(8) hyd,sigmabar,Cauchy_t(3,3),Cauchy(3,3)
      real(8) Np_bar(3,3),Cauchy_dev(3,3),Cauchy_vec(6),sigma,yf
      real(8) depsdot_dsigma
      real(8) R1,xx1,hb,dsigc,R2,xx2,Dp(3,3),invFp(3,3)
      real(8) xx1_old,sigma_old

c
      call invert3x3(Fp_t,invFp_t,det)
c      Trial elastic deformation
      Fe=matmul(F,invFp_t)

c      Trial elastic stretch
      GLStr=(matmul(transpose(Fe),Fe)-I3)/2.0

c      Vectorize strain
      call convert3x3to6(GLStr,GLStrvec)

c      Vectorized trial stress
      Cauchybarvec=matmul(zeta66_mart,GLStrvec)

c      Trial stress
      call convert6to3x3(Cauchybarvec,Cauchybar)
c      Hydrostatic component of the trial stress
      call trace(Cauchybar,3,hyd)
      hyd=hyd/3.0

```

```

c      Deviatoric component of the trial stress
      Cauchybardev=Cauchybar-(I3*hyd)
c      Equivalent trial stress
      call normmat(Cauchybardev,3,sigambar)
      sigambar=dsqrt(3.0d+0/2.0d+0)*sigambar
c      Plastic flow direction (if statement is to correct sigambar=0 situation
which give rise to 1/0 problem)
      if (sigambar.ne.0) then
          Np_bar=3.0/2.0*Cauchybardev/sigambar
      else
          Np_bar=0.0
      endif
c      Assign the initial value for the parameters before starting the loop
c      Initial guess for the shear resistance
      sigc=sigc_t

c      COMPUTE FORMER STRESS
c      Former elastic stretch
      GLS=(matmul(transpose(Fe_t),Fe_t)-I3)/2.0
c      Vectorize strain
      call convert3x3to6(GLS,GLSvec)
c      Vectorized former stress
      Cauchy_vec=matmul(zeta66_mart,GLSvec)
c      3x3 former stress tensor
c      Initial guess for the stress
      call convert6to3x3(Cauchy_vec,Cauchy)

c      Deviatoric stress
      call trace(Cauchy,3,hyd)
      hyd=hyd/3.0
      Cauchy_dev=Cauchy-(I3*hyd)
c      Equivalent stress
      call normmat(Cauchy_dev,3,sigma)

      sigma=dsqrt(3.0d+0/2.0d+0)*sigma

c      OUTER loop starts here
      do oloop=1,ounoitmax
c          INNER loop starts here
c          Assign a very large value
          xx1_old=1.0d+10
          do iloop=1,innoitmax
c              Plastic strain rate
              epsdot=epsdot0_mart*(sigma/sigc)**(1.0/mm_mart)

c              if (gammadot.gt.1.0) then
c                  write(6,*) '*****'
c                  write(6,*) 'Shear rates flying!'
c                  write(6,*) '*****'
c                  call quit(9100)
c              endif
c              Derivative of shear rate with respect to the stress
              depsdot_dsigma=epsdot0_mart/mm_mart/sigc*
&                                     (sigma/sigc)**((1.0/mm_mart)-1.0)

c              Residual
              R1=sigma-sigambar+(dt*3.0*G_mart*epsdot)

```

```

c          Relative norm of the residual
c          if (sigmabar.eq.0) then
c              xx1=0.0
c          else
c              xx1=dabs(R1)/sigmabar
c          endif
c          Check the residual
c          if residual is smaller than the tolerance EXIT
c          if (xx1.lt.innertol) then
c              exit
c          if the norm tends to increase and it is not the first
increment
c              elseif ((xx1.gt.xx1_old).and.(iloop.gt.1)) then
c                  write(6,*) 'modified initial guess'
c                  modify the initial guess
c                  sigma=((1.0-factor)*sigma)+(factor*sigma_old)
c                  Assign a very large value
c                  xx1_old=1.0d+10
c          if the residual have a converging behavior
c          elseif (xx1.lt.xx1_old) then
c              Calculate tangent for N-R iteration
c              yf=1.0+(dt*3.0*G_mart*depsdot_dsigma)
c              Store the old value of stress to update the guess
c              sigma_old=sigma
c              Stress after the iteration
c              sigma=sigma-R1/yf
c              Assign the old value of the norm for the next
iteration
c              xx1_old=xx1
c          endif
c          enddo

c          End of INNER loop
c          Hardening parameters
c          hb=h0_mart*(1.0-(sigc/ss_mart))*a_mart
c          Increment in shear resistance
c          dsigc=hb*epsdot*dt
c          Residual in the shear resistance
c          R2=sigc-dsigc-sigc_t

c          Relative tolerance
c          xx2=dabs(R2/sigc_t)
c          Check the tolerance
c          if (xx2.lt.outertol) exit
c          Update the shear resistance
c          sigc=sigc_t+dsigc
c          enddo

c          Plastic stretch tensor
c          Dp=epsdot*Np_bar
c          Plastic part of the deformation gradient
c          Fp=matmul(((Dp*dt)+I3),Fp_t)
c          call invert3x3(Fp,invFp,det)
c          Elastic part of the deformation gradient
c          Fe=matmul(F,invFp)

c          Elastic stretch
c          GLS=(matmul(transpose(Fe),Fe)-I3)/2.0d+0

```

```

c      Vectorize strain
c      call convert3x3to6(GLS, GLSvec)

c      Vectorized stress
c      stress=matmul(zeta66_mart, GLSvec)

      return
end subroutine

c
c
c
c
c

c
c      This subroutine calculates consistent tangent
c      The effect of non-local deformations at the former time step is included
only!
      subroutine SC_jacobian(F, F_t, Fe_t, Fp_t,
&rhoSSD_t, dt, el_no, ip_no, Cauchy_vec,
&jacob, notconv)
      use globalvars, only : deps, innoitmax, ounoitmax, global_jacob
      use globalsubs, only : convert6to3x3
      implicit none

c      Inputs
real(8) F_t(3,3), F(3,3), Fe_t(3,3), dt
real(8) Fp_t(3,3), rhoSSD_t(24), Cauchy(3,3)
c      Outputs
real(8) jacob(6,6)
integer notconv, el_no, ip_no
c      Variables used within this subroutine
real(8) Cauchy_per(3,3)
real(8) F_per(3,3), dFrel_vec(6), dFrel(3,3), Cauchy_per_vec(6)
real(8) Cauchy_vec(6), PK2_per_vec(6)
real(8) invFp_per(3,3), dummy2(24), dummy3(24), dummy4(3,3)
real(8) dummy5(24), dummy6(24), dummy7(24)
real(8) dummy8(3,3), dummy9(3,3), dummy10, dummy11
integer i, j, innoit, ounoit
c      Assign the convergent behavior
notconv=0
jacob=0.0

cc      Calculate inverse of the deformation gradient
c      call invert3x3(F, invF, detF)
c
cc      Calculate velocity gradient
c      L = (I3 - matmul(F_t, invF))/dt
c
c      dum = I3 - L*dt_per
c      call invert3x3(dum, dum, dummy10)

```

```

c
cc      Calculate the FUTURE defomation gradient
c      F_per_new = matmul(dum,F)
c
c      do i=1,3
c          write(6,*) (F_per_new(i,j),j=1,3)
c      enddo

      do i=1,6

c          Component-wise pertubation
c          dFrel_vec=0.0
c          dFrel_vec(i)=deps
!c          dFrel_vec(i)=deps
!          if (i.le.3) then
!              dFrel_vec(i)=deps
!c          Note it is not deps/2 since during conversion only one component is
considered
!          else
!              dFrel_vec(i)=deps/2.0
!          endif

c          Convert the vector to a matrix
c          call convert6to3x3(dFrel_vec,dFrel)

c          F_per=F+matmul(dFrel,F_t)

!          write(6,*) 'F_per'
!do k=1,3
!          write(6,*) (F_per(k,j),j=1,3)
!enddo
!
!write(6,*) 'F'
!do k=1,3
!          write(6,*) (F(k,j),j=1,3)
!enddo

c          Call the calculation procedure
c          call SC_main(F_per,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,
&dt,invFp_per,PK2_per_vec,dummy2,dummy3,dummy4,dummy5,
&dummy7,innoit,ounoit,dummy8,dummy9,Cauchy_per_vec,dummy10,dummy11)

c
!          write(6,*) 'Cauchy_per_vec'
!          do j=1,6
!              write(6,*) Cauchy_per_vec(j)
!          enddo
!
!          write(6,*) 'Cauchy_vec'
!          do j=1,6
!              write(6,*) Cauchy_vec(j)
!          enddo
!c
c          if ((innoit.ge.innoitmax).or.(ounoit.ge.ounoitmax)) then
c              notconv=1
c              return
c          endif

```

```

c
!c      Calculate the elastic part of the deformation
!      Fe_per=matmul(F_per,invFp_per)
!
!c      Calculate Cauchy stress
!      call cauchystress(PK2_per_vec,Fe_per,Cauchy_per,Cauchy_per_vec)

c      Assignment of jacobian components

      jacob(:,i)=(Cauchy_per_vec-Cauchy_vec)/deps

enddo

c      Make it symmetric
      jacob=(transpose(jacob)+jacob)/2.0

c      jacob=zeta66

      return
end subroutine

c
c
c      main subroutine that calculates the stress and correct Fp through
c      Dawson-Maniatty intergration method
c      This contains all the sub-routines called during the calculation
c      INPUTS: F(3,3), Fe_t(3,3), Fp_t(3,3), tauc_t(24), dt
c      OUPUTS: invFp(3,3), PK2_vec(6), gammadot(24), dgammadot_dtau(24),
c              Lp(3,3), tauc(24), initno, outno
c      USES: scale, innertol, outertol, innoitmax, ounoitmax
c      subroutine SC_main(F,Fe_t,Fp_t,rhoSSD_t,el_no,ip_no,dt,invFp,PK2_vec,
&gammadot,dgammadot_dtau,Lp,rhoSSD,rhoM,innoit,ounoit,Fe,Fp,stress,
&gammadot_av,rhoSSD_av)
c      use globalvars, only : ounoitmax, innoitmax, innertol, outertol,
&factor,rhoSSD0
c      use globalsubs, only : invert3x3
c      implicit none
c      Input variable declarations
c      real(8) F(3,3),Fe_t(3,3),Fp_t(3,3)
c      real(8) rhoSSD_t(24),dt
c      integer el_no,ip_no
c      Output variable declarations
c      real(8) invFp(3,3),PK2_vec(6),gammadot(24),dgammadot_dtau(24)
c      real(8) Lp(3,3),rhoSSD(24),rhoM(24)
c      integer ounoit,innoit
c      Variables used within this subroutine
c      integer i,j,is
c      real(8) detFp,invFe_t(3,3),Fp(3,3),C(3,3),Ce(3,3),taueff(24)
c      real(8) normR,R_vec(9),inres,inres0,Fp_temp(3,3),detFe,sigma(24)

```

```

    real(8) dFp(3,3),rhoSSD_dot(24)
    real(8) R_rhoSSD(24),outres
    real(8) taupass(24),Vol(24),vel_con(24),mob_con(24),rho_f(24)
    real(8) Fe(3,3),stress(6),Cauchy(3,3),gammadot_av,rhoSSD_av
c
c
c    Initial fully-plastic guess
    call invert3x3(Fe_t,invFe_t,detFe)
    Fp=matmul(invFe_t,F)
    Fp_temp=Fp_t
c    Overall Right-Cauchy-Green tensor
    C=matmul(transpose(F),F)
c    Assign initial variables
    rhoSSD=rhoSSD_t
c    write(6,*) 'rhoSSD', rhoSSD

c
c    Outer loop starts HERE!
    do ounoit=1,ounoitmax
c        Assign a high value for the residual (Necessary for scale-back
algorithm)
        inres=1.0d+10
c
c        Calculate parameters that depend on the state variables at the
outer level of iterations
        call statedeptconsts(rhoSSD,taupass,Vol,vel_con, mob_con,rho_f)

c        write(6,*) 'rho_f', rho_f
c        Inner loop starts HERE!
        do innoit=1,innoitmax
c            Inverse of the plastic deformation gradient
            call invert3x3(Fp,invFp,detFp)
c
c            Constitutive calculations
            write(6,*) ',innoit',innoit
c
c            write(6,*) 'PK2_vec', PK2_vec
            call
constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,ip_no,
&Ce,PK2_vec,taueff,sigma,gammadot,dgammadot_dtau,Lp,rhoM)
c
c            write(3,*) 'tau(1)',tau(1)
c            write(3,*) 'C'
c            do i=1,3
c                write(3,*) (C(i,j),j=1,3)
c            enddo
c            write(3,*) 'invFp'
c            do i=1,3
c                write(3,*) (C(i,j),j=1,3)
c            enddo
c            write(6,*) 'taupass',taupass
c            write(6,*) 'tau',tau
c            write(6,*) 'gammadot',gammadot
c            write(3,*) 'rhoM(1)',rhoM(1)
c            write(3,*) 'velcon',velcon
c            write(3,*) 'Vol',Vol
c            write(3,*) 'dgammadot_dtau(1)',dgammadot_dtau(1)
c            Residual calculation
            call residual(invFp,Fp_t,Lp,detFp,dt,R_vec,normR)
c            write(3,*) 'R_vec'

```

```

C          write(3,*) R_vec
C          write(6,*) 'normR',normR
C          Assign old value of residual
C          inres0=inres
C          inres=normR

C
C          If the residual is smaller than the tolerance exit the loop
C          if (normR.lt.innertol) then
C              write(3,*) innoit
C              write(3,*) 'converged'
C              exit
C          If the tolerance is increasing scale back the initial guess

C          elseif ((normR.gt.innertol).and.(inres.gt.inres0)) then
C              write(3,*) innoit
C              write(3,*) 'scaled-back the initial guess'
C              write(3,*) 'Fp',Fp
C              write(3,*) 'Fp_temp',Fp_temp
C              Fp=((1.0-factor)*Fp)+(factor*Fp_temp)
C              inres=1.0d+10
C          else otherwise calculate the tangent for Newton-Raphson
scheme
C          else
C              write(3,*) innoit
C              call tangent(R_vec,Ce,invFp,detFp,
&                  Fp_t,dgammadot_dtau,el_no,ip_no,dt,dFp)
C
C              write(3,*) 'invFp'
C              do i=1,3
C                  write(3,*) (invFp(i,j),j=1,3)
C              enddo
C              write(3,*) 'dFp'
C              do i=1,3
C                  write(6,*) (dFp(i,j),j=1,3)
C              enddo
C
C              write(3,*) 'iterate with tangent'
C              Store the current Fp as old Fp since it may be used
in scaling-back algorithm
C              Fp_temp=Fp
C              The new deformation gradient after NR iteration
C              Fp=Fp+dFp
C              write(3,*) 'dFp'
C              write(3,*) dFp
C          endif
C      enddo
C      write(3,*) 'innoit: ',innoit
C      END of INNER iteration loop
C      Calculate amount of slip system hardening
&      call hardening(gammadot,taueff,sigma,rhoSSD,rhoM,taupass,
&      rho_f,rhoSSD_dot)

C      if (ip_no.eq.8) then
C          write(6,*) 'rhoSSD_dot(1)',rhoSSD_dot(1)
C          write(6,*) 'gammadot(1)',gammadot(1)
C      endif

C
C      Residual increments of slip resistance
C      R_rhoSSD=dabs(rhoSSD-rhoSSD_t-(rhoSSD_dot*dt))/minval(rhoSSD)

```

```

c          Relative maximum change in slip system resistivity
          outres=maxval(R_rhoSSD)

c          write(6,*) '*****'
c          write(6,*) 'R_rhoSSD',R_rhoSSD
c          write(6,*) 'outres', outres
c          write(6,*) '*****'

c          Check if the change of hardening is within the tolerances
          if(outres.lt.outertol)then
            exit
          endif

c          If not accept the result and continue the implicit iteration
          rhoSSD=rhoSSD_t+(rhoSSD_dot*dt)

        enddo
c      END of OUTER iteration loop
c      write(3,*) 'ounoit: ',ounoit
c
c
c
c
c      call statedeptconsts(rhoSSD,taupass,Vol,vel_con, mob_con,rho_f)

c      call constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,ip_no,
&Ce,PK2_vec,taueff,sigma,gammadot,dgammadot_dtau,Lp,rhoM)

c      Calculate the elastic part of the deformation
      Fe=matmul(F,invFp)

c      Calculate Cauchy stress
      call cauchystress(PK2_vec,Fe,Cauchy,stress)

c      Average of the shear rates
      gammadot_av=0.0
      do is=1,24
        gammadot_av=gammadot_av+dabs(gammadot(is))
      enddo
      gammadot_av=gammadot_av/24.0

c      Average of the SSD densities
      rhoSSD_av=0.0
      do is=1,24
        rhoSSD_av=rhoSSD_av+rhoSSD(is)
      enddo
      rhoSSD_av = rhoSSD_av/24.0

      return
    end subroutine

c
c
c
c
c

```

```

c      This subroutine includes the constitutive calculations
c      INPUTS: C(3,3), invFp(3,3), tauc(24)
c      OUTPUTS: Ce(3,3), PK2_vec(6), tau(24), gammadot(24), dgammatot_dtau(24),
Lp(3,3)
c      USES: Schmid(24,3,3), Schmid_vec(24,6), zeta66(6,6), gammadot0, mm,
I3(3,3)
      subroutine constitutive(C,invFp,taupass,vel_con,mob_con,Vol,el_no,
&ip_no,Ce,PK2_vec,taueff,sigma,gammadot,dgammatot_dtau,Lp,rhoM)
      use globalvars, only : I3,zeta66,Schmid,KBT,
&Schmid_vec,Climb_vec,Burger,B0,rhoSSD0,NonSchmid1_vec,
&NonSchmid2_vec,NonSchmid3_vec,a1,a2,a3,BCF
      use globalsubs, only : convert3x3to6
      implicit none
c      Input variable declarations
      real(8) C(3,3),invFp(3,3)
      integer el_no,ip_no
      real(8) taupass(24),vel_con(24),mob_con(24),Vol(24)
c      Output variable declarations
      real(8) Ce(3,3),PK2_vec(6),taueff(24),sigma(24),rhoM(24)
      real(8) gammadot(24),dgammatot_dtau(24),Lp(3,3)
c      Variables used within the code
      real(8) E(3,3),E_vec(6),PK2(3,3)
      real(8) vel,dvel_dtau,velth,dvelth_dtau,veldr,dveldr_dtau
      real(8) drhoM_dtau,tau(24)
      real(8) tau_ns(24),tau_ns1(24),tau_ns2(24),tau_ns3(24)
      integer xx,i

c
c
c
c      open(unit=3,file="output.txt")

c      Elastic-Right Cauchy-Green tensor
      Ce=matmul(transpose(invFp),matmul(C,invFp))

c
c      Green-Lagrange strain tensor
      E=(Ce-I3)/2.0d+0

c
c      Vectorized strain (Notation as Reid, Def. Geo.)
      call convert3x3to6(E,E_vec)

c
c
c      Vectorized 2nd Piola-Kirchoff stress tensor through Elastic constitutive
law
      PK2_vec=matmul(zeta66,E_vec)

c
c      Calculation of plastic part of the velocity gradient
      Lp=0.0d+0
      taueff=0.0d+0
      tau=0.0d+0
      tau_ns1=0.0d+0
      tau_ns2=0.0d+0
      tau_ns3=0.0d+0

      do xx=1,24

c          Calculate resolved shear stress
          do i=1,6
              tau(xx)=tau(xx)+BCF*Schmid_vec(xx,i)*PK2_vec(i)
              tau_ns1(xx)=tau_ns1(xx)+a1
&              *BCF*NonSchmid1_vec(xx,i)*PK2_vec(i)
              tau_ns2(xx)=tau_ns2(xx)+a2

```

```

&      *BCF*NonSchmid2_vec(xx,i)*PK2_vec(i)
tau_ns3(xx)=tau_ns3(xx)+a3
&      *BCF*NonSchmid3_vec(xx,i)*PK2_vec(i)

enddo

tau_ns(xx)=tau_ns1(xx)+tau_ns2(xx)+tau_ns3(xx)

c      Calculate climb stress
sigma(xx)=0.0d+0
do i=1,6
    sigma(xx)=sigma(xx)+(Climb_vec(xx,i)*PK2_vec(i))
enddo

c      Effective stress
taueff(xx)=tau(xx)+tau_ns(xx)-taupass(xx)

c      If the applied stress is greater than the passing stress
if ((taueff(xx).gt.0d+0).and.(tau(xx).gt.0d+0)) then
c
c      write(6,*)'Hello Im here!'
c
c      write(6,*)'tau_ns',tau_ns(xx)
c      write(6,*)'tau',tau(xx)
c
c      Thermally activated slip velocity
velth=vel_con(xx)*dexp(taueff(xx)*Vol(xx)/KBT)
c
c      write(6,*)'velth',velth
c
c      and its derivative wrto shear stress
dvelth_dtau=vel_con(xx)*dexp(taueff(xx)*Vol(xx)/KBT)*Vol(xx)/KBT

c      Mobile dislocation
rhoM(xx)=mob_con(xx)*taueff(xx)**2.0d+0
c      Derivative of mobile dislocation density wrto shear stress
drhoM_dtau=2.0d+0*mob_con(xx)*taueff(xx)

c      Drag velocity
velldr = taueff(xx)*Burger/B0
c      Derivative of drag velocity wrto shear stress
dvelldr_dtau = Burger/B0

c      Combined velocity - geometric average
vel = 1.0/((1.0/velldr) + (1.0/velth))
c      Derivative of the combined expression
dvel_dtau=(vel/velth)**2.0*dvelth_dtau
&      + (vel/velldr)**2.0*dvelldr_dtau

else
    vel=0.0d+0
    dvel_dtau=0.0d+0
    rhoM(xx)=0.0d+0
    drhoM_dtau=0.0d+0
endif

c      Shear rate
gammadot(xx)=rhoM(xx)*Burger*vel

c      Derivative of shear rate wrto shear stress

```

```

                dgammatot_dtau(xx)=rhoM(xx)*Burger*dvel_dtau +
drhoM_dtau*Burger*vel

c                Plastic velocity gradient
                Lp=Lp+(BCF*Schmid(xx, :, :)*gammatot(xx))
        enddo
c    write(3,*) gammatot(1)
c    write(3,*) 'velpar(1)',velpar(1)
c    write(3,*) 'tau(1)',tau(1)
c    write(3,*) 'taupass(1)',taupass(1)
c    write(3,*) 'vel(1)',vel(1)
c    write(3,*) 'rhoM(1)',rhoM(1)
c
c    close(3)
c
c    return
end subroutine

c
c
c
c
c
c
c    This subroutine calculate the residual
c    INPUTS:          invFp(3,3), Fp_t, detFp, Lp, dt
c    OUTPUTS:         R_vec, normR
c    USES:            I3, lambda_p
c    subroutine residual(invFp,Fp_t,Lp,detFp,dt,
&R_vec,normR)
c    use globalvars, only: I3,lambda_p
c    use globalsubs, only: normvec,convert3x3to9
c    implicit none
c    Input variable declarations
c    real(8) invFp(3,3),Fp_t(3,3),Lp(3,3),detFp,dt
c    Output variable declarations
c    real(8) normR,R_vec(9)
c    Variables used within this subroutine
c    real(8) R1(3,3),R2(3,3),R3(3,3),R(3,3)
c    integer i,j
c
c
c
c    Lp from calculated shear rates
c    R1=Lp
c    Lp from fully-implicit integration using plastic part of the deformation
gradient
c    R2=-(I3-matmul(Fp_t,invFp))/dt
c    Residual from the determinant
c    R3=lambda_p*(detFp-1.0d+0)*I3
c    Total residual
c    R=R1+R2+R3
c
c
c    Vectorize residual
c    call convert3x3to9(R,R_vec)
c    Norm of the residual
c    call normvec(R_vec,9,normR)
c
c    return
end subroutine

c
c

```

```

c
c
c
c      This subroutine calculates the tangent and the increment in plastic part
of the deformation gradient
c      for the Newton-Raphson scheme
c      INPUTS:          R_vec(9), Ce(3,3), invFp(3,3), detFp, Fp_t(3,3),
dgamma_dot_dtau(24), dt
c      OUTPUTS:         dFp(3,3)
c      USES:            Schmid(24,3,3), zeta3333(3,3,3,3), lambda_p, I3(3,3)
      subroutine tangent(R_vec,Ce,invFp,detFp,Fp_t,
&                                dgamma_dot_dtau,el_no,ip_no,dt,dFp)
      use globalvars, only: lambdap,I3,Schmid,zeta3333,
&NonSchmid1, NonSchmid2, NonSchmid3, a1, a2, a3,BCF
      use globalsubs, only: convert3x3x3x3to9x9,invertnxn,convert9to3x3
      implicit none
c      Input variable declarations
      real(8) R_vec(9),Ce(3,3),invFp(3,3),detFp,Fp_t(3,3)
      real(8) dgamma_dot_dtau(24), dt
      integer el_no,ip_no
c      Output variable declaration
      real(8) dFp(3,3)
c      Variables used within this subroutine
      real(8) D1(3,3,3,3),D2(3,3,3,3),D3(3,3,3,3),D(3,3,3,3),det
      real(8) sum1a(24,3,3),sum1b(24,3,3),const_D1
      real(8) sum2,D9x9(9,9),invD9x9(9,9),dFp_vec(9)
      integer i,j,k,l,m,n,p,r,s,xx
c
c
c
c
c      Calculation of 1st component of the 4th order tangent, D1
      sum1a=0.0d+0
      sum1b=0.0d+0
      do k=1,3
         do l=1,3
            do xx=1,24
               do m=1,3
                  do n=1,3
                     const_D1=BCF*Schmid(xx,m,n)+
&(a1*BCF*NonSchmid1(xx,m,n))+(a2*BCF*NonSchmid2(xx,m,n))+
&(a3*BCF*NonSchmid3(xx,m,n))
                     do p=1,3
                        do r=1,3
                           do s=1,3
c      Calculation of D1a
                           sum1a(xx,k,l)=sum1a(xx,k,l)+((-1.0/2.0)*dgamma_dot_dtau(xx)*
&((invFp(1,m)*Ce(k,p))+(Ce(m,k)*invFp(1,p))))*
&zeta3333(p,n,r,s)*(Ce(r,s)-I3(r,s))*const_D1
c      Calculation of D1b
                           sum1b(xx,k,l)=sum1b(xx,k,l)+((-1.0/2.0)*dgamma_dot_dtau(xx)*
&Ce(m,p)*zeta3333(p,n,r,s)*((invFp(1,r)*Ce(k,s))+
&(Ce(r,k)*invFp(1,s)))*const_D1
                           enddo
                        enddo
                     enddo
                  enddo
               enddo
            enddo
         enddo
      enddo
c

```

```

D1=0.0d+0
do i=1,3
  do j=1,3
    do k=1,3
      do l=1,3
        do xx=1,24
          D1(i,j,k,l)=D1(i,j,k,l)+
& (BCF*Schmid(xx,i,j)*(sum1a(xx,k,l)+sum1b(xx,k,l)))
          enddo
        enddo
      enddo
    enddo
  enddo
enddo

c
cc
c Calculation of 2nd and 3rd components of the 4th order tangent, D2 and D3
do i=1,3
  do j=1,3
    do k=1,3
      do l=1,3
        sum2=0.0
        do m=1,3
          c Calculation of D2
          sum2=sum2-
& (Fp_t(i,m)*invFp(m,k)*invFp(l,j)/dt)
          enddo
        D2(i,j,k,l)=sum2
        c Calculation of D3
        D3(i,j,k,l)=I3(i,j)*lambdap*detFp*invFp(l,k)
        enddo
      enddo
    enddo
  enddo
enddo
D=D1+D2+D3

c Convert the jacobian to 9x9 matrix
call convert3x3x3x3to9x9(D,D9x9)

c Invert the Jacobian
call invertnxn(D9x9,9,invD9x9,det)

c
c
c XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
c A check must be placed here to check if the jacobian is invertible or not
c XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
c
c Increment for the deformation gradient
dFp_vec=matmul(-invD9x9,R_vec)

c
c Convert the vector to 2nd order tensor
call convert9to3x3(dFp_vec,dFp)

c
c
c return
end subroutine

c
c
c This subroutine calculates the state variables before going into the inner
loop
c INPUTS:          gammadot(24), tauc(24)
c OUTPUTS:         dtauc(24)

```

```

c      USES:          h0, ss, a, hardmat
c      subroutine statedeptconsts(rhoSSD,taupass,Vol,vel_con,mob_con,rho_f)
c      use globalvars, only: G,Burger,c1,c2,c3,m1,m2,intmat,proj
c      use globalsubs, only : matmulvec
c      implicit none
c      Input variable declarations
c      real(8) rhoSSD(24)
c      Output variable declarations
c      real(8) taupass(24),Vol(24),vel_con(24),mob_con(24),rho_f(24)
c      Variables used within this subroutine
c      real(8) lambda
c      integer xx
c
c
c      Pass the dislocation density from the interaction matrix
c      rho_f = matmul(proj,rhoSSD)
c
c      write(6,*) 'rho_f',rho_f
c
c
c      do xx=1,24
c
c          Long-range stress
c          taupass(xx)=c1*G*Burger*dsqrt(rhoSSD(xx))
c          Jump distance
c          lambda=c2/dsqrt(rhoSSD(xx))
c          Activation volume
c          Vol(xx)=c3*(Burger**2.0)*lambda
c          Vol(xx)=c3*Burger**3.0
c
c          Thermally activated velocity multiplier
c          vel_con(xx)=m1*lambda
c          Mobile dislocation density multiplier
c          if (rho_f(xx).lt.0) then
c              write(6,*) '- forest'
c          endif
c
c          mob_con(xx)=m2*rhoSSD(xx)/rho_f(xx)
c      enddo
c
c      return
c      end subroutine
c
c
c
c
c
c
c      This subroutine calculates the amount of hardening for a given shear rate
c      INPUTS:          gammadot(24), tauc(24)
c      OUTPUTS:         dtauc(24)
c      USES:            h0, ss, a, hardmat
c      subroutine hardening(gammadot,taueff,sigma,rhoSSD,rhoM,
c      &taupass,rho_f,rhoSSDdot)
c      use globalvars, only: c4,m3,m4,m5,m6,Burger,Dv,acDc,Qcs,KBT,intmat
c      implicit none
c      Input variable declarations
c      real(8) gammadot(24),rhoSSD(24),rhoM(24)
c      real(8) taueff(24),sigma(24),taupass(24),rho_f(24)
c      Output variable declarations
c      real(8) rhoSSDdot(24)
c      Variables used within this subroutine
c      real(8) rhoSSDdot1(24),rhoSSDdot2(24),rhoSSDdot3(24)

```

```

real(8) rhoSSDdot4(24),g,P,Deff
integer xx

c
rhoSSDdot = 0.0d+0
rhoSSDdot1 = 0.0d+0
rhoSSDdot2 = 0.0d+0
rhoSSDdot3 = 0.0d+0
rhoSSDdot4 = 0.0d+0

do xx=1,24

c      Evolution rate of EDGE dislocaitons
c      Hardening due to mean free path
rhoSSDdot1(xx)=c4*dsqrt(rhoSSD(xx))/Burger*gammadot(xx)

c      Annihilation by climb
Deff = Dv + rhoSSD(xx)*acDc
rhoSSDdot3(xx)=-m4*Deff*(rhoSSD(xx)**2.0)*dabs(sigma(xx))*
&      gammadot(xx)

if (taueff(xx).gt.0d+0) then

c
c      Immobilization of mobile dislocations by dipole formation
rhoSSDdot2(xx)=m3*rhoM(xx)/taueff(xx)*gammadot(xx)

c
c      write(6,*) 'rhoSSDdot1', rhoSSDdot1
c      write(6,*) 'rhoSSDdot2', rhoSSDdot2
c      write(6,*) 'rhoSSDdot3', rhoSSDdot3
c
c      Evolution rate of SCREW dislocaitons
c      Annihilation by cross-slip
c      Activation energy for cross-slip
g = m6*taueff(xx)
c      Probability of cross-slip
P = dexp(-Qcs/KBT)*dsinh(g/KBT)
rhoSSDdot4(xx)=-m5*dsqrt(rhoSSD(xx))*P*gammadot(xx)
c      write(6,*) 'rhoSSDdot4', rhoSSDdot4

c
c

c      turn hardening off
rhoSSD_dot(xx) = 0.0d+0

endif

```



```

clear -except {rhoM_time1,rhoM_time2,rhoM_time3,eps_time,sig_time}
format long

% Maximum TENSILE strain
% Call Inputs
Inputs_SC;

no_inc=100;
%max_strain=0.1;
%rate=4.5e-3/(1/sqrt(6));
rate=3.3e-4;
tmax=150;
time=linspace(0,tmax,no_inc);

dt=tmax/no_inc;

% NOT SURE!!!

% Initial value for residual
% Old and new values of the residuals have to be stored for further

% % Initial stored variables
% Critical(1,1:12)=tauc_t;
% Stress(1,1:6)=zeros(6,1);
% ShearRates(1,1:12)=zeros(12,1);

% Assign the deformation gradient of the first increment
All_Fe_t=zeros(no_inc,3,3);
All_Fp_t=zeros(no_inc,3,3);
All_rho_M_t=zeros(no_inc,nslip);
All_rho_t=zeros(no_inc,nslip);
All_gammadot=zeros(no_inc,nslip);
All_SVM=zeros(1,no_inc);
All_EVM=zeros(1,no_inc);
All_Force=zeros(1,no_inc);

All_tau_max=zeros(no_inc,1);
All_gamma_max=zeros(no_inc,1);

% Assign the deformation gradient of the first increment
All_Fe_t(1,1:3,1:3)=Fe_t0;
All_Fp_t(1,1:3,1:3)=Fp_t0;
All_rho_t(1,1:nslip)=rho_0;

% Deformation increments
for inc=2:no_inc

    % % Iterate over a number of orientations
    % for ori=1:no_grain
    % Input deformation gradient
    F_T=[ exp(-0.5*rate*time(inc)) 0 0;
          0 exp(-0.5*rate*time(inc)) 0;
          0 0 exp(rate*time(inc))];

    F_t = [ exp(-0.5*rate*time(inc-1)) 0 0;
            0 exp(-0.5*rate*time(inc-1)) 0;
            0 0 exp(rate*time(inc-1))];

```

```

%           L_T=Sum*rate;
%           F_T=(eye(3,3)+L_T*dt)*F_t;

Fe_t(1:3,1:3)=All_Fe_t(inc-1,1:3,1:3);
Fp_t(1:3,1:3)=All_Fp_t(inc-1,1:3,1:3);
rho_t(1:nslip)=All_rho_t(inc-1,1:nslip);

% Main function in which the calculations are performed

[invFp_T,T_T_vec,gammadot,dgammadot_dtau,Lp,rho,rhoM,tau,inneriterno,o
uteriterno]...
    = main_func(F_T,Fe_t,Fp_t,rho_t,dt);

rhoM_time(inc,1:24)=rhoM;

% Elastic part of the deformation gradient
Fe_T=F_T*invFp_T;

% Plastic part of the deformation gradient
Fp_T=inv(invFp_T);

% Cauchy stress
[Cauchy]=CauchyStress(T_T_vec,Fe_T);

% Calculate force at the underformed area and deformed stress
state
% Stress state at the undeformed reference
S=inv(F_T)*Cauchy*inv(F_T)'+det(F_T);

tensile=F_T*tensile0;
tensile=tensile/norm(tensile);

[Force]=CalculateForce(Cauchy,tensile);

% Von-Mises Stress
[SVM] = VonMisesStress(Cauchy);

% Green-Lagrange Strain
E=0.5*(F_T*F_T'-I_3);

% Von-Mises Strain
[EVM] = VonMisesStrain(E);

% Assign new values
All_Fe_t(inc,1:3,1:3)=Fe_T;

```

```

All_Fp_t(inc,1:3,1:3)=Fp_T;
All_rho_t(inc,1:nslip)=rho;
All_rho_M_t(inc,1:nslip)=rhoM;
All_gammadot(inc,1:nslip)=gammadot;
All_SVM(inc)=SVM;
All_EVM(inc)=EVM;
All_Force(inc)=Force;

[max_val, max_ind]=max(abs(tau));
All_tau_max(inc) = max_val;
All_gamma_max(inc) = All_gamma_max(inc-1) +
abs(gammadot(max_ind)*dt);

disp([num2str(100*(inc)/no_inc) ' % complete!' ]);
disp('_____')

end

if counter==1
    rhoM_time1=rhoM_time;
elseif iii==2
    rhoM_time2=rhoM_time;
elseif iii==3
    rhoM_time3=rhoM_time;
end

% Mohammad Hasan Joudivand
% September 6, 2019
% Inputs to the program

% Global constants

global Schmid Climb C1
global NonSchmid_1 NonSchmid_2 NonSchmid_3
global a1 a2 a3
global M M1 M2 M3
global I_3
global Zeta_wc
global Zeta_3333
global lambda_p
global scale
global innoitmax
global innertol
global ounoitmax
global outertol
global order
global nslip
global Interaction Projection

global c1 c2 c3 c4 c5 c6 c7 c8
global Dv acDc
global m1 m2 m3 m4 m5 m6
global Qcs
global KT G Qpei Burger v0 B0

```

```

% global CS_Interaction

% % Global definitions used for the calculation of jacobian
% global I_9
% global P
% global dJ

% Identity matrices
%
I_3=eye(3,3);
%I_9=eye(9,9);
%

% 11-12-13-21-22-23-31-32-33
order(1,:)= [1 1];
order(2,:)= [1 2];
order(3,:)= [1 3];
order(4,:)= [2 1];
order(5,:)= [2 2];
order(6,:)= [2 3];
order(7,:)= [3 1];
order(8,:)= [3 2];
order(9,:)= [3 3];

%global h0
%global a
%global ss
%global ounoitmax
%global outertol

% Experimental error

% Error angle
th=0;
% Rotation about y-axis
Ry = [ cosd(th), 0, -sind(th);
       0, 1, 0;
       sind(th), 0, cosd(th)];

% Angular position
phi=0;
% Rotation about zaxis
Rz = [ cosd(phi), sind(phi), 0;
       -sind(phi), cosd(phi), 0;
       0, 0, 1];

R = Rz * Ry;

% Crystal Orientation & Schmid Tensor
%
% Entering the stereographic direction IDs
if counter==1
    iii=1;

```

```

elseif counter==2
    iii=2;
elseif counter==3
    iii=3;
end

if iii==1
    % Normal Direction
    hkl=[0 0 1];

    % Rolling Direction
    uvw=[1 0 0];
elseif iii==2
    % Normal Direction
    hkl=[0 1 1];

    % Rolling Direction
    uvw=[1 -1 1];
elseif iii==3
    % Normal Direction
    hkl=[-1 1 1];

    % Rolling Direction
    uvw=[1 1 0];
end

% Normalization of cutting direction and plane
nhkl=hkl/norm(hkl);
nuvw=uvw/norm(uvw);

% Vector Product
npqr = cross(nhkl,nuvw);
% Orientation Matrix: (hkl)[uvw]: Sample to Crystal
g=zeros(3);
g(:,1)=nuvw';
g(:,2)=npqr';
g(:,3)=nhkl';

g = R * g;

%oris(1,1:3,1:3)=g;

% Loading direction to calculate the force
tensile0=[0 0 1]';

% % Active slip systems
% %
% % 123 direction
% noAS=1;
% AS=8;
% ind=1;
% %
% % 100 direction

```

```

% noAS=8;
% AS=[1,2,4,5,7,8,10,11];
% ind=[-1,1,-1,1,-1,1,-1,1];
% %
% % 112 direction
% noAS=2;
% AS=[8 10];
% ind=[1 -1];
% %
% % 111 direction
% noAS=6;
% AS=[4,5,8,9,10,12];
% ind=[1,-1,1,-1,-1,1];

% % Euler Angles
% if g(3,3)==1
%     psi=0;
%     phil=atan2(g(1,2),g(1,1));
%     phi2=-phil;
% else
%     psi=acos(g(3,3));
%     st=sin(psi);
%     phi2=atan2(g(1,3)/st,g(2,3)/st);
%     phil=atan2(g(3,1)/st,-g(3,2)/st);
% end
% %
% % Orientation Matrix: Crystal to Sample
% r=g';

% Number of slip systems
nslip = 24;

% Slip systems
% C.R. Weinberger 2012

% Possible Slip Planes

n=[
% 1-12
    0, 1, -1;
    -1, 0, 1;
    1, -1, 0;

    -1, 0, -1;
    0, -1, 1;
    1, 1, 0;

    0, -1, -1;
    1, 0, 1;
    -1, 1, 0;

    1, 0, -1;
    0, 1, 1;
    -1, -1, 0;
% 13-24
    0, 1, -1;

```

```

-1, 0, 1;
1, -1, 0;

-1, 0, -1;
0, -1, 1;
1, 1, 0;

0, -1, -1;
1, 0, 1;
-1, 1, 0;

1, 0, -1;
0, 1, 1;
-1, -1, 0];

n=n';

% Possible Slip Directions
b=[
% 1-12
    1, 1, 1;
    1, 1, 1;
    1, 1, 1;

    -1, 1, 1;
    -1, 1, 1;
    -1, 1, 1;

    -1, -1, 1;
    -1, -1, 1;
    -1, -1, 1;

    1, -1, 1;
    1, -1, 1;
    1, -1, 1;
% 13-24
    -1, -1, -1;
    -1, -1, -1;
    -1, -1, -1;

    1, -1, -1;
    1, -1, -1;
    1, -1, -1;

    1, 1, -1;
    1, 1, -1;
    1, 1, -1;

    -1, 1, -1;
    -1, 1, -1;
    -1, 1, -1];

n1 = [
% 1-12
    -1, 1, 0;
    0, -1, 1;
    1, 0, -1;

```

```

-1, -1, 0;
1, 0, 1;
0, 1, -1;

1, -1, 0;
0, 1, 1;
-1, 0, -1;

1, 1, 0;
-1, 0, 1;
0, -1, -1;
% 13-24
1, 0, -1;
-1, 1, 0;
0, -1, 1;

0, 1, -1;
-1, -1, 0;
1, 0, 1;

-1, 0, -1;
1, -1, 0;
0, 1, 1;

0, -1, -1;
1, 1, 0;
-1, 0, 1;];

n1=n1';

nn=zeros(3,nslip);
nn1=zeros(3,nslip);
nb=zeros(nslip,3);
M1=zeros(3,3,nslip);
M2=zeros(3,3,nslip);
M3=zeros(3,3,nslip);
for xx=1:nslip
    % Normalizing the slip planes&directions
    for j=1:3
        nn(j,xx)=n(j,xx)/sqrt(2);
        nb(xx,j)=b(xx,j)/sqrt(3);
        nn1(j,xx)=n1(j,xx)/sqrt(2);
    end

    dummy_n = cross(nn(1:3,xx)', nb(xx,1:3));
    dummy_n1 = cross(nn1(1:3,xx)', nb(xx,1:3));

    % Forming the Schmid Tensor
    for j=1:3
        for k=1:3
            % Schmid tensor
            M(j,k,xx)=nb(xx,j)*nn(k,xx);
            % Non-Schmid dyadic: responsible for twinning and anti-
twinning
            M1(j,k,xx)=nb(xx,j)*nn1(k,xx);
            % Non-Schmid dyadic: responsible for tension-compression
asymmetry

```

```

        M2(j,k,xx)=dummy_n(j)*nn(k,xx);
        % Non-Schmid dyadic: responsible for tension-compression
asymmetry
        M3(j,k,xx)=dummy_n1(j)*nn1(k,xx);
        % Climb tensor
        Cl(j,k,xx)=nn(j,xx)*nn(k,xx);
    end
end
end

% % Transform the Schmid tensor
% for xx=1:12
%     M(1:3,1:3,xx)=g*M(1:3,1:3,xx)*g';
% end
%
% Vectorize Schmid tensor
for xx=1:nslip

    Climb(1,xx)=Cl(1,1,xx);
    Climb(2,xx)=Cl(2,2,xx);
    Climb(3,xx)=Cl(3,3,xx);
    Climb(4,xx)=Cl(2,3,xx)+Cl(3,2,xx);
    Climb(5,xx)=Cl(1,3,xx)+Cl(3,1,xx);
    Climb(6,xx)=Cl(1,2,xx)+Cl(2,1,xx);

    Schmid(1,xx)=M(1,1,xx);
    Schmid(2,xx)=M(2,2,xx);
    Schmid(3,xx)=M(3,3,xx);
    Schmid(4,xx)=M(2,3,xx)+M(3,2,xx);
    Schmid(5,xx)=M(1,3,xx)+M(3,1,xx);
    Schmid(6,xx)=M(1,2,xx)+M(2,1,xx);

    NonSchmid_1(1,xx)=M1(1,1,xx);
    NonSchmid_1(2,xx)=M1(2,2,xx);
    NonSchmid_1(3,xx)=M1(3,3,xx);
    NonSchmid_1(4,xx)=M1(2,3,xx)+M1(3,2,xx);
    NonSchmid_1(5,xx)=M1(1,3,xx)+M1(3,1,xx);
    NonSchmid_1(6,xx)=M1(1,2,xx)+M1(2,1,xx);

    NonSchmid_2(1,xx)=M2(1,1,xx);
    NonSchmid_2(2,xx)=M2(2,2,xx);
    NonSchmid_2(3,xx)=M2(3,3,xx);
    NonSchmid_2(4,xx)=M2(2,3,xx)+M2(3,2,xx);
    NonSchmid_2(5,xx)=M2(1,3,xx)+M2(3,1,xx);
    NonSchmid_2(6,xx)=M2(1,2,xx)+M2(2,1,xx);

    NonSchmid_3(1,xx)=M3(1,1,xx);
    NonSchmid_3(2,xx)=M3(2,2,xx);
    NonSchmid_3(3,xx)=M3(3,3,xx);
    NonSchmid_3(4,xx)=M3(2,3,xx)+M3(3,2,xx);
    NonSchmid_3(5,xx)=M3(1,3,xx)+M3(3,1,xx);
    NonSchmid_3(6,xx)=M3(1,2,xx)+M3(2,1,xx);

end
%

% The 4th order Schmid x Schmid tensor

```

```

% Sum over the slip systems
Msum=zeros(3,3);
Psum=zeros(3,3,3,3);
for xx=1:nslip
    for i=1:3
        for j=1:3
            Msum(i,j)=Msum(i,j)+M(i,j,xx);
            for k=1:3
                for l=1:3
                    P(i,j,k,l,xx)=M(i,j,xx)*M(k,l,xx);
                    Psum(i,j,k,l)=Psum(i,j,k,l)+P(i,j,k,l,xx);
                end
            end
        end
    end
end
end
end

```

```

% Temperature (K)
Temp=298;

```

```

% Stiffnesses

```

```

% % Isotropic Stiffness
% % A=1
% C11=2*C44+C12;

```

```

% Cubic Stiffness
% % For Aluminum: [*]Deformation Geometry; Reid, p.69
% C11=106.78e9; %MPa
% C12=60.74e9; %MPa
% C44=28.21e9; %MPa
% % For Aluminum: Roters
% C11=106.75e3; %MPa
% C12=60.41e3; %MPa
% C44=28.31e3; %MPa
% % For Aluminum: Anand
% % Anand's temperature dependent constants
% Cmat=[ 123.323 6.7008e-8 -1.1342e-4 -7.8788e-3;
%        70.6512 4.4105e-8 -7.5498e-5 3.9992e-3;
%        31.2071 7.0477e-9 -1.2136e-5 -8.3274e-3];
% Tvec=[1; Temp^3; Temp^2; Temp];
% Cs=Cmat*Tvec;
% C11=Cs(1)*1e9;C12=Cs(2)*1e9;C44=Cs(3)*1e9;

```

```

% % For Copper
% C11=168.0e3; %MPa
% C12=121.4e3; %MPa
% C44=75.4e3; %MPa

```

```

% For Steel
C11 = (239260 - 24520/(exp(392.03/Temp)-1))*1e6;

```

```

C12 = (135780 - 6550/(exp(469.65/Temp)-1))*1e6;
C44 = (120720 - 3500/(exp(162.9/Temp)-1))*1e6/2;

% % For Steel
% C11 = 230.27e+9;
% C12 = 134.07e+9;
% C44 = 57.95e+9;

% Stiffness Matrix(6x6)
Zeta=[ C11 C12 C12 0 0 0;
        C12 C11 C12 0 0 0;
        C12 C12 C11 0 0 0;
        0 0 0 C44 0 0;
        0 0 0 0 C44 0;
        0 0 0 0 0 C44 ];

invZeta=inv(Zeta);

[Zeta_3333] = Convert66to3333(Zeta);

% To make it as work conjugate for the vectorized strains:
Zeta_wc=Zeta;
Zeta_wc(4,4)=2*Zeta(4,4);
Zeta_wc(5,5)=2*Zeta(5,5);
Zeta_wc(6,6)=2*Zeta(6,6);

% Initial dislocation density (1/m^2)
rho_0=1e11;

% Deformation gradient at time=t (F_t)
% Slip system resistivity at time=t (s_t)
% 1st PK stress state at time=t (T_t)
% Plastic part of the deformation gradient at time=t (Fp_t)
% Deformation gradient at time=tau (F_tau)
%
F_t=eye(3,3);
% Fe_t=eye(3,3);
% Fp_t=eye(3,3);
%
% Numerical integration weight
% 0.95 is found to be good for rate=0.1 and increment=100
scale=0.95;
%

% Loop exit tolerances
%
innertol=1e-6;%1e-4*s_o;
outertol=1e-3;%1e-3*s_o;
innoitmax=1000;
ounoitmax=1000;

```

```

%
% -----

% Penalty Parameter
%
lambda_p=1000;
%

% Material Constants

% Non-Schmid parameters
% % If 143 K <= T <= 298 K
% % From ref. A. Patra et al. IJP 59, 1-14, 2014.
% a2 = 0.1727 + 108.9126*exp(-0.0496*Temp);

% Yalcinkaya
% a2=-0.0293;

% % Patra
% a2 = -0.0333;

% % NO Non-Schmid effects
% a2=0;

% % Custom numbers
% a1=0.15;
% a2=0.1; % positive a2 helps
% a3=-0.55; % negative a3 helps

% Custom numbers
a1=0.22;
a2=0.16; % positive a2 helps
a3=-0.64; % negative a3 helps

% Dislocation-based Model Inputs
%
% Boltzman Constant (J/K)
K=1.38e-23;
% Shear Modulus (Pa)
G=2*sqrt((C11-C12)*C44/2);
% G = (87600-17*Temp)*1e6;
% Poissons ratio
nu=-invZeta(1,2)/invZeta(1,1);

```

```

% % Reference shear rate
% gammadot0=1e-3;

% Burgers Vector b = a/2*(sqrt(h^2+k^2+l^2)) (m)
% Burger=3.6e-10/sqrt(2);
Burger = 2.866e-10*sqrt(3)/2;

% Activation Energy for Peirls stress (J) (0.65-0.9)
Qpei = 0.95*1.0894824e-19;

% Attack Frequency (1/s)
v0=1e10;
%
% % % For copper
% % % Passing stress (unit)
% % % Range for c2: 0.1 < c1 < 0.5
% % % if 0.2 very close to 17MPa
% c1=0.5;
% % % Jump distance (unit)
% % % Range for c1: 1 < c2 < 10
% c2=1;
% % % Activation volume (unit)
% % % % Range for c3: 1 < c3 < 10
% c3=1;
% % % Immobilization of mobile dislocations due to mean free path
defined by
% % % forest dislocations (1/m)
% % % Range for c4: -
% % %5.6e-3 is found from the fit
% % %1.27e-2 (Estrin & Kubin)
% % %1/54 (Anxin Ma -- 1/c2/c4)
% % c3=1/54;
% % % Immobilization of mobile dislocations
% % %5.53(Estrin & Kubin)
% c4=1;
% % % Annihilation of immobile dislocations thorough climb and (unit)
% % % Range for c5: -
% % %2.2 is found from the fit
% % %6.67 (Estrin & Kubin)
% % %50 (Anxin Ma)
% % %9.530405405405406e-008 x k1
% c5=50;
% %
% % % Gottstein's mobile density eqn
% % % Scale
% c6=1;
% % % % Stacking Fault Energy (J/m^2)
% % % %chi=45e-3;

% For aluminum
% Geometric constant for long range stress (unit)

```

```

% Range for c1: 0.1 < c1 < 0.5
% if 0.2 very close to 17MPa
c1=0.28;
% % % % Jump distance
% % % % Range for c4: 1 < c4 < 10
c2=15;
% % % % Activation volume (unit)
% % % % Range for c3: 1 < c3 < 10
c3=5;
% Immobilization of mobile dislocations due to mean free path defined
by
% forest dislocations (1/m) - slip c4=0.0015; - tensile c4=0.0015;
% Range for c4: -
% 5.6e-3 is found from the fit
% 1.27e-2 (Estrin & Kubin)
% 1/54 (Anxin Ma -- 1/c2/c4)
%c4=0.1;
c4=0.01;

% Climb distance constant (10)
c5=10;
d_climb=c5*Burger;
Omega=Burger^3;

% Cross-slip width of site multiplier (c6=1000)
c6=8000;

% Cross-slip constant (A=5e-4)
A=5e-4;
% Cross-slip constant (B=0.07)
B=0.07;
% Stacking Fault Energy (J/m^2)
chi=80e-3;
% Cross-slip constant
Qcs=A*G*Burger^3*(G*Burger/chi)*(abs(log(B*G*Burger/chi)))^0.5;

% Scaling constant for mobile density
c7=0.1;

% Scaling constant for dipole storage
c8=0.08;

% Constant calculations
% Temperature x Boltzmann constant
KT=K*Temp;

% Velocity multiplier
m1=v0*exp(-Qpei/KT);
% Fraction of mobile density
m2=(2*pi)^(3/2)/Burger^2/G^2*c7;
% Dipole formation multiplier
m3=sqrt(3)*G/(16*pi*(1-nu))*c8;
% Climb multiplier
m4=Omega*d_climb/(2*Burger*KT);
% Cross-slip multipliers
m5=2*v0/c6^2/Burger;
m6=Qcs*(abs(log(B*G*Burger/chi)))^(-0.5)*Burger/chi;

```

```

% Climb constants
% Volume diffusion activation energy (J/mol)
Qv=197e3;
% Volume diffusion coefficient (m^2/s)
Dv0=2e-5;
% Core diffusion activation energy (J/mol)
Qc=117e3;
% Core diffusion coefficient (m^4/s)
acDc0=1e-24;

% Climb coefficients
Dv=Dv0*exp(-Qv/KT);
acDc=acDc0*exp(-Qc/KT);

% Shear wave speed (m/s)

% Drag constant
cd=45;

% % Copper
% %
https://www.0wikipedia.org/index.php?q=aHR0cHM6Ly91bi53aWtpcGVkaWEub3JnL3dpa2kvQ29wcGVy
% density=8860;
% Aluminum
% density (kg/m^3)
% The engineering toolbox - Aluminum 1100/6061
% density=2720;

% Steel
density = 7890;

vs=sqrt(G/density);

% % Number of atoms per unit cell
% z=4;

% Damping coefficient
%B0=3*KT*z*cd/vs/Burger^2/20;
B0=KT*cd/vs/Burger^2;

% _____

% Calculate projection constants

```

```
% Dislocation line on slip system "b", leads to an increase in the
forest
% density on slip plane "a"
% ==> Direct measure for forest density
%for a=1:12
%    for b=1:12
%
%prod(a,b)=(abs(dot(nn(1:3,a),nt(b,1:3)))+abs(dot(nn(1:3,a),nb(b,1:3)))
)/sqrt(2);
%    end
%end
```

```
% Initial Value for SSD (1e8-1e9) (m^-2)
rho0=rho_0*ones(1,nslip);
```

```
% %
% Self interaction
G0=1.0;
% Co-planar interaction
g1=1.5;
% Cross-slip interaction
g2=3.2;
% Glissile junction
g3=2.4;
% Hirth locks
g4=1.8;
%
```

```
Interaction = ...
[ ...
0 , G0 , G0 , g1 , g1 , g1 , g3 , g3 , g3 ,
g3 , g3 , g3 , 0 , G0 , G0 , g1 , g1 , g1 ,
g3 , g3 , g3 , g3 , g3 , g3 , g1 , g1 , g1 ,
G0 , 0 , G0 , g3 , g3 , g3 , g3 , g3 , g3 ,
g3 , g3 , g3 , G0 , 0 , G0 , g3 , g3 , g3 ,
g1 , g1 , g1 , g3 , g3 , g3 , g3 , g3 , g3 ,
G0 , G0 , 0 , g3 , g3 , g3 , g3 , g3 , g3 ,
g1 , g1 , g1 , G0 , G0 , 0 , g3 , g3 , g3 ,
g3 , g3 , g3 , g1 , g1 , g1 , 0 , G0 , G0 ,
g3 , g3 , g3 , g3 , g3 , g3 , g3 , g3 , g3 ,
g3 , g3 , g3 , G0 , 0 , G0 , g3 , g3 , g3 ,
g1 , g1 , g1 , g3 , g3 , g3 , G0 , 0 , G0 ,
g3 , g3 , g3 , g3 , g3 , g3 , G0 , G0 , 0 ,
g1 , g1 , g1 , g3 , g3 , g3 , 0 , G0 , G0 ,
g3 , g3 , g3 , g3 , g3 , g3 , g4 , g4 , g4 ,
g2 , g2 , g2 , g4 , g4 , g4 , g4 , g4 , g4 ,
0 , G0 , G0 , g2 , g2 , g2 , g2 , g2 , g2 ,
g1 , g1 , g1 , g3 , g3 , g3 , G0 , 0 , G0 ,
g3 , g3 , g3 , g1 , g1 , g1 , g3 , g3 , g3 ,
G0 , 0 , G0 , g3 , g3 , g3 , g3 , g3 , g3 ,
```

```

g3 , g3 , g3 , g1 , g1 , g1 , G0 , G0 , 0 ,
g3 , g3 , g3 , g3 , g3 , g3 , g1 , g1 , g1 ,
G0 , G0 , 0 , g3 , g3 , g3;
g4 , g4 , g4 , g4 , g4 , g4 , g2 , g2 , g2 ,
0 , G0 , G0 , g4 , g4 , g4 , g4 , g4 , g4 ,
g2 , g2 , g2 , 0 , G0 , G0;
g3 , g3 , g3 , g1 , g1 , g1 , g3 , g3 , g3 ,
G0 , 0 , G0 , g3 , g3 , g3 , g1 , g1 , g1 ,
g3 , g3 , g3 , G0 , 0 , G0;
g1 , g1 , g1 , g3 , g3 , g3 , g3 , g3 , g3 ,
G0 , G0 , 0 , g1 , g1 , g1 , g3 , g3 , g3 ,
g3 , g3 , g3 , G0 , G0 , 0;
G0 , G0 , G0 , g2 , g2 , g2 , g4 , g4 , g4 ,
g4 , g4 , g4 , 0 , G0 , G0 , g2 , g2 , g2 ,
g4 , g4 , g4 , g4 , g4 , g4;
G0 , G0 , G0 , g4 , g4 , g4 , g2 , g2 , g2 ,
g4 , g4 , g4 , G0 , 0 , G0 , g4 , g4 , g4 ,
g2 , g2 , g2 , g4 , g4 , g4;
G0 , G0 , G0 , g4 , g4 , g4 , g4 , g4 , g4 ,
g2 , g2 , g2 , G0 , G0 , 0 , g4 , g4 , g4 ,
g4 , g4 , g4 , g2 , g2 , g2;
g2 , g2 , g2 , G0 , G0 , G0 , g4 , g4 , g4 ,
g4 , g4 , g4 , g2 , g2 , g2 , 0 , G0 , G0 ,
g4 , g4 , g4 , g4 , g4 , g4;
g4 , g4 , g4 , G0 , G0 , G0 , g4 , g4 , g4 ,
g2 , g2 , g2 , g4 , g4 , g4 , G0 , 0 , G0 ,
g4 , g4 , g4 , g2 , g2 , g2;
g4 , g4 , g4 , G0 , G0 , G0 , g2 , g2 , g2 ,
g4 , g4 , g4 , G0 , 0 , G0 , g2 , g2 , g2 ,
g2 , g2 , g2 , g4 , g4 , g4;
g4 , g4 , g4 , G0 , G0 , G0 , g4 , g4 , g4 ,
g2 , g2 , g2 , g4 , g4 , g4 , g4 , g4 , g4 ,
0 , G0 , G0 , g2 , g2 , g2;
g2 , g2 , g2 , g4 , g4 , g4 , G0 , G0 , G0 ,
g4 , g4 , g4 , g2 , g2 , g2 , g4 , g4 , g4 ,
G0 , 0 , G0 , g4 , g4 , g4;
g4 , g4 , g4 , g2 , g2 , g2 , G0 , G0 , G0 ,
g4 , g4 , g4 , g4 , g4 , g4 , g2 , g2 , g2 ,
G0 , G0 , 0 , g4 , g4 , g4;
g4 , g4 , g4 , g4 , g4 , g4 , g2 , g2 , g2 ,
G0 , G0 , G0 , g4 , g4 , g4 , g4 , g4 , g4 ,
g2 , g2 , g2 , 0 , G0 , G0;
g4 , g4 , g4 , g2 , g2 , g2 , g4 , g4 , g4 ,
G0 , G0 , G0 , g4 , g4 , g4 , g2 , g2 , g2 ,
g4 , g4 , g4 , G0 , 0 , G0;
g2 , g2 , g2 , g4 , g4 , g4 , g4 , g4 , g4 ,
G0 , G0 , G0 , g2 , g2 , g2 , g4 , g4 , g4 ,
g4 , g4 , g4 , G0 , G0 , 0;

```

```
];
```

```

% % Interaction=Interaction + eye(nslip);
% %
% %

```

```

% TWEAK THE HARDENING
% %%%%%%%%%%%

```

```

dummy=zeros(nslip,nslip);
% Calculate forest dislocation projections
for i=1:nslip
    for j=1:nslip
        dummy(i,j)=abs(dot(nn(:,i),nn(:,j)));
    end
end

Projection = dummy;
% % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Projection = Interaction;

% % Weight of initial plastic guess (0<..<1)
% %
% _____
% weight=0.1;
% %
% _____

%
% % The tolerance for the jacobian
% %
% _____
% dJ=1e-6;
% %
% _____

% Input crystal orientations
% _____
% load oris_100.mat

% no_grain=size(oris,1);
% Assign the initial deformation gradients and elasticity matrices in
the
% finite element reference
Fp_t0=g;
Fe_t0=g';

%
% _____

% Clear extra variables
clear Cmat Tvec Cs dummy dummy_n

```

```

% Oct. 20, 2019
% Mohammad Hasan Joudivand

function
[invFp_T,T_T_vec,gammadot,dgammadot_dtau,Lp,rho,rhoM,tau,inneriterno,o
uteriterno]...
    = main_func(F_T,Fe_t,Fp_t,rho_t,dt)

global ounoitmax
global innoitmax
global innertol
global outertol
global scale
global nslip

% Initial Fully plastic guess
Fp_T=inv(Fe_t)*F_T;
% Overall Right Cauchy
C=F_T'*F_T;

rho=rho_t;

for outeriterno=1:ounoitmax
    % Calculate the constants dependin on the state variable
    [vel_const,rhoM_const,tau_mu,AV,rhof] =constants_func(rho);
    % Assign an initial high value of residual
    xx1=1e10;
    % INNER loop starts here!
    for inneriterno=1:innoitmax
        % Inverse of plastic part of deformation gradient
        invFp_T=inv(Fp_T);
        % Find constitutive parameters

        [Ce,T_T_vec,tau,tau_eff,tau_ns,sigma,gammadot,dgammadot_dtau,Lp,rhoM]
        =...

        constitutive_func(C,invFp_T,tau_mu,vel_const,rhoM_const,AV,rhof);
        % Find the residual
        [R,normR] =...
            residual_func(invFp_T,Fp_t,Lp,dt);
        % Vectorize residual
        R_vec=Convert33to9(R);
        % Assign the old value of old residual
        xx1_old=xx1;
        xx1=normR;

        %
        pause
        % Place an exit criteria
        if normR<innertol & inneriterno==1
        %
            disp('Inner loop converged without any iteration!')
            break
        elseif normR<innertol
        %
            disp('Inner loop converged!')
            break
        elseif normR>innertol & xx1>xx1_old & inneriterno>1
            % Scale back

```

```

%           disp('Scaled back the initial guess!')
%
Fp_T=((1-scale)*Fp_T)+(scale*Fp_T_old);
% Go back to initial part(continue
calculation(calculate=1))
% Don't do any calculation go to initial step (line 78)
xx1=1e10;
% Assign old a high value for residual and start iteration
% again

else
% Continue to iterate with the new value
[dFp_T]=...
    tangent_func(R_vec,Ce,invFp_T,Fp_t,dgammadot_dtau,dt);
% Assign the old value of Fp_T
Fp_T_old=Fp_T;
% Add the plastic deformaton increment
Fp_T=Fp_T+dFp_T;
end
end

if inneriterno==innoitmax
    disp('Inner loop has reached the maximum number of
iterations!!!')
end

% INNER loop ends here!
rho_old=rho;

% State variable update for the constitutive incremental changes
[rhodot] = hardening_func(tau_eff, tau, tau_ns, sigma, gammadot,
rho, rhoM, rhof);

rho = rho_t+(rhodot*dt);
res_rho = rho - rho_old;

xx2=zeros(1,nslip);

% Relative tolerance
for i=1:nslip
    xx2(i)=abs(res_rho(i)/rho_old(i));
end

if xx2<outertol
    break
%           disp('Outer loop converged!')
end

if outeriterno==ounoitmax

```

```

        disp('Outer loop has reached the maximum number of
iterations!!!')
    end

end

return

```

6.4 MATLAB code of EBSD data processing using MTEX toolbox

```

% Mohammad Hasan Joudivand
% Date 18/10/2022
% This code is written to load and process the EBSD data using the
MTEX
% package

clear e region_and_error Region_min_error
%
=====
===
% inputs of the plot
% angular resolution of the grains in degrees (significantly
influences the
% number of grains)
res_ang=0.05;
% grains size which are very small and can be removed in micrometer
gr_size=0.01;
% define a square unit cell in micrometer (to swithc from hexagonal to
the
% square grids)
% length of the unitcell side
a=0.05;
unitCell = [-a -a;-a a;a a;a -a];

% size of the rectangle along X in micrometer
L_x=[25];

% size of the rectangle along Y in micrometer
L_y=[25];

% Gap between the rectangles on the plot in micrometer
gap=2;

% initialization
region=[37 51 L_x L_y];

```

```

%
=====
===
% filling the notIndexed zones of the EBSD plot
% consider only indexed data
ebstd = ebstd('indexed');
% reconstruct the grain structure
[grains,ebstd.grainId,ebstd.mis2mean] =
calcGrains(ebstd,'angle',res_ang*degree);
% remove some very small grains
ebstd(grains(grains.grainSize<gr_size)) = [];
% redo grain segmentation
[grains,ebstd.grainId] = calcGrains(ebstd,'angle',res_ang*degree);
% smooth grain boundaries
grains = smooth(grains,gr_size);

% plot the orientation map
plot(ebstd('Iron (Alpha)'),ebstd('Iron (Alpha)').orientations)
% and on top the grain boundaries
hold on
plot(grains.boundary,'linewidth',1.5)
rectangle('position',region,'edgecolor','y','linewidth',2)
hold off

% -----
-----

% plotting the filled EBSD figure
ebstd_filled = fill(ebstd);
% standardizing the filled data
ebstd_filled=ebstd_filled('indexed');

figure
plot(ebstd_filled('Iron (Alpha)'),ebstd_filled('Iron
(Alpha)').orientations);
hold on
plot(ebstd_filled('Iron (Gamma)'),ebstd_filled('Iron
(Gamma)').orientations);
hold on
plot(grains.boundary,'linewidth',1.5)
rectangle('position',region,'edgecolor','y','linewidth',2);
hold off

% -----
% Plotting orientation (Pole figure)
% the vectors in the pole figure
cs = ebstd_filled('Iron (Alpha)').CS;
ori = ebstd_filled('Iron (Alpha)').orientations;

figure
plotPDF(ori,[Miller(0,0,1,cs),Miller(0,1,1,cs),Miller(-
1,1,1,cs)], 'contourf');

```

```

% -----
% saving the pole figure
% the orientations of the Alpha phase
% point scatter for plotting the pole figure in specific locations
r = regularS2Grid('resolution',0.2*degree);

h = Miller({0,0,1},{0,1,1},{-1,1,1},cs);

% calculating the odf of the main plot
odf = calcDensity(ebsd_filled('Iron (Alpha)').orientations);

% saving the pole figure
pf = calcPoleFigure(odf,h,r);

% % % plotting the standardized illustration of the pole figure
% % figure
% % plot(pf);
% % colorbar;

%
=====
===
% -----
-----

% Switching from hexagon to square grids
% picking data corresponding to the EBSD region

% creating related folder
foldername=strcat('Regions_EBSD');
mkdir(string(foldername));

% accuiring current directory
current_dir1=cd;

% additional to the directory
folder_directory=strcat(current_dir1,'\ ',foldername);


n1=size(L_x,1);
region_and_error=zeros(1,3);
counter=1;
for i=1:n1

    region(1,4)=L_y(i,1);

    for j=1:n1

        region(1,3)=L_x(j,1);

        % end point coordinate of the first region
        Y=L_y(i,1);

```

```

region(1,2)=0.0;

while (Y<=100)
    % end point coordinate of the first region
    X=L_x(j,1);
    region(1,1)=0.0;
    while (X<=100)

ebsd_region=ebsd_filled(inpolygon(ebsd_filled,region));

        % use the square unit cell for gridify the selected
region
        ebsdS=ebsd_region.gridify('unitCell',unitCell);

%           figure
%           % visualize the result
%           plot(ebsdS('Iron (Alpha)'),ebsdS('Iron
(Alpha)').orientations);

        % determining the grains of the selected region
grainsS=calcGrains(ebsdS,'angle',res_ang*degree);

%           hold on
%           % plotting the boundary of the grains
%           plot(grainsS.boundary,'lineWidth',2)

%           % -----
%           % Plotting orientation of the selected region (Pole
figure)
        csS = ebsd_region('Iron (Alpha)').CS;
        oriS = ebsd_region('Iron (Alpha)').orientations;
%           figure
%           plotPDF(oriS,[Miller(0,0,1,csS),Miller(0,1,1,csS),Miller(-
1,1,1,csS)], 'contourf');
        % -----
        % saving the pole figure of the selected region
        % the orientations of the Alpha phase

        % calculating the odf of the main plot
        odfS = calcDensity(ebsd_region('Iron
(Alpha)').orientations);

        % compares two different pole figure with same S2Grid
        e(counter,1) = calcError(odf,odfS);

        % saving the pole figure
        pf2 = calcPoleFigure(odfS,h,r);

region_and_error(counter,:)= [region(1,1),region(1,2),e(counter,1)];

%           % plotting the standardized illustration of the pole
figure
%           figure

```

```

%           plot(pf2);
%           colorbar;

% saving the image
print(gcf,'-dpng','-
r72',strcat(string(folder_directory),'\\',[num2str(counter),'_PF']));

-----
% plotting the difference of the pole figure pair
figure
plotDiff(pf,pf2);
colorbar;

saving the image
Description text box alongside the plot
figure
mm=annotation(gcf,'textbox', [0.3, 0.5, 0.4, 0.4],
'String', {[ 'X starts at: ',num2str(region(1,1))];
            [ 'Y starts at: ',num2str(region(1,2))];
            [ 'X ends at: ',num2str(region(1,3))];
            [ 'Y ends at: ',num2str(region(1,4))];
            [ 'Error between the plots is:
',num2str(e(counter,1))]}]);

print(gcf,'-dpng','-
r72',strcat(string(folder_directory),'\\',num2str(counter)));

close all

disp('-----
-----')
disp('-----
-----')
disp('A figure just created')
counter
counter=counter+1;

% updating region
region(1,1)=region(1,1)+gap;

% updating the end point along X axis
X=X+gap;

end
% updating region
region(1,2)=region(1,2)+gap;

% updating the end point along X axis
Y=Y+gap;
end
end
end

min_error=min(region_and_error(:,3));

```

```

Location_error=find(region_and_error(:,3)==min_error);
Region_min_error=region_and_error(Location_error,:);

% -----
% creating the mesh file
% inputs of the mesh (enlarges the mesh size)
Element_Size=0.65;
% Size gradient (no effect on the final results)
S_G=1;
% Element size Depending on the curvature of grain boundaries (no
effect on
% the final results since the grids are converted into square shape
from
% hexagonal shape)
K_v=1;

% The geometry described by the object G
G=gmsHGeo(grainsS);

mesh(G,'DP600_meshed.inp','ElementSize',Element_Size,'gradient',S_G,'E
lementType','Brick')
mesh(G,'DP600_meshed.inp','ElementSize',Element_Size,'ElementType','Br
ick','Curvature',K_v)
mesh(G,'DP600_meshed.inp','ElementSize',Element_Size,'ElementType','Br
ick')

savegeo(G,'DP600_meshed.geo')

load handel;
sound(y,Fs);

```

6.5 MATLAB script for phase detection from Kikuchi quality images

```

% Mohammad Hasan Joudivand
% 11/09/2022

% this code is written to insert martensite distribution image taken
using
% Kikuchi method to assign martensite phase to the grains and to make
% distinction between ferrite and martensite in the calculations

clear
clc

% number of horizontal cells
% there are three main numbers for n_x 51 (0.49), 67 (0.373) and 101
(0.247)
n_x=101;
l_x=25;

```

```

% number of vertical cells
% there are three main numbers for n_x 51 (0.49), 67 (0.373) and 100
(0.25)
n_y=100;
l_y=25;

% inserting the image file
img = imread('MART_regions_NoMicronBar.jpg');
% image(img);

% image ratio
i_r=size(img,1)/100;

% selecting the RVE box
% this command crops the image into the area of interest
I2 =
imcrop(img,[ceil(28*i_r),ceil(73*i_r),ceil(l_x*i_r),ceil(l_y*i_r)]);

%resize image
I3=imresize(I2, [n_x n_y]);

% contrast calibrator
% to increase black grains (martensite) the number should be increased
% with c_c=185 the area fraction of the martensite to ferrite (black
grains
% to white grains) is approximately 4.3%
% in the case of 101x100 grains c_c number should be 170
if n_x==51 | n_x==67
    c_c=185;
elseif n_x==101
    c_c=170;
end

% defining RGB of the pure BW image
for i=1:n_x

    for j=1:n_y

        if I3(i,j,1)>=c_c || I3(i,j,2)>=c_c || I3(i,j,3)>=c_c
            rgb_BW(i,j,1)=1;
            rgb_BW(i,j,2)=1;
            rgb_BW(i,j,3)=1;

        elseif I3(i,j,1)<c_c || I3(i,j,2)<c_c || I3(i,j,3)<c_c

            rgb_BW(i,j,1)=0;
            rgb_BW(i,j,2)=0;
            rgb_BW(i,j,3)=0;
        end
    end
end
end

```

```

% % Binarize the image based on the pixel contrasts
% BW = imbinarize(I3);

% illustrating the resized image
image(rgb_BW)

% setting X and Y axes limits
xlim([0 n_x]);
ylim([0 n_y]);

% reversing Y axis
ax = gca;
% ax.YDir = 'normal';

% grid on
% grid minor

[rows, columns, numberOfColorChannels] = size(rgb_BW);

StepSizeX=size(I3,1)/n_x;

StepSizeY=size(I3,2)/n_y;

% plotting grid lines
for i=0.5:StepSizeX:size(rgb_BW)
    xline(i,'k')
    yline(i,'k')
end

% counter for grain phase determination operation
counter=1;

% it is VERY IMPORTANT !!!
% to notice that the grain numbers increase in a columnar pattern,
this
% means that grain number increases from bottom to top in a column and
then
% when the last grain in a column is done, the numbering sequence sets
back
% to the lowest grain in the next column

for j=1:size(rgb_BW,2)
    for i=1:size(rgb_BW,1)
        if rgb_BW(i,j,1)==1 || rgb_BW(i,j,2)==1 || rgb_BW(i,j,3)==1

            % in grain_phase equals 1 it means it is a ferrite grain
            grain_phase(i,j)=1;
            counter=counter+1;

            elseif rgb_BW(i,j,1)==0 || rgb_BW(i,j,2)==0 ||
rgb_BW(i,j,3)==0

```

```

        % in grain_phase equals 2 it means it is a martensite
    grain
        grain_phase(i,j)=2;
        counter=counter+1;
    end
end

if n_x==101
    grain_phase=grain_phase(1:100,1:100);
    grain_phase(1:100,101)=ones;
end

% sequencing the grains in correct order
counter=1;
for m=1:size(grain_phase,2)
    for n=size(grain_phase,1):-1:1
        phase(counter,1)=grain_phase(n,m);
        counter=counter+1;
    end
end

% writing text file of the grains
num=size(phase,1);
filename=['grain_phaseID_L_x_25_L_y_25_grid_',num2str(n_x),'x',num2str(
n_y),'.txt'];
fileID2= fopen(filename,'w');
for i=1:num

    % writing the phase IDs in the txt file
    formatSpec = '%8.1f\n';
    fprintf(fileID2,formatSpec,phase(i,1))

end

```

6.6 MATLAB script for plotting misorientation pole figure using MSC MARC output files

```

% This code is written by Mohammad Hasan Joudivand
% 02/06/2021
% Plot the misorientation
clear all
clc
%close all

% bypassing the output files
BOF=1;
if BOF==1

```

```

%
=====
===
%      % number of output files
n_out=4;
%
% name of the output files
for i=1:n_out

file_names(i,1)={ [num2str(i), 'l_x_25l_y_25gridsize_05_job1.out']};
%      file_names(i,1)={'model12_job1.out'};
end

%      file_names={'model196_job1.out'};

%
=====
===
% reading the euler angle file
% Euler angles file name
file_name_euler=['Euler_Bunge_L_x_25_L_y_25_grid0.5.dat'];
% reading the text file
eu = fileread(file_name_euler);
eu2=split(eu);
eu3=str2double(eu2);
eu4=eu3(2:end-1,1);
n1=size(eu4,1);
n2=n1/3;
%      n2=49;
euler_angles=zeros(n2,3);
for i=1:n1
    euler_angles(ceil(i/3),round((3*(i/3-ceil((i-3)/3)))))=eu4(i);
end

%
=====
===
% grains that are determined to be tracked
%
gr_list=[122;124;281;434;611;669;687;800;1131;1317;1369;1655;1741;1794
;1916;
%      2004;2007;2243;2330;2371];

gr_list=[330;685;717;907;1024;1104;1173;1296;1489;1639;1778;1929;1961;
2150;2284;2404;2415;2440;2508;2523];
%      gr_list=[36];

%
=====
===
% iteration tolerance
it_tol=1e-14;

%
=====
===
% reading the misorientations from output files of MSC MARC
checkk=1;
el=0;
for j=1:n_out

```

```

% reading the text file
t1 = fileread(char(file_names(j,1)));
t2=split(t1);
% t3=str2double(t2);
r=size(t2,1);

counter=1;
inc=1;
for k=1:r
    if strcmpi(t2(k,1),{'Element_number_flag'})==1
        for n=1:size(gr_list,1)
            if str2num(cell2mat(t2(k+1,1)))==gr_list(n,1)
                el=n;
                break
            else
                el=0;
            end
        end
    end

    if strcmpi(t2(k,1),{'F_elastic_flag'})==1 && el~=0
        Fe(1,1,el,inc)=str2num(cell2mat(t2(k+1,1)));
        Fe(1,2,el,inc)=str2num(cell2mat(t2(k+2,1)));
        Fe(1,3,el,inc)=str2num(cell2mat(t2(k+3,1)));
        Fe(2,1,el,inc)=str2num(cell2mat(t2(k+4,1)));
        Fe(2,2,el,inc)=str2num(cell2mat(t2(k+5,1)));
        Fe(2,3,el,inc)=str2num(cell2mat(t2(k+6,1)));
        Fe(3,1,el,inc)=str2num(cell2mat(t2(k+7,1)));
        Fe(3,2,el,inc)=str2num(cell2mat(t2(k+8,1)));
        Fe(3,3,el,inc)=str2num(cell2mat(t2(k+9,1)));

        % iteration to decouple R and U and find rotation
        tensor
        A=0.5*(Fe(:, :, el, inc)+inv(Fe(:, :, el, inc)'));
        % initial tolerance
        tolerance=100;

        while tolerance>=it_tol

            B=0.5*(A+inv(A')) ;

            tolerance=abs(norm(B)-norm(A)) ;

            A=B;

        end

        R(:, :, el, inc)=B;

    end

    if strcmpi(t2(k,1),{'inc_flag'})==1 && el~=0

```

```

        inc_new=str2num(cell2mat(t2(k+1,1)));
        if inc_new~=inc && inc_new~=0
            counter=counter+1;
            inc=inc_new;
        end
    end

end

end

elseif BOF==2
    load("Fe.mat");
    load("R.mat");
end

%
=====
% the pole figure poles
poles=[1,0,0;
       1,1,0;
       1,1,1];

%
=====

% Crystal Symmetry Operators
crsym(1:3,1:3,1)= [1 0 0;
                  0 1 0;
                  0 0 1];

crsym(1:3,1:3,2)= [0 0 1;
                  1 0 0;
                  0 1 0];

crsym(1:3,1:3,3)= [0 1 0;
                  0 0 1;
                  1 0 0];

crsym(1:3,1:3,4)= [0 -1 0;
                  0 0 1;
                  -1 0 0];

crsym(1:3,1:3,5)= [0 -1 0;
                  0 0 -1;
                  1 0 0];

crsym(1:3,1:3,6)= [0 1 0;
                  0 0 -1;
                  -1 0 0];

```

```

crsym(1:3,1:3,7)= [0 0 -1;
  1 0 0;
  0 -1 0];

crsym(1:3,1:3,8)= [0 0 -1;
  -1 0 0;
  0 1 0];

crsym(1:3,1:3,9)= [0 0 1;
  -1 0 0;
  0 -1 0];

crsym(1:3,1:3,10)= [-1 0 0;
  0 1 0;
  0 0 -1];

crsym(1:3,1:3,11)= [-1 0 0;
  0 -1 0;
  0 0 1];

crsym(1:3,1:3,12)= [1 0 0;
  0 -1 0;
  0 0 -1];

crsym(1:3,1:3,13)= [0 0 -1;
  0 -1 0;
  -1 0 0];

crsym(1:3,1:3,14)= [0 0 1;
  0 -1 0;
  1 0 0];

crsym(1:3,1:3,15)= [0 0 1;
  0 1 0;
  -1 0 0];

crsym(1:3,1:3,16)= [0 0 -1;
  0 1 0;
  1 0 0];

crsym(1:3,1:3,17)= [-1 0 0;
  0 0 -1;
  0 -1 0];

crsym(1:3,1:3,18)= [1 0 0;
  0 0 -1;
  0 1 0];

crsym(1:3,1:3,19)= [1 0 0;
  0 0 1;
  0 -1 0];

crsym(1:3,1:3,20)= [-1 0 0;
  0 0 1;
  0 1 0];

crsym(1:3,1:3,21)= [0 -1 0 ;
  -1 0 0;

```

```

    0 0 -1];

crysym(1:3,1:3,22)= [0 1 0;
    -1 0 0;
    0 0 1];

crysym(1:3,1:3,23)= [0 1 0;
    1 0 0;
    0 0 -1];

crysym(1:3,1:3,24)= [0 -1 0;
    1 0 0;
    0 0 1];

%
=====
==
%
=====
==
% misorientation plotting point extractor

% pole benchmarks
% Plot the pole figure
u=[0 0 1]';
u=u/norm(u);

% in the case of u=[0 1 1], v=[1 0 0]
% in the case of u=[0 0 1], v=[0 1 0]
% in the case of u=[-1 1 1], v=[1 0 1]
if u(1,1)==0 && u(2,1)==0
    v=[0 1 0]';
    v=v/norm(v);
elseif u(1,1)==0 && u(2,1)==1/sqrt(2)
    v=[0 1 -1]';
    v=v/norm(v);
elseif u(1,1)==-1/sqrt(3) && u(2,1)==1/sqrt(3)
    v=[1 0 1]';
    v=v/norm(v);
end

% ND=[0 0 1]';

% normal vector product of the 1st and 2nd axes
w=cross(v,u);
w=w/norm(w);

% passive rotation matrix
% M=[u,v,w];
M=[w,v,u];

% main transformation matrix
% M=M1*RF;

% passive rotatted axes
U=(u'*M)';

```

```

V=(v'*M)';
W=(w'*M)';

% number of increments in the orientation evolution
n1=size(R,4);
% number of grains in the model
n2=size(R,3);
% positive tolerance
t_p=1e-2;
% negative tolerance
t_n=-t_p;
for i=1:n2
    nn=0;
    for j=1:n1
        % crystal to sample
        gs2c=R(:, :, i, j)';

        for k=1:24
            % Crystal symmetry operator
            crysym_c(1:3,1:3)=crsym(1:3,1:3,k);
            pole1=crsym_c*u;

            % passive rotation for the pole figure basic reference
            pole2=(pole1'*M)';

            %Oriented slip systems vector
            vec_ori=gs2c*pole2;

            if vec_ori(3,1)>t_n

                % Equal area correction factor
                (NP=2/sqrt(2)*r*sin(alpha/2))
                %
                vec_pole=vec_ori-ND_to_Pole0;
                vec_pole=vec_ori-U;
                %
                vec_pole=vec_pole/norm(vec_pole);

                alpha_half=acos(dot(vec_pole,U)/(norm(vec_pole)*norm(U)));

                if vec_ori(1,1)>t_n && vec_ori(2,1)>t_n
                    nn=nn+1;
                    theta=atan(vec_ori(2,1)/vec_ori(1,1));
                    %
                    r=sqrt(2)*sin(alpha_half);
                    r=norm(vec_pole);
                    point_coord(nn,1,i)=theta;
                    point_coord(nn,2,i)=r;

                elseif vec_ori(1,1)<t_p && vec_ori(2,1)>t_n
                    nn=nn+1;
                    theta=atan(vec_ori(2,1)/vec_ori(1,1))+pi;

```

```

%           r=sqrt(2)*sin(alpha_half);
           r=norm(vec_pole);
           point_coord(nn,1,i)=theta;
           point_coord(nn,2,i)=r;

elseif vec_ori(1,1)<t_p && vec_ori(2,1)<t_p
    nn=nn+1;
    theta=atan(vec_ori(2,1)/vec_ori(1,1))+pi;
%           r=sqrt(2)*sin(alpha_half);
           r=norm(vec_pole);
           point_coord(nn,1,i)=theta;
           point_coord(nn,2,i)=r;

elseif vec_ori(1,1)>t_n && vec_ori(2,1)<t_p
    nn=nn+1;
    theta=atan(vec_ori(2,1)/vec_ori(1,1))+2*pi;
%           r=sqrt(2)*sin(alpha_half);
           r=norm(vec_pole);
           point_coord(nn,1,i)=theta;
           point_coord(nn,2,i)=r;

        end
    end
end

% starting points of the misorientation
for i=1:n2
    nn=0;
    for j=1:1
        % crystal to sample
        gs2c=R(:, :, i, j)';

        for k=1:24
            % Crystal symmetry operator
            crysym_c(1:3,1:3)=crsym(1:3,1:3,k);
            pole1=crsym_c*u;

            % passive rotation for the pole figure basic reference
change        pole2=(pole1'*M)';

            %Oriented slip systems vector
            vec_ori=gs2c*pole2;

            if vec_ori(3,1)>t_n

                % Equal area correction factor
                (NP=2/sqrt(2)*r*sin(alpha/2))

```

```

%           vec_pole=vec_ori-ND_to_Pole0;
vec_pole=vec_ori-U;
%           vec_pole=vec_pole/norm(vec_pole);

alpha_half=acos(dot(vec_pole,U)/(norm(vec_pole)*norm(U)));

    if vec_ori(1,1)>t_n && vec_ori(2,1)>t_n
        nn=nn+1;
        theta=atan(vec_ori(2,1)/vec_ori(1,1));
%           r=sqrt(2)*sin(alpha_half);
        r=norm(vec_pole);
        point_coord_s(nn,1,i)=theta;
        point_coord_s(nn,2,i)=r;

    elseif vec_ori(1,1)<t_p && vec_ori(2,1)>t_n
        nn=nn+1;
        theta=atan(vec_ori(2,1)/vec_ori(1,1))+pi;
%           r=sqrt(2)*sin(alpha_half);
        r=norm(vec_pole);
        point_coord_s(nn,1,i)=theta;
        point_coord_s(nn,2,i)=r;

    elseif vec_ori(1,1)<t_p && vec_ori(2,1)<t_p
        nn=nn+1;
        theta=atan(vec_ori(2,1)/vec_ori(1,1))+pi;
%           r=sqrt(2)*sin(alpha_half);
        r=norm(vec_pole);
        point_coord_s(nn,1,i)=theta;
        point_coord_s(nn,2,i)=r;

    elseif vec_ori(1,1)>t_n && vec_ori(2,1)<t_p
        nn=nn+1;
        theta=atan(vec_ori(2,1)/vec_ori(1,1))+2*pi;
%           r=sqrt(2)*sin(alpha_half);
        r=norm(vec_pole);
        point_coord_s(nn,1,i)=theta;
        point_coord_s(nn,2,i)=r;

    end

end

end

end

cn1=1;
n5=size(point_coord_s,3);
n6=size(point_coord_s,1);
for i=1:n5
    for j=1:n6
        polar_points_start(cn1,:)=point_coord_s(j,:,i);
        cn1=cn1+1;
    end
end
end

```

```

% poles coordinates of the pole figure
% counter
cn=1;
for i=1:size(poles,1)
    for j=1:24
        pole2=crysym(:, :, j)*poles(i, :)' ;
        pole2=pole2;
        pole2=pole2/norm(pole2);

        if pole2(3,1)>=t_n

            vec_t=pole2-U;

            if pole2(1,1)>=t_n && pole2(2,1)>=t_n

                tet=atand(pole2(2,1)/pole2(1,1));
                r=sqrt(pole2(1,1)^2+pole2(2,1)^2);
                r=norm(vec_t);
                polar_point_coord(cn,1)=tet;
                % if polar_point_coord(cn,1) NaN (not a number)
                if tet~=tet
                    polar_point_coord(cn,1)=0;
                end

                polar_point_coord(cn,2)=r;
                cn=cn+1;

            elseif pole2(1,1)<t_p && pole2(2,1)>=t_n
                tet=180-atand(abs(pole2(2,1))/abs(pole2(1,1)));
                r=sqrt(pole2(1,1)^2+pole2(2,1)^2);
                r=norm(vec_t);
                polar_point_coord(cn,1)=tet;
                % if polar_point_coord(cn,1) NaN (not a number)
                if tet~=tet
                    polar_point_coord(cn,1)=0;
                end
                polar_point_coord(cn,2)=r;
                cn=cn+1;

            elseif pole2(1,1)<t_p && pole2(2,1)<t_n
                tet=180+atand(abs(pole2(2,1))/abs(pole2(1,1)));
                r=sqrt(pole2(1,1)^2+pole2(2,1)^2);
                r=norm(vec_t);
                polar_point_coord(cn,1)=tet;
                % if polar_point_coord(cn,1) NaN (not a number)
                if tet~=tet
                    polar_point_coord(cn,1)=0;
                end
                polar_point_coord(cn,2)=r;
                cn=cn+1;

            elseif pole2(1,1)>t_n && pole2(2,1)<t_n
                tet=360-atand(abs(pole2(2,1))/abs(pole2(1,1)));
                r=sqrt(pole2(1,1)^2+pole2(2,1)^2);
                r=norm(vec_t);
                polar_point_coord(cn,1)=tet;
                % if polar_point_coord(cn,1) NaN (not a number)
                if tet~=tet

```

```

        polar_point_coord(cn,1)=0;
    end
    polar_point_coord(cn,2)=r;
    cn=cn+1;
end
end
end

% making the RD and TD direction consistent
n3=size(point_coord,1);
% initializing polar plot
% p=polar([0;pi/2;pi;3*pi/2;2*pi],[1;1;1;1;1],'.k');

p=polar(polar_point_coord(:,1)*pi/180,polar_point_coord(:,2)/sqrt(2),'.k');
% p=polar(polar_point_coord(:,1)*pi/180,polar_point_coord(:,2),'.k');

p.MarkerSize = 15;

hold on
p=polar(polar_points_start(:,1),polar_points_start(:,2)/sqrt(2),'.r');
% p=polar(polar_points_start(:,1),polar_points_start(:,2),'.r');

p.MarkerSize = 15;

for i=1:n2
    hold on
    p=polar(point_coord(:,1,i),point_coord(:,2,i)/sqrt(2),'.b');
    % p=polar(point_coord(:,1,i),point_coord(:,2,i),'.b');

    p.MarkerSize = 5;
end

% XLim([-1.42 1.42])
% YLim([-1.42 1.42])

% title('CPFEM of TS (tensile sample)')
hh=legend('Numerical - single phase (Ferrite)', 'Experimental', 'Numerical - dual phase (Ferrite-Martensite)');

% set(hh,'Location', 'northoutside')
set(hh,'Location', 'northoutside')
get(gca,'fontname')
set(gca,'fontname','Times New Roman','fontsize',36)
print(gcf,'Misorientation.jpg','-dpng','-r300');

%
=====
===
% saving figure in pdf format

pos = get(gcf,'Position');

```

```

set(gcf, 'PaperPositionMode', 'Auto', 'PaperUnits', 'Inches', 'PaperSize', [
pos(3), pos(4)])
figure('WindowState', 'maximized')
set(gcf, 'WindowState', 'maximized')

print(gcf, 'Narayanan_TS_CPFEM', '-dpdf', '-r300')
%
% %
=====
===
% %
=====
===
%
=====
=====
=====

```

6.7 Dislocation density post processing MATLAB script

```

% This code is written by Mohammad Hasan Joudivand
% 20/10/2023
% Plot the misorientation
clear all
clc
%close all

% bypassing the output files
%
=====
===
% % number of output files
n_out=40;

% name of the output files
for i=1:n_out
    file_names(i,1)={ [num2str(i), 'l_x_25l_y_25gridsize_05_job1.out'] };
    % file_names(i,1)={ ['10x10model_job1-DP.out'] };
end

%
=====
===
% reading the misorientations from output files of MSC MARC
checkk=1;
% no-existing element number definition
elNo=0;

% load file or read the saved arguments
sd=1;

if sd==1
    load Dis_av.mat;
    load Dis.mat;

```

```

elseif sd==2
    for j=1:n_out

        % reading the text file
        t1 = fileread(char(file_names(j,1)));
        t2=split(t1);
        %      t3=str2double(t2);
        r=size(t2,1);

        counter=1;
        inc=0;
        for k=1:r
            if strcmpi(t2(k,1),{'inc_flag'})==1
                inc=str2num(cell2mat(t2(k+1,1)));
            end
            if strcmpi(t2(k,1),{'Element_number_flag'})==1

                %
                elNo=str2num(cell2mat(t2(k+1,1)));

            end

            if elNo~=0 & inc~=0 & strcmpi(t2(k,1),{'rhoM_flag'})==1
                Dis(1,inc,elNo)=str2num(cell2mat(t2(k+1,1)));
                Dis(2,inc,elNo)=str2num(cell2mat(t2(k+2,1)));
                Dis(3,inc,elNo)=str2num(cell2mat(t2(k+2,1)));
                Dis(4,inc,elNo)=str2num(cell2mat(t2(k+3,1)));
                Dis(5,inc,elNo)=str2num(cell2mat(t2(k+4,1)));
                Dis(6,inc,elNo)=str2num(cell2mat(t2(k+5,1)));
                Dis(7,inc,elNo)=str2num(cell2mat(t2(k+6,1)));
                Dis(8,inc,elNo)=str2num(cell2mat(t2(k+7,1)));
                Dis(9,inc,elNo)=str2num(cell2mat(t2(k+8,1)));
                Dis(10,inc,elNo)=str2num(cell2mat(t2(k+9,1)));
                Dis(11,inc,elNo)=str2num(cell2mat(t2(k+10,1)));
                Dis(12,inc,elNo)=str2num(cell2mat(t2(k+11,1)));
                Dis(13,inc,elNo)=str2num(cell2mat(t2(k+12,1)));
                Dis(14,inc,elNo)=str2num(cell2mat(t2(k+13,1)));
                Dis(15,inc,elNo)=str2num(cell2mat(t2(k+14,1)));
                Dis(16,inc,elNo)=str2num(cell2mat(t2(k+15,1)));
                Dis(17,inc,elNo)=str2num(cell2mat(t2(k+16,1)));
                Dis(18,inc,elNo)=str2num(cell2mat(t2(k+17,1)));
                Dis(19,inc,elNo)=str2num(cell2mat(t2(k+18,1)));
                Dis(20,inc,elNo)=str2num(cell2mat(t2(k+19,1)));
                Dis(21,inc,elNo)=str2num(cell2mat(t2(k+20,1)));
                Dis(22,inc,elNo)=str2num(cell2mat(t2(k+21,1)));
                Dis(23,inc,elNo)=str2num(cell2mat(t2(k+22,1)));
                Dis(24,inc,elNo)=str2num(cell2mat(t2(k+23,1)));

                %
                Dis_av(inc,elNo)=sum(Dis(1:24,1,inc))/24;
                Dis_av(inc,elNo)=sum(Dis(:,inc,elNo));
                %
                inc=inc+1;
            end
        end
    end

```

```

        end

    end

end

%
=====
===
% plotting image
% determining the size of grid
% vertical discretization
n_v=51;

%horizontal discretization
n_h=51;

% intended increment to illustrate the distribution
inc_n=1000;

% element counter
ele_cn=1;

for j=1:n_h
    for i=1:n_v

        % elements arrangement
        ele_n_array(i,j)=ele_cn;

        % counting up
        ele_cn=ele_cn+1;

    end
end

% adapting element number arrangment with MSC MARC convention
% ele_n_array=flip(ele_n_array);

% martensite elements detection
mar_ele=0;

% taking the dislocation density values in grid arrangement
for k=1:size(ele_n_array,2)
    for m=1:size(ele_n_array,1)

        % the element number for dislocation density record
        dis_ele=ele_n_array(m,k);

        % average dislocation to plot
        dis_den(m,k)=Dis_av(inc_n,dis_ele);
    end
end

```

```

        if any(mar_ele==dis_ele)
            dis_den(m,k)=0;

        end

    end

end

% contour plot
[X,Y]=meshgrid(linspace(1,n_h,n_h),linspace(1,n_v,n_v));

% plotting contour
[M,c] = contour(X,Y,dis_den);
contourcbar
c.LineWidth = 3;
ax = gca;
ax.LineWidth = 2;
pbaspect([1.1 1 1])
% colormap(winter)
% axis equal

% % dislocation density distribution illustration
% image(dis_den/1e11)
% colorbar

%
% xlabel('Strain')
%
% ylabel('Mobile Dislocation Density')

% % title('CPFEM of TS (tensile sample)')
% hh=legend([legend_name;'Avg. slip Sys.']);
% set(hh,'Location','northeastoutside')
% get(gca,'fontname')
set(gca,'fontname','Times New Roman','fontsize',20)
% print(gcf,'SP1.jpg','-dpng','-r300');
%
% %
=====
===
% %
=====
===
%
=====
=====

```

6.8 MATLAB script for selecting martensite banded area in the EBSD

```
% Mohammad Hasan Joudivand
% Date 12/09/2023
% This code is written to encase a RVE from martensite banded zone on
the
% Kikuchi quality image

clear e region_and_error Region_min_error
%
=====
===
% inputs of the plot
% angular resolution of the grains in degrees (significantly
influences the
% number of grains)
res_ang=0.05;
% grains size which are very small and can be removed in micrometer
gr_size=0.01;
% define a square unit cell in micrometer (to swithc from hexagonal to
the
% square grids)
% length of the unitcell side
a=0.05;
unitCell = [-a -a;-a a;a a;a -a];

% size of the rectangle along X in micrometer
L_x=[25];

% size of the rectangle along Y in micrometer
L_y=[25];

% Gap between the rectangles on the plot in micrometer
gap=2;

% initialization
region=[37 51 L_x L_y];

%
=====
===
% filling the notIndexed zones of the EBSD plot
% consider only indexed data
ebstd = ebstd('indexed');
% reconstruct the grain structure
[grains,ebstd.grainId,ebstd.mis2mean] =
calcGrains(ebstd,'angle',res_ang*degree);
% remove some very small grains
ebstd(grains(grains.grainSize<gr_size)) = [];
% redo grain segmentation
[grains,ebstd.grainId] = calcGrains(ebstd,'angle',res_ang*degree);
% smooth grain boundaries
grains = smooth(grains,gr_size);

% -----
```

```

% Plotting orientation (Pole figure)
% the vectors in the pole figure
cs = ebsd_filled('Iron (Alpha)').CS;
ori = ebsd_filled('Iron (Alpha)').orientations;

% -----
% saving the pole figure
% the orientations of the Alpha phase
% point scatter for plotting the pole figure in specific locations
r = regularS2Grid('resolution',0.2*degree);

h = Miller({0,0,1},{0,1,1},{-1,1,1},cs);

% calculating the odf of the main plot
odf = calcDensity(ebsd_filled('Iron (Alpha)').orientations);

% saving the pole figure
pf = calcPoleFigure(odf,h,r);

%
=====
===
% -----
-----
% Switching from hexagon to square grids
% picking data corresponding to the EBSD region

% creating related folder
foldername=strcat('Regions_EBSD');
mkdir(string(foldername));

% accuiring current directory
current_dir1=cd;

% additional to the directory
folder_directory=strcat(current_dir1,'\ ',foldername);

n1=size(L_x,1);
region_and_error=zeros(1,3);
counter=1;

% end point coordinate of the first region
Y=L_y;

% end point coordinate of the first region
X=L_x;

ebsd_region=ebsd_filled(inpolygon(ebsd_filled,region));

% use the square unit cell for gridify the selected region
ebsdS=ebsd_region.gridify('unitCell',unitCell);

```

```

% determining the grains of the selected region
grainsS=calcGrains(ebsdS,'angle',res_ang*degree);

% -----
% Plotting orientation of the selected region (Pole figure)
csS = ebsd_region('Iron (Alpha)').CS;
oriS = ebsd_region('Iron (Alpha)').orientations;
% -----
% saving the pole figure of the selected region
% the orientations of the Alpha phase
% calculating the odf of the main plot
odfS = calcDensity(ebsd_region('Iron (Alpha)').orientations);

% compares two different pole figure with same S2Grid
e = calcError(odf,odfS);

% saving the pole figure
pf2 = calcPoleFigure(odfS,h,r);

region_and_error(counter,:)= [region(1,1),region(1,2),e(counter,1)];

ebsd_region=ebsd_filled(inpolygon(ebsd_filled,region));

% length of the unitcell side
a1=[25/100];

unitCell2 = [-a1(i,1) -a1(i,1);-a1(i,1) a1(i,1);a1(i,1)
a1(i,1);a1(i,1) -a1(i,1)];

% use the square unit cell for gridify the selected region for the
second
% time
ebsdS2=ebsd_region.gridify('unitCell',unitCell2);

% determining the grains of the selected region
grainsS2=calcGrains(ebsdS2,'angle',res_ang*degree);

% calculating the odf of the main plot
odfS2 = calcDensity(ebsdS2('Iron (Alpha)').orientations);

% transform quaternion to Euler (Bunge) angles
rotS2=rotation(grainsS2.prop.meanRotation);

Euler_Bunge=[rotS2.phi1 rotS2.Phi rotS2.phi2]*180/pi;

num5=size(grainsS2);

% extraction of the phases of the grain
grain_phase=grainsS2(1:num5).phase;

% transform quaternion to Euler (Bunge) angles
rotS2=rotation(grainsS2.prop.meanRotation);

Euler_Bunge=[rotS2.phi1 rotS2.Phi rotS2.phi2]*180/pi;

```

```

num5=size(grainsS2);

% extraction of the phases of the grain
grain_phase=grainsS2(1:num5).phase;

num6=size(Euler_Bunge,1);
fileID = fopen('Euler_Bunge_L_x_25_L_y_25_grid_imp.txt','w');
fileID2= fopen('grain_phaseID_L_x_25_L_y_25_grid_imp.txt','w');
for i=1:num6
    % writing the Euler angles in the txt file
    formatSpec = '%8.3f %8.3f %8.3f\n';

    fprintf(fileID,formatSpec,Euler_Bunge(i,1),Euler_Bunge(i,2),Euler_Bunge(i,3));

    % writing the phase IDs in the txt file
    formatSpec = '%8.1f\n';
    fprintf(fileID2,formatSpec,grain_phase(i,1))
end

```