

Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики
Высшая школа теоретической механики

Работа допущена к защите

Директор высшей школы

_____ А. М. Кривцов

«__» _____ 20__ г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА
«ТЕРМОМЕХАНИЧЕСКАЯ МОДЕЛЬ ИЗУЧЕНИЯ МИКРОСТРУКТУРЫ
ПРИ АДДИТИВНОМ ПРОИЗВОДСТВЕ»**

по направлению 01.04.03 «Механика и математическое моделирование»
по образовательной программе 01.04.03_02 «Механика и математическое
моделирование (международная образовательная программа)»

Выполнил
студент гр. 3640103/80201



Н. Моханан

Руководитель
Доцент, к.ф.-м.н.



В. А. Кузькин

Консультант
по нормоконтролю



Е. А. Хайбулова

Санкт-Петербург

2020

Peter the Great St.Petersburg Polytechnic University
Institute of Applied Mathematics and Mechanics
Higher School of Theoretical Mechanics

Work approved

Head of the Higher school

_____ A. M. Krivtsov

«__» _____ 20__ г.

GRADUATE QUALIFICATION WORK
«THERMO-MECHANICAL MODEL TO STUDY MICROSTRUCTURE
DURING ADDITIVE MANUFACTURING»

Subject 01.04.03 «Mechanics and Mathematical Modeling»
Educational program 01.04.03_02 «Mechanics and Mathematical Modeling
(international educational program)»

Submitted by
Student of group No.3640103/80201



N. Mohanan

Scientific advisor
Associate Professor, PhD.



V. A. Kuzkin

Regulatory
advisor

E. A. Khaibulova

Saint Petersburg

2020

ABSTRACT

Metal-Additive Manufacturing (AM) is a process where a metal wire or powder is molten using an energy source and deposited in layers to build parts. During this process, a melt pool is created which rapidly solidifies into a polycrystalline microstructure. When a new layer is deposited over the solidified microstructure, it undergoes a solid-state thermal cycling (SSTC) until the manufacturing process is completed. Understanding and controlling this process can help us to manufacture parts with desired material properties. The aim of this thesis is to lay foundation for a model capable of predicting the role of SSTC on a polycrystalline microstructure during an AM process. To that end, a novel fully coupled thermo-elastodynamic solver with the ability to predict the role of SSTC on an elastically heterogeneous and anisotropic microstructure is proposed. We recall the governing equations for an elastodynamics problem and then couple it with the governing equations for the heat conduction problem, to create a fully coupled Thermo-Elastodynamics(T-ED) model. With the addition of heterogeneous elasticity to the T-ED model, the SSTC response of a heterogeneous elastic microstructure is simulated. Towards the end, the T-ED model is employed to simulate the SSTC process to study the microstructure evolution. In conclusion, a validated heterogeneous T-ED model for simulating solid-state thermal loading over a heterogeneous microstructure is established.

Keywords: Linear Elastodynamics, Transient Heat Conduction, Thermo-elastodynamics, Finite Element modeling, Heterogeneous Elasticity, Solid-State Thermal Cycling, Microstructure Evolution.

CONTENTS

1.	Introduction.....	9
	1.1 Motivation.....	9
	1.2 Research Aim and Objectives.....	10
2	Introduction to FEniCS.....	11
	2.1 FEniCS Fundamentals.....	11
	2.2 FEniCS Programming.....	11
	2.2.1 Reformulation.....	11
	2.2.2 Program.....	12
	2.2.3 Post-Process.....	15
3	Linear Elasticity.....	16
	3.1 Governing Equations.....	16
	3.2 Finite Element Formulation and Solution Schemes.....	16
	3.2.1 Space Discretization.....	17
	3.2.2 Time Discretization.....	17
	3.3 Constitutive Equations.....	18
	3.4 Solution Methods.....	18
	3.5 Validation Test Case: Linear Elastostatics.....	20
	3.5.1 Problem Definition.....	20
	3.5.2 Solution and Inference.....	21
	3.6 Validation Test Case: Linear Elastodynamics.....	22
	3.6.1 Problem Definition.....	22
	3.6.2 Solution and Inference.....	23
4	Transient Heat Conduction.....	25
	4.1 Governing Equations.....	25
	4.2 Finite Element Formulation and Solution Schemes.....	25
	4.2.1 Space Discretization.....	26
	4.2.2 Time Discretization.....	26
	4.3 Solution Methods.....	27
	4.4 Validation Test Case: Transient Heat Conduction.....	29
	4.4.1 Problem Definition.....	29
	4.4.2 Solution and Inference.....	30

5	Coupled Thermo-Elastodynamics.....	32
	5.1 Governing Equations.....	32
	5.1.2 The coupled elastodynamic equation.....	33
	5.1.2 The coupled heat equation.....	33
	5.2 Finite Element Formulation and Solution Schemes.....	33
	5.2.1 Space Discretization.....	34
	5.2. 2 Time Discretization.....	34
	5.3 Solution Methods.....	35
	5.4 Validation Test Case: Thermo-Elastodynamics.....	36
	5.4.1 Problem Definition.....	37
	5.4.2 Solution and Inference.....	38
	5.5 Mesh Sensitivity and Timestep Study.....	40
6	Heterogeneous Elasticity.....	42
	6.1 Formulation.....	42
	6.2 Homogenization Problem.....	43
	6.3 Validation Test Case: EVP FFT Crystal Plasticity Model.....	44
	6.3.1 Problem Definition.....	44
	6.3.2 Solution and Inference.....	45
7	Heterogeneous Thermo-Elastodynamics.....	47
	7.1 Governing Equations.....	47
	7.1.1 The coupled displacement field.....	47
	7.1.2 The coupled temperature field.....	47
	7.2 Solution Methods.....	48
	7.2.1 Staggered Algorithm.....	48
	7.2.2 Interpolation Degree and Mesh Element.....	49
	7.2.3 Mesh and Timestep Selection.....	49
	7.3 Model Demonstration.....	49
	7.4 Sample Solution.....	50
8	Conclusion and Perspectives.....	53
9	Appendix.....	54
	I Notations	54
	II Implementation Code.....	55
	Bibliography.....	64

LIST OF TABLES

Table (3.1)	Material Properties and Assumptions: Elastostatics.....	21
Table (3.2)	Material Properties and Assumptions: Elastodynamics.....	13
Table (4.1)	Material Properties and Assumptions: Transient Heat Conduction.....	30
Table (5.1)	Material Properties and Assumptions: Thermo-Elastodynamics.....	38

LIST OF FIGURES

Figure (3.1)	Problem Geometry and Boundary Conditions.....	20
Figure (3.2)	Simplified geometry with tetrahedral mesh discretization.....	21
Figure (3.3)	Plot of σ_{yy} Vs Radial length Y.....	22
Figure (3.4)	σ_{yy} Contour Plot over domain.....	22
Figure (3.5)	Bar Problem Geometry and Boundary Conditions.....	22
Figure (3.6)	Plot of σ/ρ vs x/L at time $t = 0.6s$ over the length.....	23
Figure (4.1)	Problem Definition for Second Danilovskaya Problem.....	29
Figure (4.2)	Comparison of Solution to the Pure Heat Conduction Case of First Danilovskaya Problem.....	31
Figure (4.3)	Comparison of Solution to Pure Heat Conduction Case of the Second Danilovskaya Problem with $H = 0.5$ and 5.0	31
Figure (5.1)	Problem Definition for Second Danilovskaya Problem.....	36
Figure (5.2)	Comparison of Displacement u in Coupled Thermo-Elastodynamics of the Second Danilovskaya Problem with $H = 0.5$ and 5.0	39
Figure (5.3)	Comparison of Temperature T in Coupled Thermo-Elastodynamics of the Second Danilovskaya Problem with $H = 0.5$ and 5.0	39
Figure (5.4)	Mesh Sensitivity Study.....	40
Figure (5.5)	Timestep Study.....	41
Figure (6.1)	EBSD of 316L stainless steel by SLM AM technique: surfaces perpendicular (left) and parallel(right) to the build direction.....	42
Figure (6.2)	A Sample Map of (a) Euler angle ϕ ($^{\circ}$) and (b) ϵ_{2222} (MPa) represented over an RVE.....	43
Figure (6.3)	Map of ϵ_{2222} (a) EVP FFT (b) FEniCS FEM (MPa)	45

Figure (6.4)	Normal Strain values plotted over the loading direction along X_A and X_B	45
Figure (6.5)	Normal Stress values plotted over the loading direction along X_A and X_B	46
Figure (6.6)	ϵ_{11} values plotted over the cross-section of geometry under consideration (a)EVP FFT (b) FEniCS FEM for a microstr 100 grains...	46
Figure (6.7)	σ_{11} values plotted over the cross-section of geometry under consideration (a)EVP FFT (b) FEniCS FEM for a microstr 100 grains...	46
Figure (7.1)	A map of \mathfrak{C}_{1111} on a 4 Grain 16 Cube microstructure.....	50
Figure (7.2)	Temperature θ (°) Vs Time t (s) at surface $\mathbf{x} = \mathbf{0}$	51
Figure (7.3)	Displacement (\mathbf{u}_x) Vs Time t (s) at cube centroid.....	51
Figure (7.4)	Strain (ϵ_{11}) Vs Time t (s) at cube centroid.....	52
Figure (7.5)	Stress (σ_{11}) Vs Time t (s) at cube centroid.....	52

ACKNOWLEDGEMENTS

I wish to express my sincere and deep gratitude to my supervisors Prof. Manas Upadhyay and Prof. Vitaly Kuzkin for their continuous support and necessary guidance throughout my study and research. Their patience, motivation, enthusiasm, and immense knowledge encouraged me to complete my research and this thesis. I could not have imagined advancing this far without the immense support from my supervisors.

I am indebted to Prof. Eric Charkaluk and Dr. Matthias Rambauser, for their invaluable guidance throughout my research, helping me overcome roadblocks and pointing me in the appropriate directions.

I would like to pay my special regards to Prof. Irina Suslova, Prof. Alexander Nemov and Dr. Mikhail Barbenkov for their guidance in analytical problems and Dr. Jeremy Bleyer for his insights into solving computational models.

I would also like this opportunity to thank the Erasmus+ program for providing the necessary funding for this research to go smoothly; Laboratoire de Mécanique des Solides and Ecole Polytechnique for providing the necessary groundwork to make this research happen; and the Department of Theoretical Mechanics, SPBPU for providing me with the opportunity to undertake research abroad.

And finally, I wish to acknowledge the support and love of my friends and family, my father, Mohanan; my mother, Hema; my sister, Nikhitha; my best friends: dear, Dary; Umut and Bharath. They helped me keep going, providing confidence and encouragement. Without them, this endeavor would have been unsuccessful.

Nikhil Mohanan

CHAPTER 1: INTRODUCTION

Since inception in the 1990s, metal AM has revolutionized the modern manufacturing industry. Parts with complex geometries and specific design requirements can now be created through AM directly from a computer design without the need for expensive tooling or processes with little or no wastage of material.

Over time, modeling and simulations have taken an important role in advancing AM's capabilities by optimizing the process parameters influencing the properties of the resulting components. Because all metallic parts are polycrystalline at room temperature, the macroscopic behavior of these parts, depends highly on the effect of the manufacturing process on the microstructure, and thus, an understanding the behavior of its microstructure lays the groundwork improving the AM process.

1.1 MOTIVATION

Based on literature, much of the research into the study of microstructure evolution during additive manufacturing is focused on either its formation during the solidification process or on understanding the collective behavior of its microstructure once the manufacturing process is completed.

Microstructure simulation during the solidification process is roughly divided into macroscale (phase transformations), mesoscale (grain texture) and microcosmic scale (nucleation and grain growth). Methods such as Cellular Automata (CA) [1,2,3,4], Monte Carlo (MC) [5,6,7], Phase Field (PF) [8,9,10,11,12], and Molecular Dynamics(MD) [13,14] among others, help researchers to provide detailed morphologies analysis and dynamics of the microstructure formation. The research in this direction thus focus on the final microstructure as a product of the solidification process.

Once the solidification process results in a microstructure, the research direction shifts to property modeling. A clear understanding of the microstructure distribution, texture and morphologies is required to model macroscopic material properties. Based on different objectives, the following research are worth mentioning. Bronkhorst et al.[15-20], developed a coupled elasto-visco-plastic model to simulate a 2D single crystal based on the homogenization theory. Moulinec and Suquet [21-22] created an FFT based micromechanical model for crystal plasticity, developed into the VP-FFT Crystal plasticity solver by Lebensohn and Tome [23-26], to simulate multiaxial loading on multigrain microstructures. Bassani et al.[27,28], proposed a phenomenological-based texture evolution model (PBTE) focusing on flow rule calibrated by crystal plasticity. These researches are focused on the microstructure evolution after the manufacturing is completed.

Recently, Kürnsteiner, Jägle, Raabe et al.[29,30] and Rodrigues et al.[31], reported that the controlled intrinsic heat treatment (or SSTC) during the AM process, can extensively alter the strength of the printed material due to precipitation. The works of Yang et al.[32], Liu et al.[33], and Zhong et al.[34], support this by using selective laser melting to observing variations in martensite formation and grain growth on subsequent layers. These are evidences to the work of Zheng et al.[35,36] which concluded that the heating and cooling cycles during AM play a critical role in the evolution of the microstructure.

In addition, many other researchers have concluded that the thermal cycling of metallic parts affects the microstructure and in turn, the strength of the material. In many cases, precipitation, and phase transformations [37-41] have been found to occur during cyclic heating below the solidus temperature. Furthermore, Kürnsteiner, Jägle, Raabe et al.[29,30] and Zheng[36] have also reported experimental observations of grains coarsening as a result of the SSTC during AM.

Considering the above literature, it is clear that a general micromechanical model predicting the T-ED behavior of SSTC during AM for polycrystalline microstructure is nonexistent, and thus our motivation to create a fully coupled T-ED polycrystalline model that enables us to model and to understand the effect of SSTC process on the microstructure came to light.

1.2 RESEARCH AIM AND OBJECTIVES

The aim of this thesis is to support the development of a general micromechanical solver to understand and study the evolution of microstructure due to SSTC process during additive manufacturing.

As the foundational research, towards modeling microstructure evolution during SSTC, the following procedural objectives have been selected:

- i. Formulate and validate the linear elasticity model
 - Include formulation for anisotropic static and dynamic elasticity
- ii. Formulate and validate the transient heat conduction model
 - Include formulation for orthotropic heat conduction
- iii. Formulate and validate the thermo-elastodynamics model
 - Include formulation for fully coupled and weakly coupled problem in addition to thermo-elastostatics and thermo-elastodynamics.
- iv. Formulate and validate the heterogeneous elasticity model
 - Include the homogenization problem for periodic heterogeneous RVE
- v. Build the coupled thermo-elastodynamics model
 - Model a single-cycle, SSTC process conforming to additive manufacturing

The entire modeling approach is aimed at avoiding assumptions which are not mandatory while considering fully anisotropic material constants. Thus, a validation case for each objective is a requirement.

CHAPTER 2: INTRODUCTION TO FENICS PROJECT

FEniCS Project Library is a research software library aimed to simplify and quickly transform scientific models into efficient finite element code. It is an intuitive, easy, flexible and efficient tool for solving partial differential equations using the finite element method. FEniCS can be programmed both in C++ and Python interfaces

Developed since 2003, FEniCS Project presently unites researchers and research institutions from across the world to solve problems in fluid mechanics, solid mechanics, thermodynamics, electromagnetics, and geophysics.

2.1 FENICS FUNDAMENTALS

In this chapter we consider the solution process involved of solving PDEs using FEniCS. Let us take the following Poisson Problem as an example to explain the modeling-solution procedure.

Consider the following boundary value problem:

$$-\nabla^2 \mathbf{u}(x) = \mathbf{f}(x), \quad x \text{ in } \Omega \quad (2.1)$$

$$\mathbf{u}(x) = \mathbf{u}_D(x), \quad x \text{ on } \partial\Omega_D \quad (2.2)$$

$$\frac{\partial \mathbf{u}(x)}{\partial \mathbf{n}} = T_S(x), \quad x \text{ on } \partial\Omega_S \quad (2.3)$$

here, $\mathbf{u} = \mathbf{u}(x)$ is the unknown field, $\mathbf{f} = \mathbf{f}(x)$ is the known field, ∇^2 is the Laplacian operator, Ω is the spatial domain, $\partial\Omega$ is the specified boundary of the domain.

We are now required to complete the following steps:

1. Reformulation:
To reformulate the Partial Differential Equation from its strong form to a finite element variational weak form.
2. Program:
To write a program in Python to solve the formulated variational problem, using FEniCS.
3. Post-Process:
To create visual representation and to calculate specific results

2.2 FENICS PROGRAMMING

2.2.1 REFORMULATION

The simplest method of transforming a PDE in Strong form to a Weak variational problem is to multiply it by a Test Function \mathbf{v} and then integrate it over the domain Ω .

$$-\int_{\Omega} (\nabla^2 \mathbf{u}) \cdot \mathbf{v} \, dx = \int_{\Omega} \mathbf{f} \cdot \mathbf{v} \, dx \quad (2.4)$$

here, dx denotes the differential element for integrating over the entire domain Ω ,
 ds denote the differential element for integrating over the boundary of the domain $\partial\Omega$.

To keep the order of the derivatives of u and v as small as possible, we remove the divergence by using integration by parts:

$$-\int_{\Omega} (\nabla^2 u) \cdot v \, dx = \int_{\Omega} \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \frac{\partial u}{\partial n} \cdot v \, ds \quad (2.5)$$

where, $\partial u / \partial n = \nabla u \cdot n$ is the derivative of u in the outward normal direction n .

Finally, when we combine (2.4) and (2.5) we acquire the weak form or the variational form of the given boundary value problem.

$$\int_{\Omega} \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \frac{\partial u}{\partial n} \cdot v \, ds = \int_{\Omega} f \cdot v \, dx \quad (2.6)$$

Since, we require that this equation is true for all test function v in a suitable test function space \hat{V} , we will obtain a uniquely defined problem that determines the solution u in the trial function space V

Thus, the complete definition of the problem is as follows:

$$\int_{\partial\Omega} \nabla u \cdot \nabla v \, dx - \int_{\partial\Omega} \frac{\partial u}{\partial n} \cdot v \, ds = \int_{\Omega} f \cdot v \, dx \quad \forall v \in \hat{V} \quad (2.7)$$

For linear problems, we specify the Equation (2.7) in the following form:

$$a(u, v) = L(v) \quad (2.8)$$

where, $a(u, v)$ is known as the bilinear form and $L(v)$ is the linear form.

And for nonlinear problems, we specify the Equation (2.7) in the following form:

$$F(u, v) = 0 \quad (2.9)$$

where, $F(u, v)$ is the combined bilinear - linear form function equated to zero.

2.2.2 PROGRAM

Let us now create a simple program using FEniCS to solve the above problem.

a. Specify the domain and create the mesh

Let us consider the domain to be a Cube of 8mm x 8mm x 8mm, discretized into 10 elements in all three directions and centered at (0,0,0)

```
mesh= BoxMesh(Point(-4.0,-4.0,-4.0),Point( 4.0,4.0,4.0),10,10,10)
```

where, $BoxMesh(A,B,C,D,E)$ function creates a Cuboid with two extreme vertices placed at A and B and number of elements specified by C, D, E in X, Y, Z

$Point(X_1, Y_1, Z_1)$ with the location (X_1, Y_1, Z_1)

b. Specifying Functional Space, Constants and Test-Trial Functions

Let us create a scalar valued function space V for: the unknown variable function, the trial function, the test function and the constants. Assign T for Neumann boundary condition as 10.0 and F for the prescribed volumetric field as 0.0

```
V = FunctionSpace(mesh, "CG", 1)
T = Constant(10.0)
F = Constant(0.0)
u = TrialFunction(V)
v = TestFunction(V)
```

where, *mesh* is the previously created mesh for the domain

“CG” is the specification for Lagrange/Continuous-Galerkin element specification
 1, is the interpolation degree. For, *P1* Elements degree=1, *P2* Elements degree=2.
Constant(A), specifies the constant value of A assigned for FEniCS calculations.
TrialFunction(V) specifies the TrialFunction variable over Function Space V
TestFunction(V) specifies the TestFunction variable over Function Space V

c. Specifying Boundary Locations

Let us now define different boundary locations: Let there be a Dirichlet boundary condition specified over the boundary at ($X = -4.0$) and the Neumann boundary condition specified over the boundary at ($X = 4.0$). This can be done using the *def* command.

```
def XMinusBoundary(x,on_boundary):
    return (near(x[0],-4.0,EPS_DOLFIN) and on_boundary)
def XPlusBoundary(x,on_boundary):
    return (near(x[0],4.0,EPS_DOLFIN) and on_boundary)
```

where, *x[i]* specifies the direction of specification, $i = (0, 1, 2)$ for X, Y, Z direction

near(A,B,C) defines a tolerance range for node selection, A is the direction, B is the location, while C is the tolerance value. EPS_DOLFIN is the default FEniCS tolerance value set by the library

on_boundary is used to constrain the selection to only boundary nodes

d. Specifying Boundary Conditions and Form Function

Let us now specify the Dirichlet and Neumann Boundary conditions along with the Variational Form Function

```
bc = DirichletBC(V,Constant(0.0),XMinusBoundary)

boundary_subdomain = MeshFunction("size_t", mesh, mesh.topology().dim - 1)
boundary_subdomain.set_all(0)
AutoSubDomain(XPlusBoundary).mark(boundary_subdomain, 1)
dss = ds(subdomain_data = boundary_subdomain)
```

where, DirichletBC(A,B,C) is the dirichlet boundary condition applied over the function space V, with the specified value as B, and at predefined location C

The selection for the integrating surface *dss* is rather a complex process-

- i. Select all the nodes using a *MeshFunction* as *boundary_subdomain*
- ii. Initialize the value for the selected *boundary_subdomain* as 0
- iii. Mark the *XPlusBoundary* onto the *boundary_subdomain* as 1
- iv. Assign the marked locations to the variable *dss*, and using *dss(i)* we can call any previously marked subdomain surfaces into the form

e. Creating the Variational Form

We create the variational form function F as defined in the variational formulation.

$$F = \text{grad}(u) * \text{grad}(v) * dx - T * v * dss(1) - F * v * dx$$

where, *u* is the trial function specified over the function space V

v is the test function specified over the function space V

grad(u) specifies the gradient operator over the field *u*

dx specifies that the integration is over the entire finite elements of the domain

dss(i) specifies that the integration is over the marked subdomain boundary

Based on the solver type one can use $F(u,v) == 0$ for nonlinear solver, or $a(u,v) == L(v)$ bilinear-linear form for the linear solver. FEniCS uses the following to create a and L

```
a = lhs(F)
L = rhs(F)
```

where, $lhs(F)$ collects the functions in bilinear form (u,v)
 $rhs(F)$ collects the functions in linear form (v)

f. Creating the Solver

To create the linear *solver*, one must first specify the reassign u as a *Function* of the space V , create the *problem*, and then proceed to *solver* creation. This is done by the following steps:

```
u = Function(V)
problem = LinearVariationalProblem(a, L, u, bc)
solver = LinearVariationalSolver(problem)
```

where, $LinearVariationalProblem(a, L, u, bc)$ function is called to create the problem with a as the bilinear form, L as the linear form, u as the solve variable and bc as the list of Dirichlet boundary conditions

In the case of nonlinear *solver*, there are some additional steps involved:

```
u = Function(V)
u_trial = TrialFunction(V)
...
J = derivative(F,u,u_trial)
problem = NonlinearVariationalProblem(F, u, bc, J, ffc_options)
solver = NonlinearVariationalSolver(problem)
```

here, u is initially created as a *Function* of V while u_trial is created as the *TrialFunction*.
 J is calculated as the jacobian derivative of the form function F with respect to u and u_trial .
The problem is specified with similar parameters as the linear solver, using F instead of a,L and an additional option for *form_compiler_parameters* as *ffc_options*, it contains detail for the compiler, such as quadrature degree etc.

g. Initializing Variables

In case the problem has an initial field variable, u or u_n , with an initial field value $Constant(1.0)$, we do this by the following step:

```
u_n = project(Constant(1.0), V)
```

here, we project the constant value of 1.0 over the function space V using the $project(A,V)$ function. In addition, one can also use the $interpolate(A,V)$ as well

h. Defining Outputs

We can define the output folder and file using the following commands

```
File = XDMFFile("Foldername/Filename.xdmf")
File.parameters["flush_output"] = True
File.parameters["function_share_mesh"] = True

File.write(u,t)
```

where, the file is created using the function `XDMFFile("Foldername/Filename.xdmf")` with the Folder and Filenames in the specified format. The `.xdmf` extension is used to write the file, which can easily be read using Paraview

`"flush_output" = True`, ensures that after each function call the file is written to instead of a full buffer write.

`"function_share_mesh" = True`, is used when the mesh remains common throughout the program. If the mesh changes at any point, this parameter must be left as False and a new mesh data will be created for each timestep (t). In case only one step is solved, one may assign ($t = 0.0$)

u is the output field, and t is the specified cumulative time for the step

i. Solve Function

Finally, the solve function may be called by invoking the following command:

`solver.solve()`

In an additional note, if one were to make the Dirichlet boundary condition time dependent, then one may have to re-form the problem and solver, before executing the solve function.

2.2.3 POST-PROCESS

As soon as the above finite element program is solved, one is provided with a filename.xdmf in the /Foldername/ location. This can be postprocessed to provide meaningful results using the opensource Paraview software. One can find additional material at [42] or [43] to post process and acquire specific results.

CHAPTER 3: LINEAR ELASTICITY

In this chapter, we introduce the necessary fundamentals of linear elasticity, in specific the Linear Elastodynamics (ED). For a more comprehensive review, one may refer to Bonet and Wood [44], Lemaître and Chaboche [45] or Zeinkiewicz and Taylor [46]. We begin by presenting the balance equations for structural mechanics in the strong and weak forms, then moving to discretization schemes for finite element method (FEM) and solution techniques involved.

3.1 GOVERNING EQUATIONS

The displacement field is assumed to be governed by the local material form of the linear momentum balance from newton's second law as:

$$\text{div } \boldsymbol{\sigma} + \mathbf{f} = \rho \ddot{\mathbf{u}}, \quad \text{in } \Omega \quad (3.1)$$

here, \mathbf{u} is the displacement vector field,
 \mathbf{f} is the volumetric vectoral body force,
 $\boldsymbol{\sigma}$ is the Cauchy stress tensor.

In the case of static elasticity, this equation reduces to:

$$\text{div } \boldsymbol{\sigma} + \mathbf{f} = 0, \quad \text{in } \Omega \quad (3.2)$$

We will consider the complete dynamic equation for our formulations, with a side note to the static equation.

3.2 FINITE ELEMENT FORMULATIONS AND SOLUTION SCHEMES

In this section we define the initial boundary value problem (IBVP) using this governing equation for dynamic linear elasticity (3.1), combining it with kinematic relations, applying a constitutive model, and specifying a set of initial and boundary conditions, respectively.

The boundary $\partial\Omega$ will be divided into Dirichlet and Neumann conditions applied on $\partial\Omega_D$ and $\partial\Omega_S$, respectively. On the Dirichlet boundary $\partial\Omega_D$, the displacements are specified and on the Neumann boundary $\partial\Omega_S$, the traction vector is specified. This leads to the following conditions:

$$\mathbf{u} = \mathbf{u}_D, \quad \text{on } \partial\Omega_D \quad (3.3)$$

$$\boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{T}_S, \quad \text{on } \partial\Omega_S \quad (3.4)$$

which needs to be satisfied. It is also worth noting that the boundary $\partial\Omega$ is divided into disjoint partitions;

$$\partial\Omega_D \cup \partial\Omega_S \in \partial\Omega, \quad \partial\Omega_D \cap \partial\Omega_S = 0 \quad (3.5)$$

In addition, the initial conditions on the displacements along with the initial velocity $\dot{\mathbf{u}}$ at $t = 0$ has to be specified:

$$\mathbf{u} = \mathbf{u}(x, 0) = \mathbf{u}_0, \quad \text{on } \Omega \quad (3.6)$$

$$\dot{\mathbf{u}} = \dot{\mathbf{u}}(x, 0) = \dot{\mathbf{u}}_0, \quad \text{on } \Omega \quad (3.7)$$

The above representation of the PDEs along with the governing equation defines the strong form of the IBVP definition, and this is required to be satisfied in a pointwise strong solution. The analytical solution to this type of IBVP problems is only possible for only simple cases requiring small deformation assumption and simple geometries etc.

3.2.1 SPACE DISCRETIZATION

The foundation of FEM is the conversion of the problem definition from strong form to the weak or variational form. In distinction, the weak form satisfies the equations on an integral basis. The derivation of the weak form involves the application of the virtual work principle, by multiplying the equation with a weighting test function and then integrating the system over the entire domain Ω .

In the problem definition, let us consider \mathbf{u} as the trial function and \mathbf{v}_u as the test function.

The strong form of the equation transforms to:

$$\rho \int_{\Omega} \ddot{\mathbf{u}} \cdot \mathbf{v}_u dx = \int_{\Omega} \text{div } \boldsymbol{\sigma} \cdot \mathbf{v}_u dx + \int_{\Omega} \mathbf{f} \cdot \mathbf{v}_u dx \quad (3.8)$$

Applying integration by parts and rearranging we have the weak variational form:

$$\rho \int_{\Omega} \ddot{\mathbf{u}} \cdot \mathbf{v}_u dx = \int_{\partial\Omega_s} \mathbf{T}_S \cdot \mathbf{v}_u ds - \int_{\Omega} \boldsymbol{\sigma} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f} \cdot \mathbf{v}_u dx \quad (3.9)$$

here, dx denotes the differential element for integrating over the entire domain Ω ,
 ds denote the differential element for integrating over the boundary of the domain $\partial\Omega_s$.

3.2.2 TIME DISCRETIZATION

The formulated equation (3.9) is still continuous with respect to time, and hence we are required to apply a suitable time integration to acquire the fully discretized structural equation. With respect to the highly fluctuating nature of the problem, we use the trapezoidal rule [47] as defined as follows for displacement and then velocity:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \frac{\Delta t}{2} (\dot{\mathbf{u}}_n + \dot{\mathbf{u}}_{n+1}) \quad (3.10)$$

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{\Delta t}{2} (\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1}) \quad (3.11)$$

Combining Equation (3.10) and (3.11), and then rearranging for $\ddot{\mathbf{u}}_{n+1}$

$$\ddot{\mathbf{u}}_{n+1} = \frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \quad (3.12)$$

here, Δt is the timestep of the discretization,
 $n/n+1$ refers to the previous/current step, respectively.

Now substituting Equation (3.12) in Equation (3.9), we acquire the fully discretized weak variational form of the problem:

$$\begin{aligned} \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u dx \\ = \int_{\partial\Omega_s} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u ds - \int_{\Omega} \boldsymbol{\sigma}_{n+1} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u dx \end{aligned} \quad (3.13)$$

In the case of static formulation, the right-hand side of equation (3.13) becomes zero leading to:

$$\int_{\partial\Omega_s} [\mathbf{T}_s]_{n+1} \cdot \mathbf{v}_u \, ds - \int_{\Omega} \boldsymbol{\sigma}_{n+1} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u \, dx = \mathbf{0} \quad (3.14)$$

Thus, we have reviewed the necessary governing equations for elasticity.

3.3 CONSTITUTIVE EQUATIONS

Now, with the variational form defined, what remains is to discuss the definition of the stress with reference to the above structural problem. Hence the following constitutive laws for anisotropic elasticity is presented.

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon} \quad (3.15)$$

here, \mathbb{C} represents the fourth order stiffness tensor,
 $\boldsymbol{\varepsilon}$ represents the linearized elastic strain tensor,

Further, the elastic strain tensor is defined as:

$$\boldsymbol{\varepsilon} = \frac{1}{2} (\nabla \mathbf{u} + (\nabla \mathbf{u})^T) \quad (3.16)$$

In the simplified case of isotropy,

$$\boldsymbol{\sigma} = \lambda \, \text{tr}(\boldsymbol{\varepsilon}) \, \mathbf{I} + 2 \, \mu \, \boldsymbol{\varepsilon} \quad (3.17)$$

where, λ and μ are Lamé Constants

Since, we focus on anisotropic material model for heterogeneous, we choose (3.15)

3.4 SOLUTION METHODS

Combining the defined constitutive equation with the variational form, we have completely defined the given IBVP and can use either the direct solver or the Newton Raphson iterative solver. The Newton Raphson solver is more efficient and robust to solve linear systems. In the case of large FE Models (with considerably large number of DOFs) the solution process of a given step is the most computationally expensive aspect of the complete solution procedure.

Since we are using the trapezoidal rule, we are required to calculate the acceleration and velocity at each timestep before the next step begins:

Step 1: Solve for displacement field (\mathbf{u}_{n+1})

$$\begin{aligned} \int_{\partial\Omega_S} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u \, ds - \int_{\Omega} \boldsymbol{\sigma}_{n+1} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u \, dx \\ = \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u \, dx \end{aligned} \quad (3.13)$$

With Dirichlet boundary condition: $\mathbf{u} = \mathbf{u}_D$, on $\partial\Omega_D$ (3.3)

Step 2: Solve for acceleration field ($\ddot{\mathbf{u}}_{n+1}$)

$$\rho \int_{\Omega} \ddot{\mathbf{u}}_{n+1} \cdot \mathbf{v}_u \, dx = \int_{\partial\Omega_S} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u \, ds - \int_{\Omega} \boldsymbol{\sigma}_{n+1} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u \, dx \quad (3.9)$$

Step 3: Predict the velocity field ($\dot{\mathbf{u}}_{n+1}$)

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{\Delta t}{2} (\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1}) \quad (3.11)$$

Step 4: Assign Previous Timestep Variables

$$\begin{aligned} \text{Displacement:} \quad & \mathbf{u}_n = \mathbf{u}_{n+1} \\ \text{Velocity:} \quad & \dot{\mathbf{u}}_n = \dot{\mathbf{u}}_{n+1} \\ \text{Acceleration:} \quad & \ddot{\mathbf{u}}_n = \ddot{\mathbf{u}}_{n+1} \end{aligned} \quad (3.18)$$

Step 5: Continue Time Iteration

Note: In the case of the static governing equation, we use a direct solver.

3.5 VALIDATION TEST CASE: LINEAR ELASTO-STATICS (ES)

To validate the linear Elasto-Static (ES) implementation for static elasticity, let us consider the Kirsch Problem. A more detailed explanation to the problem and analytical solution referenced in this verification problem can be found in Kulesh et al.,[48].

3.5.1 PROBLEM DEFINITION

Consider a two-dimensional square section of width $L = 2\text{mm}$ with a circular hole of radius $R = 0.2\text{ mm}$ at its geometric center. Let a traction of $T_s = 1\text{ MPa}$ be applied at $y = 1.0\text{ mm}$ and $y = -1.0\text{ mm}$ boundaries. The body is initially at rest. The problem is depicted in Figure (3.1):

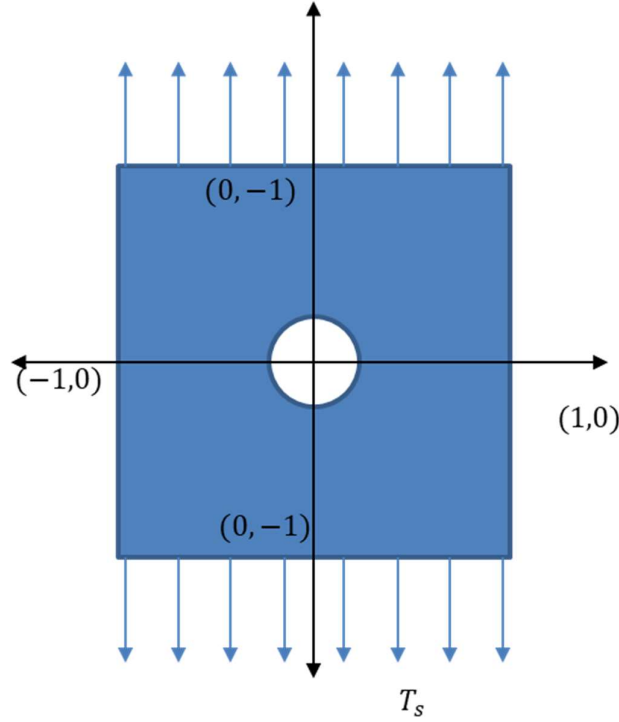


Figure (3.1): Problem Geometry and Boundary Conditions

Therefore, the complete initial boundary value is defined as:

$$\text{div } \boldsymbol{\sigma} = 0, \quad \text{in } \Omega \quad (3.2)$$

With initial condition:

$$\mathbf{u}((x, y), 0) = 0 \quad (3.20)$$

The boundary condition:

$$\begin{aligned} \boldsymbol{\sigma} \cdot \mathbf{n} &= \mathbf{T}_s((x, +1.0), t) = 1.0 \text{ MPa/mm}^2 \text{ in Y direction on } \partial\Omega_{S1} \\ \boldsymbol{\sigma} \cdot \mathbf{n} &= \mathbf{T}_s((x, -1.0), t) = -1.0 \text{ MPa/mm}^2 \text{ in -Y direction on } \partial\Omega_{S2} \end{aligned} \quad (3.21)$$

The constitutive law:

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon} \quad (3.15)$$

And the strain $\boldsymbol{\varepsilon}$ is defined as:

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) \quad (3.16)$$

For simplicity let us consider a symmetric quarter section as shown in Figure (3.2):

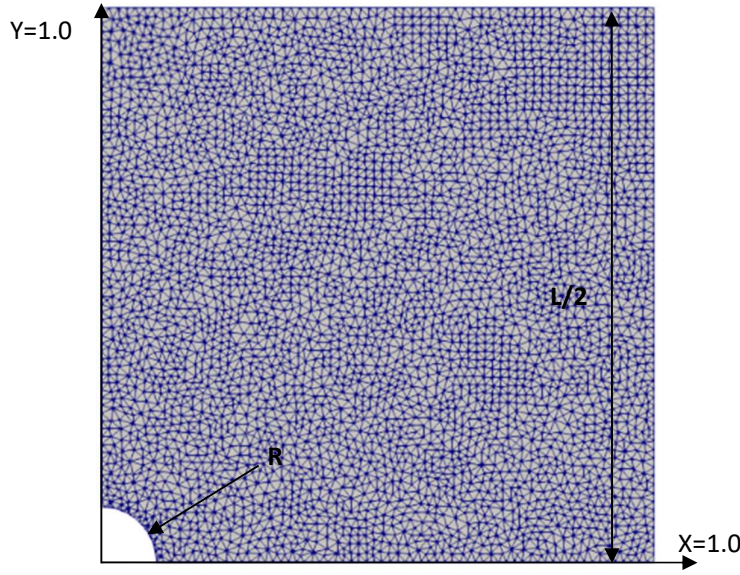


Figure (3.2) Simplified geometry with tetrahedral mesh discretization

Thus, we now have the following boundary condition:

$$\begin{aligned} \mathbf{u}(x, 0.0, t) \text{ is constrained in } X, \quad & \text{in } \Omega \\ \mathbf{u}(0.0, y, t) \text{ is constrained in } Y, \quad & \text{in } \Omega \end{aligned} \quad (3.22)$$

The variational formulation after applying the finite element discretization and time discretization according to the procedure described in Section 3.2, we have:

$$\int_{\partial\Omega_{S1}} \mathbf{T}_s \cdot \mathbf{v}_u \cdot d\mathbf{s}(1) - \int_{\Omega} \nabla \mathbf{u} : \nabla \mathbf{v}_u \cdot d\mathbf{x} = 0 \quad (3.14)$$

Now, we follow the direct solution approach with linear solver for solving the above equation.

Variable	Assumption
\mathbb{C}	\mathbf{I} Identity Tensor
$\mathbf{T}_s((x, +1.0), t)$	1.0 MPa/mm ²
$\mathbf{u}((x, 0.0), t)$	(0.0, free)
$\mathbf{u}((0.0, y), t)$	(free, 0.0)
L/2	1.0 mm
R	0.1 mm

Table 3.1 Material Properties and Assumptions

3.5.2 SOLUTION AND INFERENCES

The numerical computation was achieved using a uniform mesh of density 50 Lagrange P2 elements. The results are compared with analytical results from [48]. Figure (3.3) shows the results of normal stress σ_{yy} against radial length in Y. Figure (3.3) shows the normal stress σ_{yy} variation over the domain.

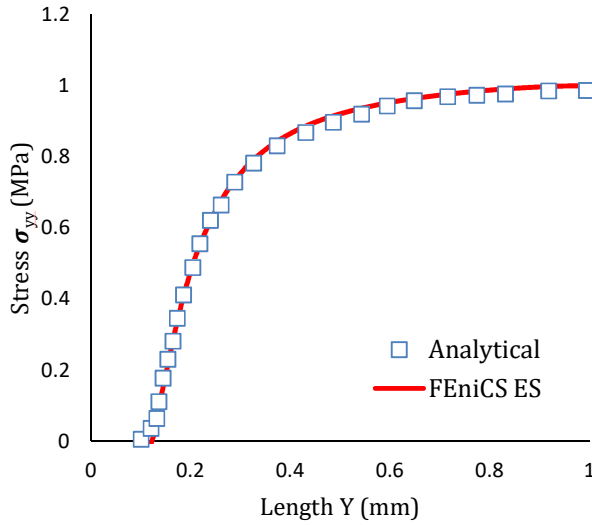


Figure (3.3) Plot of σ_{yy} Vs radial length Y

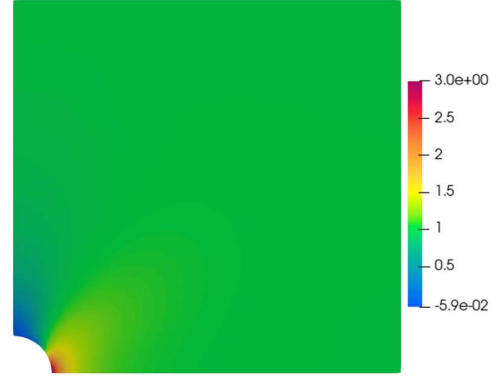


Figure (3.4) σ_{yy} Contour Plot over domain

As we can see from figure (3.3), with the numerical solution in FEniCS matching the exact analytical solution. And thus, we now consider our ES model to be verified.

3.6 VALIDATION TEST CASE: LINEAR ELASTO-DYNAMICS (ED)

To validate the linear Elastodynamic (ED) implementation, we shall compare our results to analytical solution for ED, as described by Eran Grosu et al.,[49]. A more detailed explanation and solution to analytical results can be found Idesman et al., [50].

3.6.1 PROBLEM DEFINITION

Consider a one-dimensional wave propagation on a bar of length L having a constant stiffness tensor \mathbb{C} and thus a constant speed of sound c_0 within the medium Ω . The bar is fully constrained at $x = 0$ and a constant pressure p is applied at $x = L$. The bar is assumed to be initially at rest.

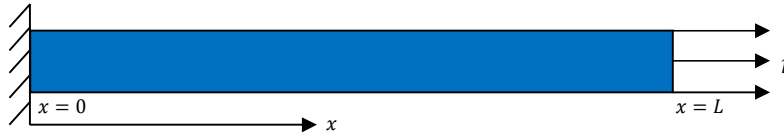


Figure (3.5): Bar Problem Geometry and Boundary Conditions

Therefore, the complete initial boundary value is defined as:

$$\rho \ddot{\mathbf{u}} = \text{div } \boldsymbol{\sigma}, \quad \text{in } \Omega \quad (3.26)$$

With initial condition:

$$\begin{aligned} \mathbf{u}(x, 0) &= 0 \\ \dot{\mathbf{u}}(x, 0) &= 0 \end{aligned} \quad (3.27)$$

And the boundary condition:

$$\begin{aligned} \boldsymbol{\sigma} \cdot \mathbf{n} &= \mathbf{T}_s(L, t) = p \\ \mathbf{u}(0, t) &= 0 \end{aligned} \quad (3.28)$$

And the constitutive law:

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon} \quad (3.18)$$

And the strain $\boldsymbol{\varepsilon}$ is defined as:

$$\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) \quad (3.19)$$

For simplification of the analytical solution, we assume $\mathbb{C} = \mathbf{I}$ (the identity tensor), length $L = 1$, the density $\rho = 1$ and the deformation gradient is symmetric. Thus, the equation reduces to:

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} = \text{div}(\nabla \mathbf{u}), \quad \text{in } \Omega \quad (3.29)$$

The variational formulation after applying the finite element discretization and time discretization according to the procedure described in Section 3.2, we have:

$$\int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u \, dx = \int_{\partial\Omega_s} \mathbf{p} \cdot \mathbf{v}_u \, ds - \int_{\Omega} \nabla \mathbf{u} : \nabla \mathbf{v}_u \, dx \quad (3.16)$$

Now, we follow the staggered approach for the Trapezoidal Rule formulation, also using nonlinear solver for the primary solution for displacement as explained in Section 3.4.

Variable	Assumption
\mathbb{C}	\mathbf{I} Identity Tensor
$T_s(L, t)$	1.0 MPa/mm ²
$\mathbf{u}(0, t)$	(0.0)
\mathbf{L}	1.0 mm
dt	0.006 s

Table 3.2 Material Properties and Assumptions

3.6.2 SOLUTION AND INFERENCES

The numerical computation was achieved using a uniform mesh of 100 linear Lagrange P2 elements for 5 different timesteps, corresponding to $CFL = 1.0, 0.5, 0.1, 0.05$ and 0.01 corresponding to the time step $dt = 0.01, 0.05, 0.001, 0.005$ and 0.0001 respectively. The results are compared with analytical results from Eran Grosu et al.,[49]. Figure (3.6) shows the results of normal stress per density along the rod versus at displacement over length at $t = \frac{0.6L}{c_0} = 0.6 \text{ s}$.

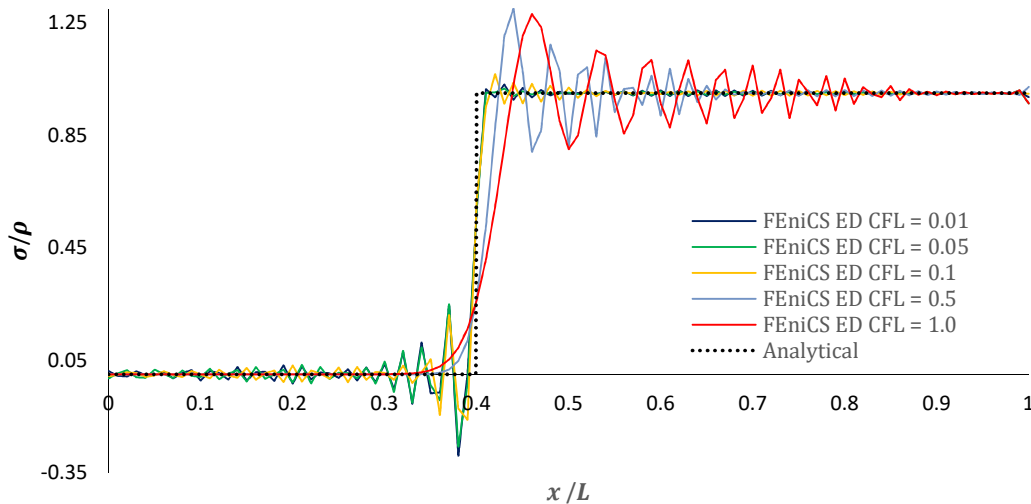


Figure (3.6) Plot of $\frac{\sigma}{\rho}$ vs $\frac{x}{L}$ at time $t = 0.6 \text{ s}$ over the length.

As pointed out by Eran Grosu et al.[49], the trapezoidal rule applied to the ED equation sometimes leads to spurious oscillations. At relatively large timesteps (corresponding to CFL number > 1.0), these oscillations travel ahead of the wave front, but for smaller timesteps (corresponding to CFL number < 1.0) it precedes it. The occurrence of these oscillations is due the fact that the trapezoidal rule lacks algorithmic damping. In the case of thermo-elastodynamics, these oscillations must be behind the traveling wave front in-order to fully understand the displacement wave propagation with two-way full field coupling.

From the figure (3.6), we see that for smaller timesteps corresponding to CFL = 0.05-0.01 or lower, we can reduce the oscillations. Thus, the results are in good agreement with the exact solution and we have verified the ED model's validity. Although, it is advisable to conduct a mesh convergence and timestep study on the test problems, once the realistic material properties are applied.

CHAPTER 4: TRANSIENT HEAT CONDUCTION

For the development of the fully coupled micromechanical model, we are required to consider the heat equation governing the temperature field. In this chapter, we formulate variational form the governing equations required to model the temperature field using FEM. Lemaitre and Chaboche[45], MA Biot[51] and Marc Bonnet[52] have described in detail about the foundations, derivations, and about existing finite element implementations. We now proceed to acquire the fully discretized heat conduction equation, in a similar fashion to that for Linear Elasticity, described in Chapter 3.

4.1 GOVERNING EQUATIONS

The temperature field is assumed to be governed by the following strong form equation derived from the first and second laws of thermodynamics [45]:

$$-div \mathbf{q} + r = \rho C_v \dot{\theta}, \quad in \Omega \quad (4.1)$$

here, θ is the temperature field,
 r is the volumetric heat generation rate,
 \mathbf{q} is the heat flux density.

The Fourier law shows that the local heat flux density \mathbf{q} is equal to the product of the thermal conductivity \mathbf{k} and the negative local temperature gradient, $-\nabla\theta$.

In differential form we have,

$$\mathbf{q} = -\mathbf{k} \cdot \nabla\theta \quad (4.2)$$

$$\mathbf{q} = -\mathbf{k} \cdot \left(\frac{\partial\theta}{\partial x} + \frac{\partial\theta}{\partial y} + \frac{\partial\theta}{\partial z} \right) \quad (4.3)$$

where, \mathbf{k} is the orthotropic thermal conductivity tensor

In addition to the orthotropic Fourier's law, if Duhamel's law applicable in the derivation, is restricted to thermally isotropic behavior, the conductivity tensor reduces to $\mathbf{k} = k \mathbf{I}$. Nevertheless, for the present work Fourier's law as stated in (4.2) is best suited to model heterogeneous behavior and is considered in this thesis.

4.2 FINITE ELEMENT FORMULATIONS AND SOLUTION SCHEMES

The IBVP definition for the thermal field is governed by equations (4.1) and (4.3), combined with kinematic relations, in addition to a set of initial and boundary conditions. The boundary $\partial\Omega$ will be divided into Dirichlet, Neumann and Robin conditions applied on $\partial\Omega_D$, $\partial\Omega_S$, and $\partial\Omega_H$ respectively. On the Dirichlet boundary $\partial\Omega_D$: the temperatures are specified (θ_D), on the Neumann boundary $\partial\Omega_S$: the normal heat flux density vector (Q_S) is specified, and on the Robin Boundary $\partial\Omega_H$: a heat flux according to the newton's law of cooling also called as convective-heat boundary condition(h). This leads to the following conditions to be satisfied:

$$\theta = \theta_D, \quad on \partial\Omega_D \quad (4.4)$$

$$\mathbf{q} \cdot \mathbf{n} = Q_S, \quad on \partial\Omega_S \quad (4.5)$$

$$\mathbf{q} \cdot \mathbf{n} = h(\theta - \theta_\infty), \quad on \partial\Omega_H \quad (4.6)$$

here, θ is the current boundary temperature
 θ_∞ is the ambient temperature or the temperature at infinity.

It is also worth noting that the boundary $\partial\Omega$ is divided into disjoint partitions;

$$\partial\Omega_D \cup \partial\Omega_S \cup \partial\Omega_{SH} \in \partial\Omega, \quad \begin{aligned} \partial\Omega_D \cap \partial\Omega_S &= 0, & \partial\Omega_D \cap \partial\Omega_H &= 0, \\ \partial\Omega_H \cap \partial\Omega_S &= 0 \end{aligned} \quad (4.7)$$

In addition, the initial conditions on the temperature θ_0 along with the initial temperature rate $\dot{\theta}_0$ at $t = 0$ has to be specified:

$$\theta = \theta(x, t = 0) = \theta_0, \quad \text{on } \Omega \quad (4.8)$$

$$\dot{\theta} = \dot{\theta}(x, t = 0) = \dot{\theta}_0, \quad \text{on } \Omega \quad (4.9)$$

Note: The initialization of $\dot{\theta}$ is depends on the specific time discretization scheme used.

4.2.1 SPACE DISCRETIZATION

The finite element method is applied to get the spatial discretization of the heat equation (4.1). In doing so, we acquire the finite variational form. The following formulation is similar in nature to the steps in the linear elasticity equations in section 3.2.1. We begin by taking the equation (4.1), multiply it with a suitable weighting function and integrate it over the domain Ω .

Let us consider θ as the trial function and v_T as the scalar test function corresponding to the temperature field. The strong form transforms to:

$$-\int_{\Omega} \text{div } \mathbf{q} \cdot v_T \, dx + \int_{\Omega} r \cdot v_T \, dx = \rho C_v \int_{\Omega} \dot{\theta} \cdot v_T \, dx \quad (4.10)$$

Applying the integration by parts on the divergence, and assuming separate Neumann and Robin conditions on the surface on equation (4.10), we have:

$$\begin{aligned} \rho C_v \int_{\Omega} \dot{\theta} \cdot v_T \, dx = & - \int_{\partial\Omega_S} Q_S \cdot v_T \, ds(S) - \int_{\partial\Omega_H} h (\theta - \theta_{\infty}) v_T \, ds(H) \\ & + \int_{\Omega} \mathbf{q} \cdot \nabla v_T \, dx + \int_{\Omega} r \cdot v_T \, dx \end{aligned} \quad (4.11)$$

here, dx denotes the differential element for integrating over the entire domain Ω , $ds(s)$ denote the differential element for integrating over the boundary of the domain $\partial\Omega_S$ and $ds(H)$ over the boundary $\partial\Omega_H$

The equation (4.12) represents the required variational form for solving the heat conduction equation using finite element method.

4.2.2 TIME DISCRETIZATION

The fully discretized heat equation is obtained by applying a time integration scheme to the above variational form. In simple scenarios, a finite difference scheme of implicit Euler Backward(EB) method is highly recommended. However, in the later chapters we intend to have a strong coupling between the mechanical form and the thermal forms, the Trapezoidal Rule(TR) which leads to unconditional stability. In this section, we will derive the fully discretized weak form with EB scheme and using the TR scheme. The EB time integration can be used with static linear ES equations because of the monolithic fashion, while the TR time integration is preferred to be used with the linear ED equations.

4.2.2.1 EULER BACKWARD TIME INTEGRATION

The Euler Backward time integration can be defined as follows:

$$\dot{\theta} = \frac{\theta_{n+1} - \theta_n}{\Delta t} \quad (4.12)$$

Applying the EB to the space discretized variational form (4.11) we have:

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{\theta_{n+1} - \theta_n}{\Delta t} \right) v_T dx \\ = - \int_{\partial\Omega_S} [Q_S]_{n+1} v_T ds(S) - \int_{\partial\Omega_H} h (\theta - \theta_{\infty}) v_T ds(H) \\ + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (4.13)$$

Implicit EB is very robust and has a high fidelity in most cases, but the main drawback of EB is its first order accuracy. Even at shorter timesteps, we can only acquire first order accuracy. When modeling weakly coupled static thermoelasticity, EB is sufficiently accurate to produce verifiable results, but during fully coupled thermo-elastodynamics, TR gives second order accuracy, reliability and better stability.

4.2.2.2 TRAPEZOIDAL RULE TIME INTEGRATION

As we have seen in the elasticity equations, the trapezoidal rule is defined as follows:

$$\dot{\theta}_{n+1} = \frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \quad (4.14)$$

Substituting (4.16) into (4.12):

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx \\ = - \int_{\partial\Omega_S} [Q_S]_{n+1} v_T ds(S) - \int_{\partial\Omega_H} h (\theta - \theta_{\infty}) v_T ds(H) \\ + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (4.15)$$

With the time integration schemes applied, let us now consider the solution methods.

4.3 SOLUTION METHODS

Finally, a suitable solver must be chosen to solve the thermal equations. As discussed in the solution methods for Linear Elasticity, we have the option of choosing a direct linear solver or a nonlinear iterative solver. As our focus, includes nonlinearity in the later stages of the project, it is preferred to use a nonlinear iterative newton solver to solve the heat equation.

While the variational form using EB scheme requires only a single direct solver per timestep increment, the form with the TR scheme, requires an additional step to predict the temperature rate, to be used in the current step. Hence the following procedure is to be followed while solving the Heat Equation:

Step 1: Solve for temperature field (θ_{n+1}):

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx \\ = - \int_{\partial\Omega_S} [Q_S]_{n+1} v_T ds(S) - \int_{\partial\Omega_H} h (\theta - \theta_{\infty}) v_T ds(H) \\ + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (4.15)$$

With Dirichlet boundary condition: $\theta = \theta_D$, on $\partial\Omega_D$ (4.4)

Step 2: Predict the Temperature Rate ($\dot{\theta}_{n+1}$):

$$\begin{aligned} \rho C_v \int_{\Omega} \dot{\theta}_{n+1} v_T dx \\ = - \int_{\partial\Omega_S} [Q_S]_{n+1} v_T ds(S) - \int_{\partial\Omega_H} h (\theta - \theta_{\infty}) v_T ds(H) \\ + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx + \int_{\Omega} r_{n+1} v_T \cdot dx \end{aligned} \quad (4.11)$$

Step 3: Assign Previous Timestep Variables:

Temperature:	$\theta_n = \theta_{n+1}$		
Temperature Rate:	$\dot{\theta}_n = \dot{\theta}_{n+1}$		(4.16)

Step 4: Continue Time Iteration

4.4 VALIDATION TEST CASE: TRANSIENT HEAT CONDUCTION

To validate the implementation of transient heat conduction equation, we shall consider two sets of test cases, formally known as the First and Second Danilovskaya Problem [55-54]. In relevance to the later validations, we focus only on the heat conduction part of the problem to validate our implementation.

4.4.1 PROBLEM DEFINITION

Consider a semi-infinite half-space ($x > 0$) with the bounding plane at $x = 0$. The domain is assumed to be thermally insulated leading to the following conditions:

$$\theta = \theta(x, t) \quad (4.17)$$

The bounding plane is assumed to be exposed two types of boundary condition a) sudden exposure to a constant temperature heating and b) sudden exposure to a high ambient temperature θ_∞ through a boundary layer of finite conductance. The system is as shown in Figure (4.1):

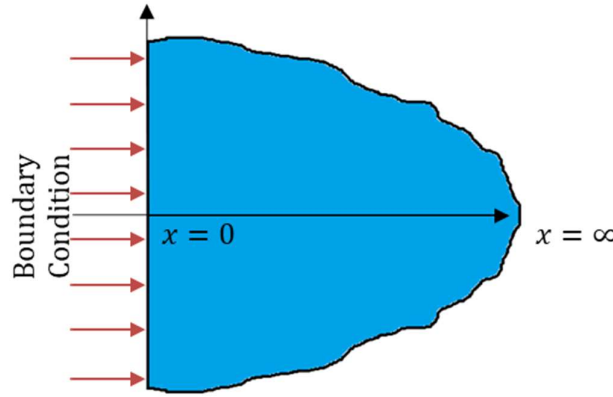


Figure (4.1) Problem Definition for Second Danilovskaya Problem

Therefore, heat conduction differential equation is:

$$\rho C_v \dot{\theta} = -\text{div } \mathbf{q}, \quad \text{in } \Omega \quad (4.1)$$

With initial condition:

$$\theta(x, 0) = 0 \quad (4.18)$$

And the boundary condition on temperature is defined as either:

a) sudden exposure to a constant temperature heating using Dirichlet boundary condition:

$$\theta(0, t) = \begin{cases} 0 & t < 0 \\ 1 & 0 \leq t \end{cases} \quad (4.19)$$

b) for the boundary layer conductance as a boundary flux density condition:

$$\mathbf{q}(0, t) \cdot \mathbf{n} = h(\theta - 1) \quad (4.20)$$

here, h is the convective heat transfer coefficient.

To compare with the analytical results, we assume that the diffusivity $\kappa = \frac{k}{\rho C_v}$ to be unity, and $H = \frac{h}{\rho C_v}$ by de-dimensionalization leading to the following partial differential equation to solve:

$$\frac{\partial \theta}{\partial t} = \text{div } (\nabla \theta), \quad \text{in } \Omega \quad (4.21)$$

For simplicity, we assume $\theta_1 = 1$ and $\theta_\infty = 1$, and the only variable remains to be H . These simplifications are taken in order to meet the specifications of the analytical solution.

The variational formulation after applying the finite element discretization and time discretization according to the procedure described in Section 4.2, we have:

For Backward Euler Time Discretization:

$$\int_{\Omega} (\theta_{n+1} - \theta_n) v_T dx + \int_{\partial\Omega} H(\theta - 1) \cdot v_T ds - \int_{\Omega} \nabla\theta \cdot \nabla v_T dx = 0 \quad (4.22)$$

For Trapezoidal Rule Time Discretization:

$$\int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx + \int_{\partial\Omega} H(\theta - 1) v_T ds - \int_{\Omega} \nabla\theta \cdot \nabla v_T dx = 0 \quad (4.23)$$

Now, we follow a direct solver for the Euler backward formulation while, the staggered approach for the Trapezoidal Rule formulation.

Variable	Assumption
$\kappa = \frac{k}{\rho C_v}$	\mathbf{I} Identity Tensor
$\theta(0, t)$	$\begin{cases} 0 & t < 0 \\ 1 & 0 \leq t \end{cases}$
$\mathbf{q}(0, t) \cdot \mathbf{n}$	$h(\theta - 1)$
\mathbf{L}	10 mm
dt	0.001 s

Table 4.1 Material Properties and Assumptions

4.4.2 SOLUTION AND INFERENCE

The numerical computation was realized using a uniform fine mesh of 300 linear Lagrange P1 elements over a length of 10 mm. A fine timestep of $dt = 0.001s$ was chosen to remove any errors. The temperature θ was measured over time at a location $x = 1$ mm, while no Dirichlet boundary conditions were specified.

In the case of the pure heat conduction of First Danilovskaya Problem (Figure 4.2), both the Euler Backward and the Trapezoidal Rule Time Integration gave results matching the exact analytical result.

And, in the case of the pure heat conduction of the Second Danilovskaya Problem (Figure 4.3), two separate test cases were studied with the convective heat transfer coefficient as $H = 0.5$ and $H = 5.0$. Here as well, the results matched those of the exact analytical solution.

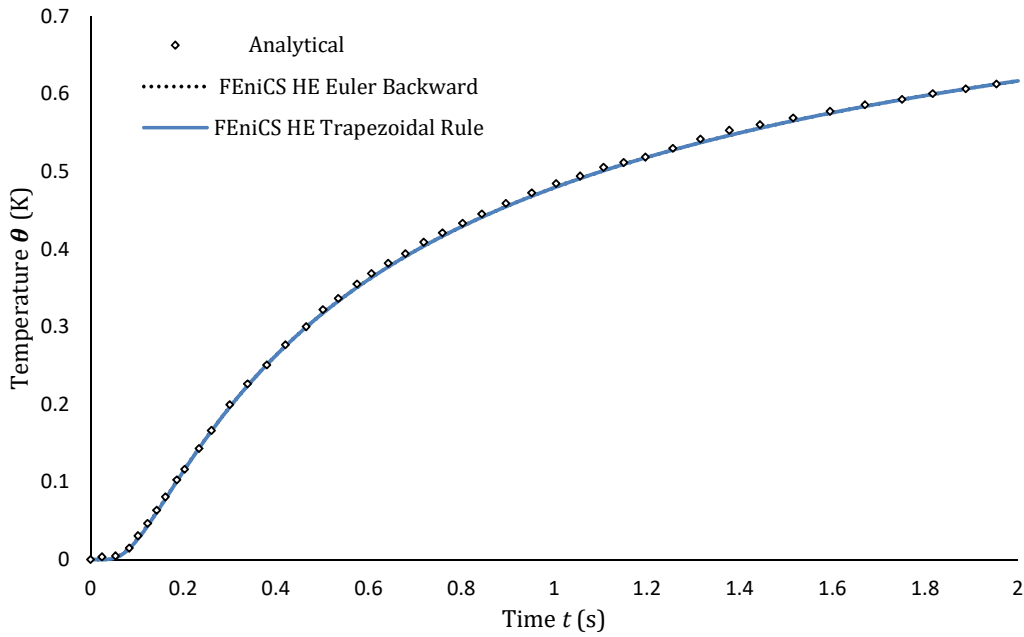


Figure (4.2) Comparison of Solution to the Pure Heat Conduction Case of First Danilovskaya Problem

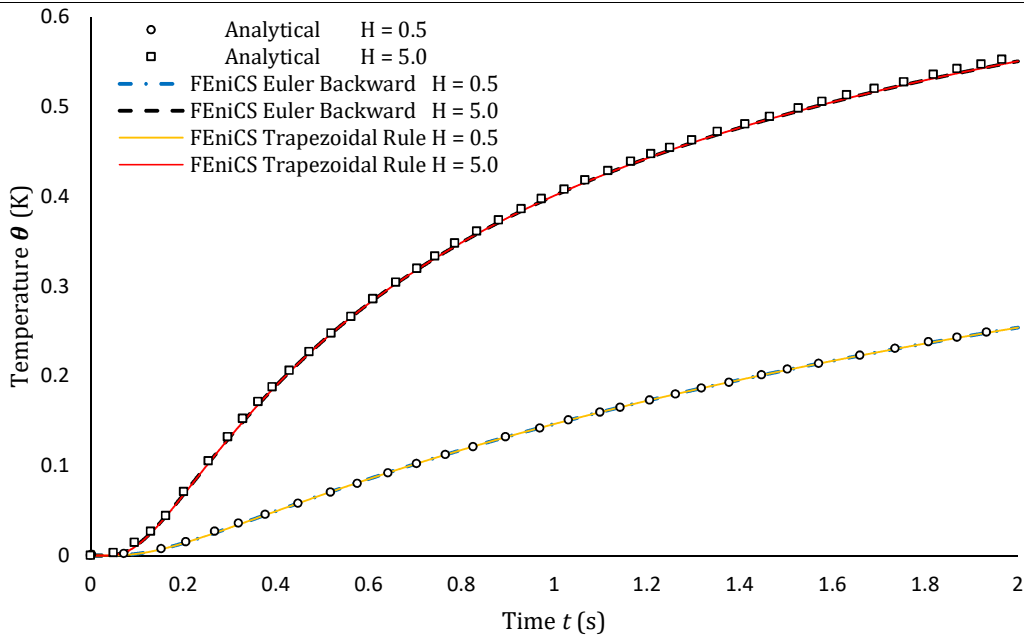


Figure (4.3) Comparison of Solution to Pure Heat Conduction Case of the Second Danilovskaya Problem with $H = 0.5$ and 5.0

Thus, from Figure (4.2) and (4.3) we have validated the heat conduction equation with Dirichlet and Neumann/Robin for both EB and TR schemes. Timestep plays a major role in the convergence of the heat conduction equation, and so it is preferred to conduct a timestep study for realistic cases.

CHAPTER 5: COUPLED THERMO-ELASTODYNAMICS

Thermo-elastodynamics (T-ED) is one of the key focus areas of this research thesis. This involves enabling a two-way coupling between the elasticity equations and the heat conduction equations. Strong temperature gradients play an influential role in metal additive manufacturing which results in the evolution of the microstructure of the printed material.

A temperature change in the material can cause volumetric deformations which could be either isotropic or orthotropic in nature. This effect is introduced into the elasticity by the addition of a temperature-dependent stress, which are in turn created by thermal strains. Within our formulations, we split the total strain tensor into elastic and thermal strains:

$$\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^E + \boldsymbol{\varepsilon}^T \quad (5.1)$$

The derivations related to the thermal strains can be found referring to Lemaitre and Chaboche[45] and can be defined as the following:

$$\boldsymbol{\varepsilon}^T = \boldsymbol{\alpha} \Delta\theta \quad (5.2)$$

where, $\boldsymbol{\alpha}$ is the orthotropic thermal expansions coefficient (CTE) tensor at θ_{Ref} and the temperature difference $\Delta\theta = \theta - \theta_{Ref}$.

Additionally, if temperature dependent thermal coefficient tensor at different temperatures are available, this equation maybe modified:

$$\boldsymbol{\varepsilon}^T = \boldsymbol{\alpha}_T (\theta - \theta_T) \quad (5.3)$$

The CTE is generally defined under the assumption of a single crystal – cubic system, from the equation of state (PVT Relations) of the material as show by L. Dubovinsky [55]:

$$\boldsymbol{\alpha} = \frac{1}{V} \left(\frac{\partial V}{\partial \theta} \right)_P \quad (5.4)$$

here, V represents the total volume, ∂V represents a finite change in volume, with respect to an infinitesimal change in temperature $\partial\theta$ at constant pressure \boldsymbol{P}

In the case of the thermal equations, the coupling term derived from the first and second laws of thermodynamics as show by Lemaitre and Chaboche [45] to be:

$$\Phi = -\theta_{Ref} \boldsymbol{\alpha} : \mathbb{C} : \dot{\boldsymbol{\varepsilon}} = -\theta_{Ref} \boldsymbol{\beta} : \dot{\boldsymbol{\varepsilon}} \quad (5.5)$$

where, $\boldsymbol{\beta}$ is the thermal stress coefficient tensor,
 $\boldsymbol{\alpha}$ is the thermal expansion coefficient tensor,
 $\dot{\boldsymbol{\varepsilon}}$ is the strain rate

5.1 GOVERNING EQUATIONS

T-ED is a combination of the basic frameworks defined in Chapter 3 and Chapter 4, with basic modification to account for their coupling. In this section, the governing equations are now re-discussed with the modifications and the variational formulations are discussed in brief.

5.1.1 THE COUPLED ELASTODYNAMIC EQUATION

We consider the displacement field as seen in Section 3.1. The Strong form of the governing equation remains the same:

$$\text{div } \boldsymbol{\sigma} + \mathbf{f} = \rho \ddot{\mathbf{u}}, \quad \text{in } \Omega \quad (3.1)$$

The definition of the elastic strain tensor:

$$\boldsymbol{\varepsilon}^E = \boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T \quad (5.6)$$

The constitutive equation defines the stress as a function of the elastic strain tensor:

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon} \quad (3.15)$$

Since the total strain in the case of pure elasticity is nothing but elastic strain, $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^E$. However, in the case of thermoelasticity, the change in the definition of the total strain, leads to the following constitutive equation:

$$\boldsymbol{\sigma} = \mathbb{C} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T) \quad (5.7)$$

5.1.2 THE COUPLED HEAT CONDUCTION EQUATION

The heat conduction equation remains almost the same, with an additional coupling term which is now added to account for the heat dissipation in the displacement field.

$$-\text{div } \mathbf{q} + \Phi + r = \rho C_v \dot{\theta}, \quad \text{in } \Omega \quad (5.8)$$

In the finite strain formulation, this is specified by Equation (5.5), leading to:

$$-\text{div } \mathbf{q} - \theta_{Ref} \boldsymbol{\beta} : \dot{\boldsymbol{\varepsilon}} + r = \rho C_v \dot{\theta}, \quad \text{in } \Omega \quad (5.9)$$

Where the heat flux is defined by the Fourier law as:

$$\mathbf{q} = -\mathbf{k} \cdot \nabla \theta \quad (4.2)$$

5.2 FINITE ELEMENT FORMULATIONS AND SOLUTION SCHEMES

Let us now fully define the IBVP with the above two governing equations, combining them with initial and boundary conditions. As we know the boundary conditions are specific to each field, we have separate Dirichlet, and Neumann boundary conditions for the Displacement and Temperature fields.

Let the Dirichlet boundary condition for displacement and temperature be enforced on $\partial\Omega_{Du}$ and $\partial\Omega_{DT}$, respectively, while the Neumann boundary condition for displacement and temperature be enforced on $\partial\Omega_{Su}$ and $\partial\Omega_{ST}$, respectively. This leads to the following conditions:

$$\text{Displacement Field} \quad \mathbf{u} = \mathbf{u}_D, \quad \text{on } \partial\Omega_{Du} \quad (5.10)$$

$$\text{Displacement Field} \quad \boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{T}_S, \quad \text{on } \partial\Omega_{Su} \quad (5.11)$$

$$\text{Temperature Field} \quad \theta = \theta_D, \quad \text{on } \partial\Omega_{DT} \quad (5.12)$$

$$\text{Temperature Field} \quad \mathbf{q} \cdot \mathbf{n} = Q_S, \quad \text{on } \partial\Omega_{ST} \quad (5.13)$$

It is also worth noting that the boundary $\partial\Omega$ is divided into disjoint partitions in the respective cases of displacement and temperature fields:

$$\partial\Omega_{Du} \cup \partial\Omega_{DT} \cup \partial\Omega_{Su} \cup \partial\Omega_{ST} \in \partial\Omega, \quad \begin{aligned} \partial\Omega_{Du} \cap \partial\Omega_{Su} &= 0 \\ \partial\Omega_{DT} \cap \partial\Omega_{ST} &= 0 \end{aligned} \quad (5.14)$$

Further, the initial conditions on both displacement field and temperature field at $t = 0$ has to be specified:

$$\mathbf{u} = \mathbf{u}(x, 0) = \mathbf{u}_0, \quad \text{on } \Omega \quad (5.15)$$

$$\dot{\mathbf{u}} = \dot{\mathbf{u}}(x, 0) = \dot{\mathbf{u}}_0, \quad \text{on } \Omega \quad (5.16)$$

$$\theta = \theta(x, 0) = \theta_0, \quad \text{on } \Omega \quad (5.17)$$

$$\dot{\theta} = \dot{\theta}(x, t = 0) = \dot{\theta}_0, \quad \text{on } \Omega \quad (5.18)$$

Finally, we have now specified the strong form of the coupled thermo-elastodynamic problem.

5.2.1 SPACE DISCRETIZATION

We follow the same procedure as in Section 3.2.1 and 4.2.1 to acquire the weak variational form of the coupled T-ED problem. Let us assuming the solve variable \mathbf{u}, θ to be trial functions, which are to be multiplied by corresponding test functions \mathbf{v}_u, v_T and integrated over the domain Ω .

For the displacement field we get the following weak variational form:

$$\int_{\partial\Omega_S} \mathbf{T}_S \cdot \mathbf{v}_u \, ds(u) - \int_{\Omega} \boldsymbol{\sigma} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f} \cdot \mathbf{v}_u \, dx = \rho \int_{\Omega} \ddot{\mathbf{u}} \cdot \mathbf{v}_u \, dx \quad (3.11)$$

However, the change from the regular variational form of Linear Elasticity, as discussed, comes from the constitutive relation. Thus, we have:

$$\int_{\partial\Omega_{Su}} \mathbf{T}_S \cdot \mathbf{v}_u \, ds(u) - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T) : \mathbb{C} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f} \cdot \mathbf{v}_u \, dx = \rho \int_{\Omega} \ddot{\mathbf{u}} \cdot \mathbf{v}_u \, dx \quad (5.19)$$

Now, for the temperature field we have:

$$\rho C_v \int_{\Omega} \dot{\theta} v_T \, dx = - \int_{\Omega} \text{div } \mathbf{q} v_T \, dx - \theta_{Ref} \int_{\Omega} \boldsymbol{\beta} : \dot{\boldsymbol{\varepsilon}} v_T \, dx + \int_{\Omega} \mathbf{r} \cdot \mathbf{v}_T \, dx \quad (5.20)$$

Again, the change from the regular variational form of the Heat Conduction Equation, arises from the coupling term Φ , which is now multiplied with the test function and integrated over the domain Ω . This leads to the following variational form for the coupled heat equation:

$$\begin{aligned} \rho C_v \int_{\Omega} \dot{\theta} v_T \, dx = & - \int_{\partial\Omega_{ST}} Q_S v_T \, ds(T) + \int_{\Omega} \mathbf{q} \cdot \nabla v_T \, dx \\ & - \theta_{Ref} \int_{\Omega} \boldsymbol{\beta} : \dot{\boldsymbol{\varepsilon}} v_T \, dx + \int_{\Omega} \mathbf{r} \cdot \mathbf{v}_T \, dx \end{aligned} \quad (5.21)$$

Now, we can move to time discretization.

5.2.2 TIME DISCRETIZATION

In view of the solution algorithm which we will discuss in Section 3.3, we intend to use the trapezoidal rule for time integration. From applying the TR time integration scheme from Equation (3.12) into the variational form of the coupled-ED equation (5.19) we have:

$$\begin{aligned} \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u \, dx = & \int_{\partial\Omega_{Su}} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u \, ds(u) \\ & - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \mathbb{C} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u \, dx \end{aligned} \quad (5.22)$$

Similarly, applying the TR integration scheme from Equation (4.14) into the variational form of the coupled heat conduction equation (5.21) we have:

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx \\ = - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx \\ - \theta_{Ref} \int_{\Omega} \boldsymbol{\beta} : \boldsymbol{\varepsilon}_n v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (5.23)$$

Finally, we have the variational forms of the coupled thermo-elastodynamics equations.

5.3 SOLUTION METHODS

In the wake of existence of very strong coupling between the dynamic displacement field and the temperature field, we select a staggered algorithm to solve this IBVP problem. Farhat et al, [47] proposed an unconditionally stable Staggered approach using the Trapezoidal rule which solves the coupled heat equation in a step by step manner. Initially, the coupled heat equation is solved, followed by the coupled ED equation, then followed by acceleration, velocity and temperature rates. Although, the number of solve processes are higher, this algorithm has proof of unconditional stability and is best fulfils our strong coupling requirement. Since, the variational form in the staggered approach remains in linear forms, in case of high gradient loading, a nonlinear iterative solver is preferred to solve the Coupled T-ED.

After careful implementation and verification, the following version of the algorithm proposed by Farhat C et al.[47], is selected for unconditional stability. Here, we use an initially predicted strain rate for the coupled heat equation:

Step 1: Solve the Coupled Heat Equation for Temperature Field (θ_{n+1})

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx \\ = - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx \\ - \theta_{Ref} \int_{\Omega} \boldsymbol{\beta} : \boldsymbol{\varepsilon}_n v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (5.23)$$

Step 2: Solve the Coupled Elastodynamic Equation for Displacement Field (\mathbf{u}_{n+1})

$$\begin{aligned} \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u dx = \int_{\partial\Omega_{Su}} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u ds(u) \\ - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \mathbb{C} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u dx \end{aligned} \quad (5.22)$$

Step 3: Solve for the Coupled Acceleration Field ($\ddot{\mathbf{u}}_{n+1}$)

$$\begin{aligned} \rho \int_{\Omega} \ddot{\mathbf{u}}_n \cdot \mathbf{v}_u dx = \int_{\partial\Omega_{Su}} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u ds(u) \\ - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \mathbb{C} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u dx \end{aligned} \quad (3.11)$$

Step 4: Update the Velocity Field ($\dot{\mathbf{u}}_{n+1}$)

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{\Delta t}{2} (\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1}) \quad (3.13)$$

Step 5: Solve for the Coupled Temperature Rate ($\dot{\theta}_{n+1}$)

$$\begin{aligned} \rho C_v \int_{\Omega} \dot{\theta}_{n+1} v_T dx = & - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx \\ & - \theta_{Ref} \int_{\Omega} \boldsymbol{\beta} : \dot{\boldsymbol{\epsilon}}_{n+1} v_T dx + \int_{\Omega} r_{n+1} v_T dx \end{aligned} \quad (5.21)$$

Step 6: Assign Previous Timestep Variables:

$$\begin{aligned} \text{Temperature:} & \quad \boldsymbol{\theta}_n = \boldsymbol{\theta}_{n+1} \\ \text{Temperature Rate:} & \quad \dot{\boldsymbol{\theta}}_n = \dot{\boldsymbol{\theta}}_{n+1} \\ \text{Displacement:} & \quad \mathbf{u}_n = \mathbf{u}_{n+1} \\ \text{Velocity:} & \quad \dot{\mathbf{u}}_n = \dot{\mathbf{u}}_{n+1} \\ \text{Acceleration:} & \quad \ddot{\mathbf{u}}_n = \ddot{\mathbf{u}}_{n+1} \end{aligned} \quad (5.24)$$

Step 7: Continue Time Increment

5.4 VALIDATION TEST CASE: THE SECOND DANILOVSKAYA PROBLEM

The first analytical solution to a dynamic thermoelastic initial boundary value problem was obtained by Danilovskaya [53,54] in 1950-52. This problem concerns a linear elastic half-space subjected to a uniform sudden temperature at its bounding plane. The analytical solution was calculated from the classical heat equation without the coupling term and then a forcing function within the elasticity equation would provide the solution to the dynamics. This initial boundary value problem is called as the First Danilovskaya Problem. Danilovskaya later, extended her results accounting boundary layer conductance along the bounding plane, called the Second Danilovskaya Problem. Although initially this was first proposed as a weakly coupled problem by Danilovskaya, the effects of the thermomechanical coupling as well as inertia was later accounted by Boley and Tolins [56] in addition to Muki and Breuer [57] leading to the problem having a fully coupled analytical solution.

Nickel and Sackman [58] also provided an approximate solution to solve the above initial boundary value problem using the Ritz method. We constrain ourselves to the analytical results specified in their paper to now compare our fully coupled implementation finite element implementation to assess its validity.

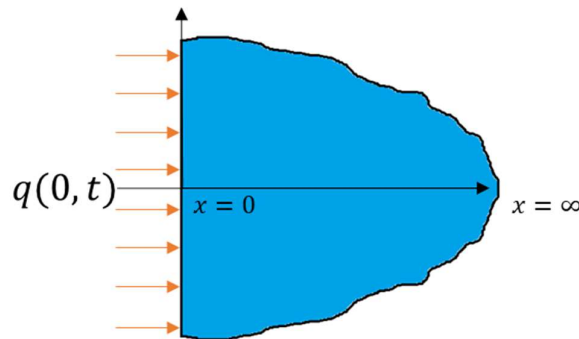


Figure (5.1) Problem Definition for Second Danilovskaya Problem

5.4.1 PROBLEM DEFINITION

Consider an elastic half-space ($x > 0$) with the bounding plane at $x = 0$ as shown in Figure (5.1). This boundary is assumed free of traction at all time. The domain is assumed to be fully mechanically constrained and thermally insulated leading to the following displacement conditions;

$$\begin{aligned} \mathbf{u}_x &= \mathbf{u}_x(x, t) \\ \mathbf{u}_y &= \mathbf{u}_z = \mathbf{0} \end{aligned} \quad (5.24)$$

While the temperature is of the form:

$$\theta = \theta(x, t) \quad (5.25)$$

The bounding plane is assumed to be exposed to a sudden exposure to a high ambient temperature θ_∞ through a boundary layer of finite conductance.

Therefore, coupled thermo-elastodynamic differential equations are:

$$\rho \ddot{\mathbf{u}} = \text{div} \left(\mathbb{C} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T) \right), \quad \text{in } \Omega \quad (5.26a)$$

$$\rho C_v \dot{\theta} = -\text{div} \mathbf{q} - \theta_{Ref} \boldsymbol{\beta} : \dot{\boldsymbol{\varepsilon}}, \quad \text{in } \Omega \quad (5.26b)$$

With initial conditions:

$$\begin{aligned} \mathbf{u}_x(x, 0) &= \frac{\partial \mathbf{u}_x}{\partial t}(x, 0) = \mathbf{0} \\ \theta(x, 0) &= \theta_{Ref} \end{aligned} \quad (5.27)$$

The boundary conditions for the traction at $x = 0$ is defined as:

$$\sigma_{xx}(0, t) = 0 \quad (5.28)$$

And for the boundary layer conductance, the boundary condition on temperature is defined as:

$$\mathbf{q}(0, t) \cdot \mathbf{n} = h(\theta - \theta_\infty) \quad (5.29)$$

To compare with the analytical results, we transform the above initial boundary value problem using dimensionless coefficients, to the following problem statement:

$$\frac{\partial^2 \mathbf{u}}{\partial \tau^2} = \text{div} \left(\frac{\partial \mathbf{u}}{\partial \xi} - T \right), \quad \text{in } \Omega \quad (5.30a)$$

$$\frac{\partial T}{\partial \tau} = \text{div} \left(\frac{\partial T}{\partial \xi} \right) - \delta \frac{\partial^2 \mathbf{u}}{\partial \xi \partial \tau}, \quad \text{in } \Omega \quad (5.30b)$$

where, $\mathbf{u} = f_1(u_x)$; $T = f_2(\theta)$; $\tau = f_3(t)$ and $\xi = f_4(x)$ where f_1, f_2, f_3 and f_4 are transformation functions to dimensionless quantities.

The detailed definition of the above functions can be found with full explanation in Nickel and Sackman [049]. The δ represents a transformed function of constants which controls the coupling effect.

Similarly, the transformed initial conditions are:

$$\mathbf{u}(\xi, 0) = \frac{\partial \mathbf{u}}{\partial \tau}(\xi, 0) = T(\xi, 0) = 0 \quad (5.31)$$

And boundary conditions:

$$\sigma_{\xi\xi}(0, t) = 0 \quad (5.32)$$

$$\frac{\partial T}{\partial \xi}(0, \tau) \cdot \mathbf{n} = H(T - 1) \quad (5.33)$$

For simplifying the analytical solution, we assume, $(\theta - \theta_\infty)/\theta_0 = 1$ leading to $T_\infty = 1$ in the transformed boundary condition and H is the transformed coefficient corresponding to the convective heat transfer coefficient (h)

Following the steps in chapter 3, 4 and 5, the transformed coupled elastodynamic equation (5.30) after the finite element discretization becomes:

$$\int_{\Omega} \ddot{\mathbf{u}} \cdot \mathbf{v}_u d\xi = - \int_{\Omega} (\nabla \mathbf{u} - T\mathbf{I}) : \nabla \mathbf{v}_u d\xi \quad (5.34)$$

And after applying trapezoidal rule for the discretization of the time function τ :

$$\int_{\Omega} \left(\frac{4}{\Delta\tau^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta\tau \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u d\xi = - \int_{\Omega} (\nabla \mathbf{u} - T\mathbf{I}) : \nabla \mathbf{v}_u d\xi \quad (5.35)$$

And now in the case of the transformed coupled heat conduction equation (0.0) the finite element formulation leads to:

$$\int_{\Omega} \dot{T} v_T dx = - \int_{\partial\Omega} H(T - 1) v_T ds + \int_{\Omega} \nabla T \cdot \nabla v_T dx - \delta \nabla \dot{\mathbf{u}} : \mathbf{I} v_T dx \quad (5.36)$$

And after applying trapezoidal rule for the discretization of the time function τ :

$$\begin{aligned} \int_{\Omega} \left(\frac{2}{\Delta\tau} (T_{n+1} - T_n) - \dot{T}_n \right) v_T dx \\ = - \int_{\partial\Omega} H(T - 1) v_T ds + \int_{\Omega} \nabla T \cdot \nabla v_T dx - \delta \nabla \dot{\mathbf{u}} : \mathbf{I} v_T dx \end{aligned} \quad (5.37)$$

As stated in Chapter 5, Section 5.3 we use the staggered approach by Farhat C et al.[47], to solve the above discretized variational formulations. Additionally, as FEniCS already enforces a zero-traction boundary condition on non-Dirichlet boundaries, the condition of $\sigma_{\xi\xi}(0, t) = 0$ is enforced automatically.

Variable	Assumption
$\kappa = \frac{k}{\rho C_v}$	\mathbf{I} Identity Tensor
(a) : $\theta(0, t)$	$\begin{cases} 0 & t < 0 \\ 1 & 0 \leq t \end{cases}$
(b) : $\mathbf{q}(0, t) \cdot \mathbf{n}$	$h(\theta - 1)$
\mathbf{L}	10 mm
dt	0.001 s
\mathbb{C}	\mathbf{I} Identity Tensor
$\sigma_{\xi\xi}(0, t)$	0

Table 5.1 Material Properties and Assumptions

5.4.2 SOLUTION AND INFERENCE

For the validation test case with our finite element problem implementation, we now consider a one-dimensional domain of length 10.0 ξ units, discretized into 600 elements and a total time of 2.0 τ units over 2000 equally divided timesteps. The mesh and timesteps have been chosen such that the mesh is too fine and the timestep small enough to remove any errors that may come from the CFL condition for timestep and the mesh convergence errors.

The exact solution solved by E Sternberg and J.G Chakravorty[59] and obtained from Nickel and Sackman[58]. As we can see from Figure (5.2) and (5.3), the numerical solution of our implementation,

matches to that of the exact solution. And we find our implementation to be capable of solving a fully coupled thermo-elastodynamic problem.

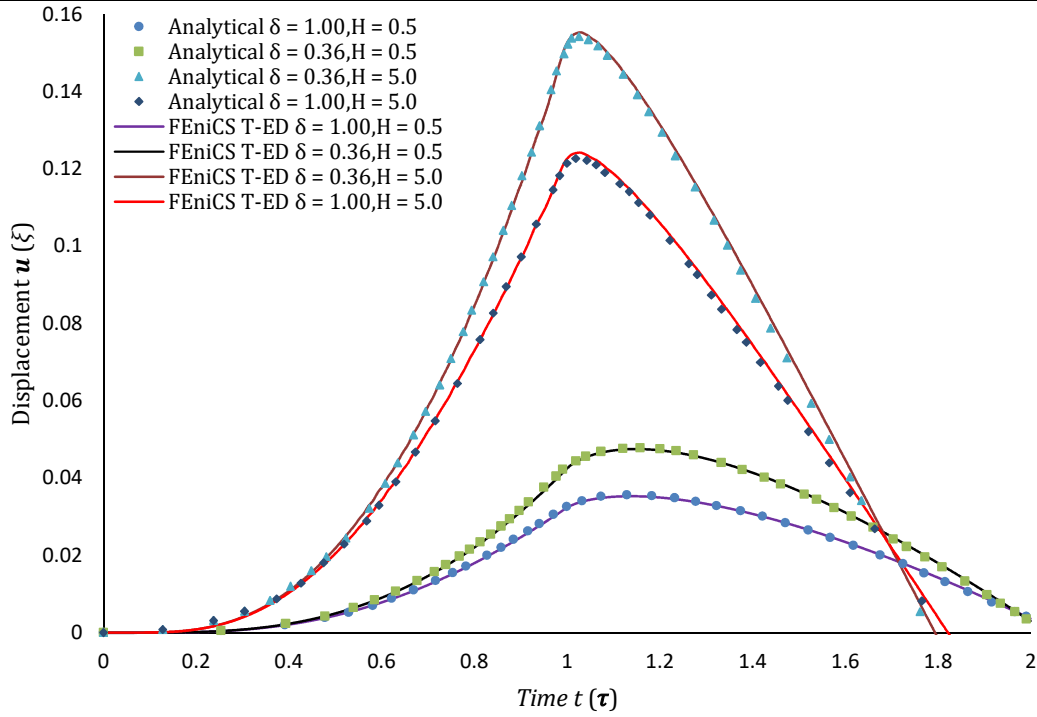


Figure (5.2) Comparison of Displacement u in Coupled Thermo-Elastodynamics of the Second Danilovskaya Problem with $H = 0.5$ and 5.0

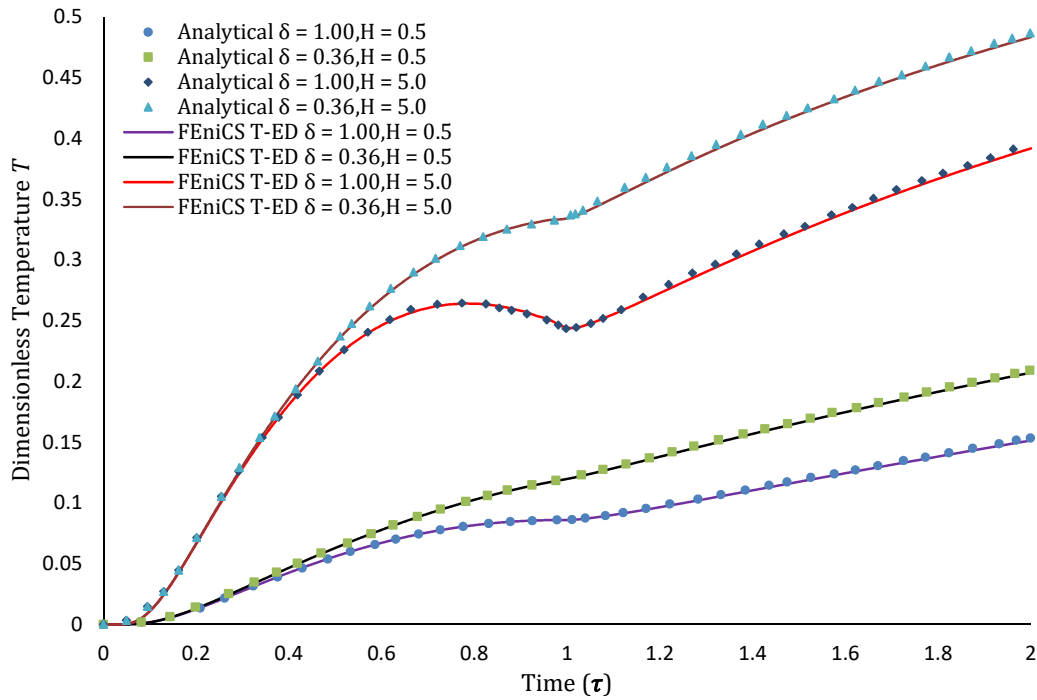


Figure (5.3) Comparison of Temperature T in Coupled Thermo-Elastodynamics of the Second Danilovskaya Problem with $H = 0.5$ and 5.0

However, since we have not used any material constants in the above verification, we are required to undertake a timestep and a mesh sensitivity study to find the required discretization for our implementation to work.

5.5 MESH SENSITIVITY AND TIMESTEP STUDY

In the mesh sensitivity and timestep study, we consider a one-dimensional domain with properties of 316L Stainless Steel, over a length of 10mm, divided into elements of equal length. A constant temperature boundary condition is applied to one surface and the displacement of that surface is considered, and in the case of the displacement field all surfaces are considered traction free. We begin with a coarse element size of 0.1mm and start decreasing the size until we acquire a suitable mesh size, where decreasing the mesh further will not change the resulting solution. Once the required, mesh size has been found, we move on to the timestep study to understand the effect of timestep, using the converged mesh size.

Based on the comments of Eran Grosu [49] on stability of the dynamic solution using Trapezoidal Rule time integration scheme, we select the initial timestep as $1e-4$ s corresponding to CFL number of 1e0. Then, we begin decreasing the CFL number 1e-1, 1e-2 etc., until we can fully observe the variation in the solution because of dynamics.

Based on our observations, we plot the displacement u_x vs time t for different mesh sizes (Figure 5.4) and for different timesteps (Figure 5.5) to understand their behavior.

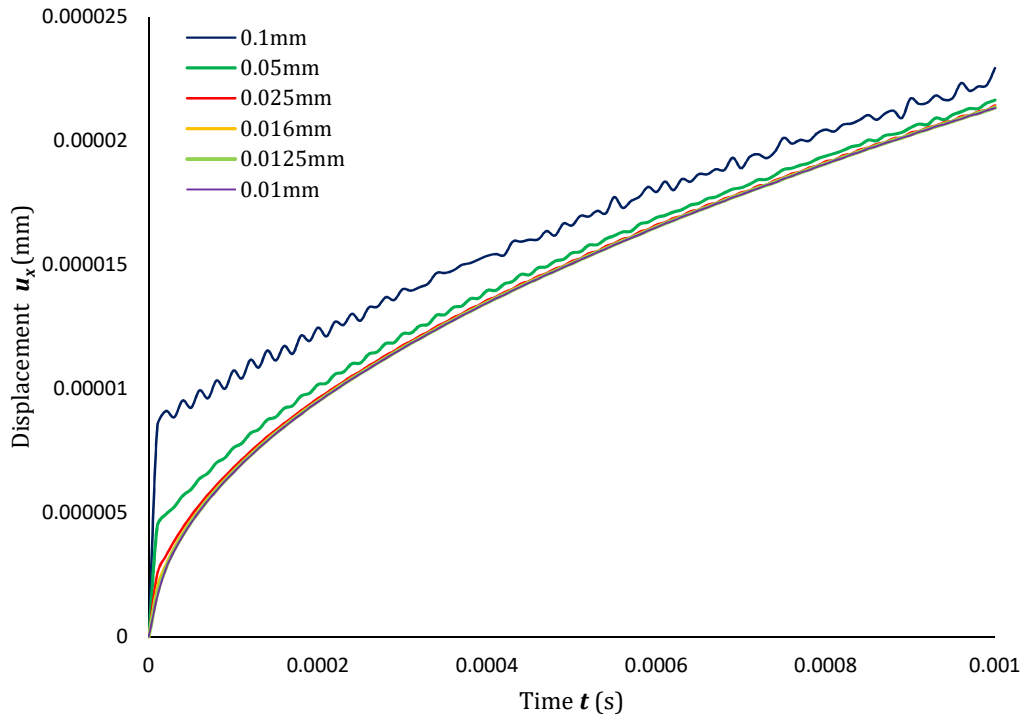


Figure (5.4) Mesh Sensitivity Study

In the case of mesh convergence, we see that the solution starts to converge at $dx = 0.025$ mm. Mesh size smaller than the convergence value does not deviate the solution. Hence, this size has been selected to now find the best timestep.

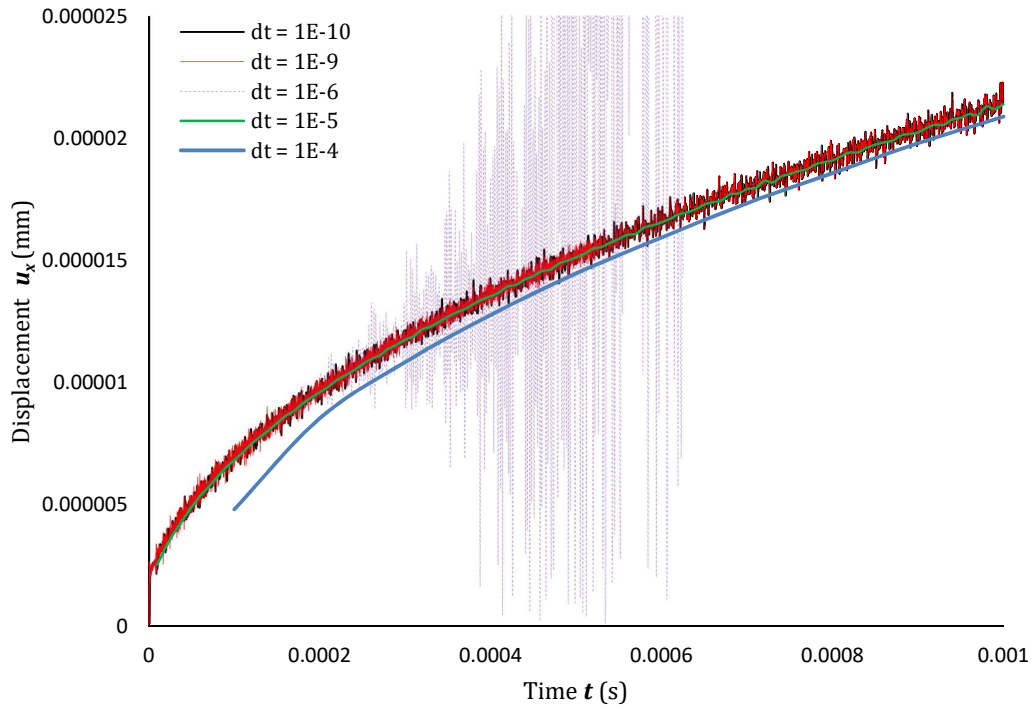


Figure (5.5) Timestep Study

In the case of Timestep Study, the complication in the timestep arises from the fact that the stress wave is discontinuous. According to Eran [039], the sharp gradients in front of the stress wave leads to spurious oscillations, which corrupt the numerical solution. Since the trapezoidal rule, does not provide algorithmic damping, a relatively smaller timestep is required for the solution.

From figure (5.5) of the timestep study, we see that a timestep higher than $1e-5$ s (CFL: $1e-1$) gives stable solution. In between timesteps of $1e-5$ s and $1e-8$ s (CFL: $1e-2$ to $1e-5$) we have a zone of instability, where the solution diverges. Decreasing further than the timestep of $1e-9$ s (CFL: $1e-5$) we begin to see the minute oscillations due to dynamics. Thus, one must use a timestep corresponding to CFL number of $1e-1$ to have a converged solution. In addition, if we are interested in observing the minute dynamic oscillations within the system, we can use a timestep corresponding to CFL number of $1e-5$ or lower.

CHAPTER 6: HETEROGENEOUS ELASTICITY

Metals are in general polycrystalline in nature. Electron Backscatter Diffraction (EBSD) and X-ray Computed Tomography (XRCT) techniques have shown that a metal is composed of multiple grains with different crystalline orientations. These grains are separated by grain boundaries and the movement of dislocations through these boundaries account for plastic deformations.

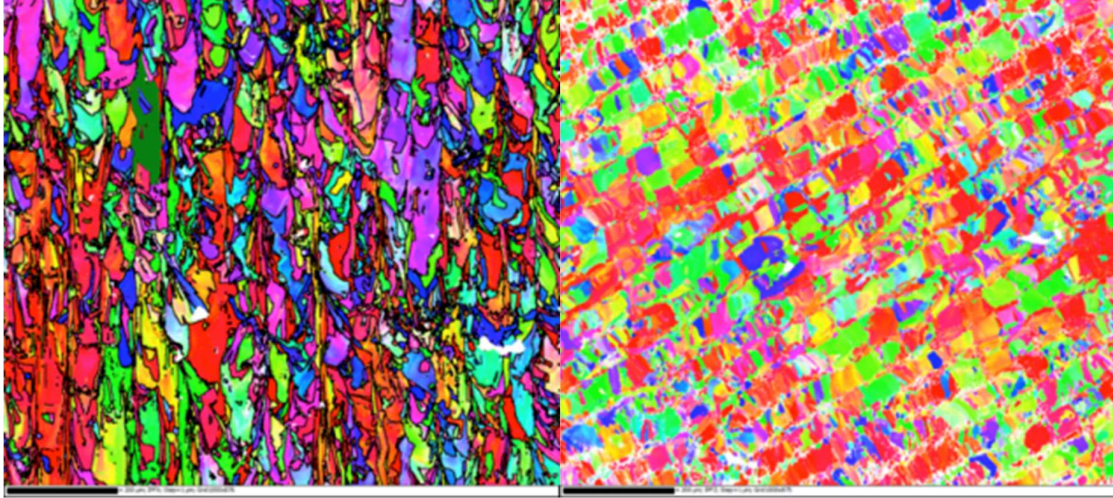


Figure (6.1) EBSD of 316L stainless steel by SLM AM technique: surfaces perpendicular and parallel to the building direction. Source: Nikolay Khailov, LMS Ecole Polytechnique

Based on the resolution of the EBSD image data, it can be used to create an orientation map of each grains in the given microstructure, containing their Euler angles.

Within the framework of Finite Element Method, one can use these Euler angles to rotate the given anisotropic stiffness properties of a single crystal and create the entire microstructure. In figure (6.2), we can see that one can create a map of stiffness properties with variable different Euler angles to recreate an entire microstructure. This property map, when coupled with a FEM Solver can be used to study the stress-strain evolution within a metal additive manufactured component.

6.1 FORMULATION

The formulation we use, to create a local 4th order elastic stiffness tensor, is based on the work of C.N. Tome and R.A Lebensohn in the VPSC Full-Field FFT Crystal Plasticity code[23-26].

Let us consider a set of Euler angles representing a given grain orientation as (θ, ϕ, ψ) . The rotation tensor \mathbf{A} and then be defined as:

$$\mathbf{A}(\theta, \phi, \psi) =$$

$$\begin{bmatrix} \cos(\psi) \cos(\phi) - \cos(\theta) \sin(\phi) \sin(\psi) & -\sin(\psi) \cos(\phi) - \cos(\theta) \sin(\phi) \cos(\psi) & \sin(\theta) \sin(\phi) \\ \cos(\psi) \sin(\phi) + \cos(\theta) \cos(\phi) \sin(\psi) & -\sin(\psi) \sin(\phi) + \cos(\theta) \cos(\phi) \cos(\psi) & -\sin(\theta) \cos(\phi) \\ \sin(\psi) \sin(\theta) & \cos(\psi) \sin(\theta) & \cos(\theta) \end{bmatrix} \quad (6.1)$$

In index notation, we can now create the oriented stiffness matrix (\mathfrak{C}) for a given element as:

$$\mathfrak{C}_{ijkl} = A_{im} A_{jn} A_{kp} A_{lq} \mathfrak{C}_{mnop} \quad (6.2)$$

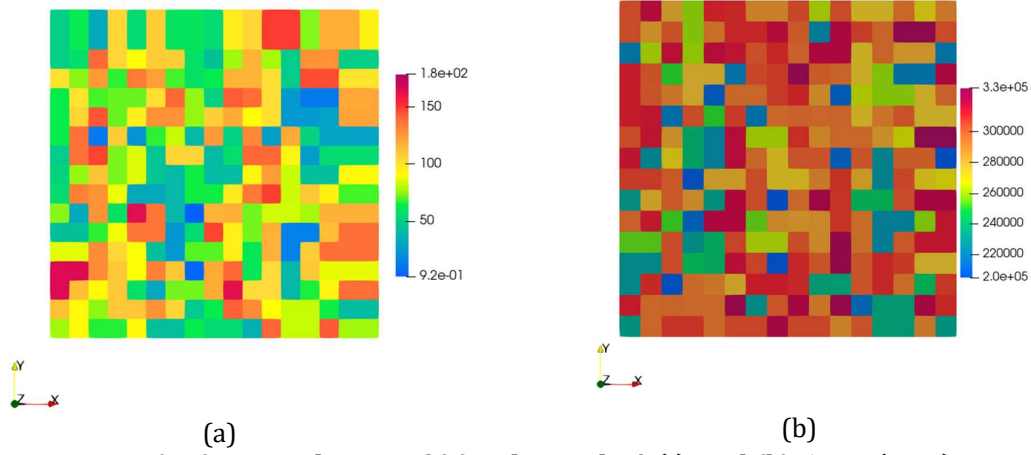


Figure (6.2) A Sample Map of (a) Euler angle ϕ (°) and (b) \mathfrak{C}_{2222} (MPa) represented over an RVE

In the case of Heterogeneous Elasticity, the governing equation remains the same as that of linear elasticity. Thus, we have:

$$\text{div } \sigma + f = \rho \ddot{u}, \quad \text{in } \Omega \quad (3.1)$$

The modification required is that now within our constitutive relation we need to include the oriented stiffness tensor, instead of the homogeneous stiffness tensor.

$$\sigma = \mathfrak{C} : \varepsilon \quad (6.3)$$

where, \mathfrak{C} is the stiffness tensor oriented to the re-oriented with the rotation tensor A .
 ε is the linearized strain tensor

6.2 HOMOGENIZATION PROBLEM

The purpose of homogenization is to calculate the apparent stress-strain behavior of an anisotropic RVE. The complete formulation can be found in [60] In linear elastic setting, the homogenization amounts the solution of the following auxiliary problem:

The governing equation:

$$\text{div } \sigma = 0, \quad \text{in } \Omega \quad (3.2)$$

The constitutive law:

$$\sigma = \mathfrak{C} : \varepsilon \quad (6.3)$$

The definition of strain:

$$\varepsilon = \varepsilon_G + \nabla^s u^* \quad (6.4)$$

where, ε_G is the applied macroscopic strain
 u^* is a periodic fluctuation in displacement field

The boundary condition:

$$\varepsilon_G = \varepsilon_G(t) \quad (6.5)$$

$$T_S = \sigma \cdot n \quad (6.6)$$

where, T_S is the enforced antiperiodic traction boundary condition.

Finite Element Discretization of equation (3.2) based on the constitutive law (6.3) and strain definition (6.4) leads to :

$$\int_{\Omega} (\boldsymbol{\varepsilon}_G + \nabla^s \mathbf{u}^*) : \mathfrak{C} : \nabla \mathbf{v}_u \cdot d\mathbf{x} = \mathbf{0} \quad (6.7)$$

Since the above problem is not well-posed due to rigid body translations, we must additionally solve for the fluctuation field \mathbf{u}^* as zero-averaged. This is done by considering an a vectoral Lagrange multiplier Λ , such that the monolithic form of equation (6.7) becomes:

$$\int_{\Omega} (\boldsymbol{\varepsilon}_G + \nabla^s \mathbf{u}^*) : \mathfrak{C} : \nabla \mathbf{v}_u \cdot d\mathbf{x} + \int_{\Omega} \Lambda \mathbf{v}_u \cdot d\mathbf{x} + \int_{\Omega} v_{\Lambda} \mathbf{u}^* \cdot d\mathbf{x} = \mathbf{0} \quad (6.8)$$

where, v_{Λ} is a test function for the auxiliary problem

Now, we move to the validation test case to prove our model's validity.

6.3 VALIDATION TEST CASE: EVP FFT CRYSTAL PLASTICITY MODEL

6.3.1 PROBLEM DEFINITION

For the validation test case, we consider a 1 mm^3 Cube 100 grain microstructure, Euler angles generated from an EBSD data file. The cube is subdivided into 16 Elements in X, Y and Z directions. Let us consider an anisotropic stiffness tensor corresponding to 316L Stainless Steel[052] for the non-oriented (0,0,0) crystal, represented in voigt notation as follows:

$$\mathbb{C}_{66} = \begin{bmatrix} 204.6e3 & 137.7e3 & 137.7e3 & 0 & 0 & 0 \\ 137.7e3 & 204.6e3 & 137.7e3 & 0 & 0 & 0 \\ 137.7e3 & 137.7e3 & 204.6e3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 126.2e3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 126.2e3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 126.2e3 \end{bmatrix} \text{ MPa} \quad (6.9)$$

The cube is considered equivalent to a (Representative Volume Element (RVE) with fully periodic boundary conditions in X, Y and Z directions. Since, the auxiliary problem is zero averaged, no additional Dirichlet boundary conditions are provided while a macroscopic strain rate $\dot{\boldsymbol{\varepsilon}}$ of 1.0 s^{-1} is provided on the X direction. A P2 second order interpolation degree is chosen for the displacement field, such that the strains are continuous with a P1 first order interpolation.

We compare our solution with the EVP-FFT Crystal Plasticity model[23] to validate our model. A major difference between our FEniCS FEM homogenization formulation and the EVP-FFT Crystal Plasticity formulation is the positioning stiffness tensor. In FEM, the stiffness tensor is assigned to an element, and with P2 interpolation, is shared between 12 integration points. While in the case of EVP-FFT, the stiffness matrix is assigned to one discrete integration point.

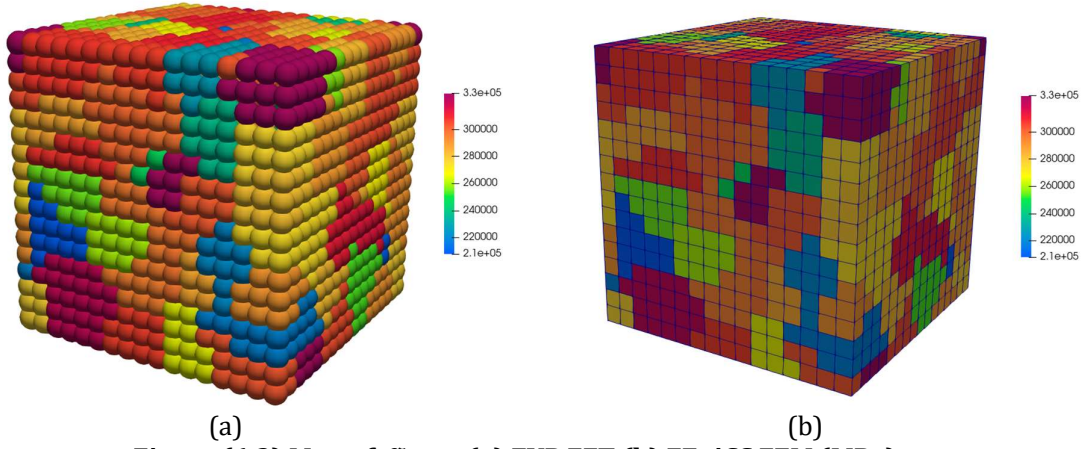


Figure (6.3) Map of \mathfrak{C}_{2222} (a) EVP FFT (b) FEniCS FEM (MPa)

6.3.2 SOLUTION AND INFERENCE

To compare the solution, we consider the independent normal stress and normal strain components, corresponding to $X_A(X, Y, Z) = (0.0, 0.5, 0.5)$ and $X_B(X, Y, Z) = (1.0, 0.5, 0.5)$ in addition to the similarities in the profile.

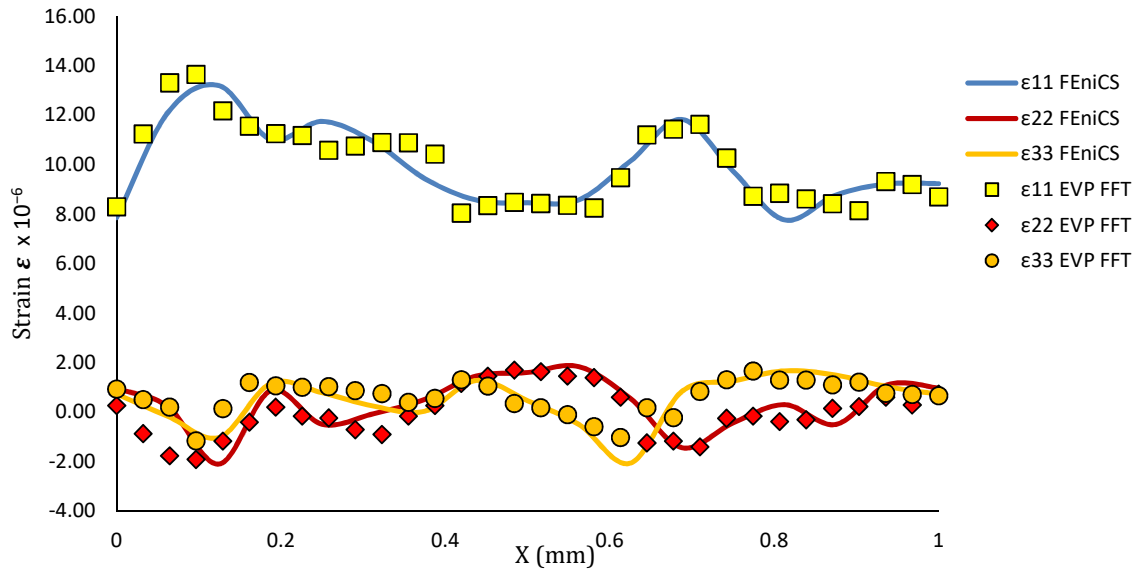


Figure (6.4) Normal Strain (ϵ) values plotted over the loading direction along X_A and X_B

From Figure (6.4) and Figure (6.5) we see that the trend for the normal stress as well as normal strain of follows the same as that for EVP FFT. Since we are comparing an FEM formulation with 12 integration points for a hexahedral element with the EVP FFT formulation with 1 integration point per material point, we understand that the values would not match exactly.

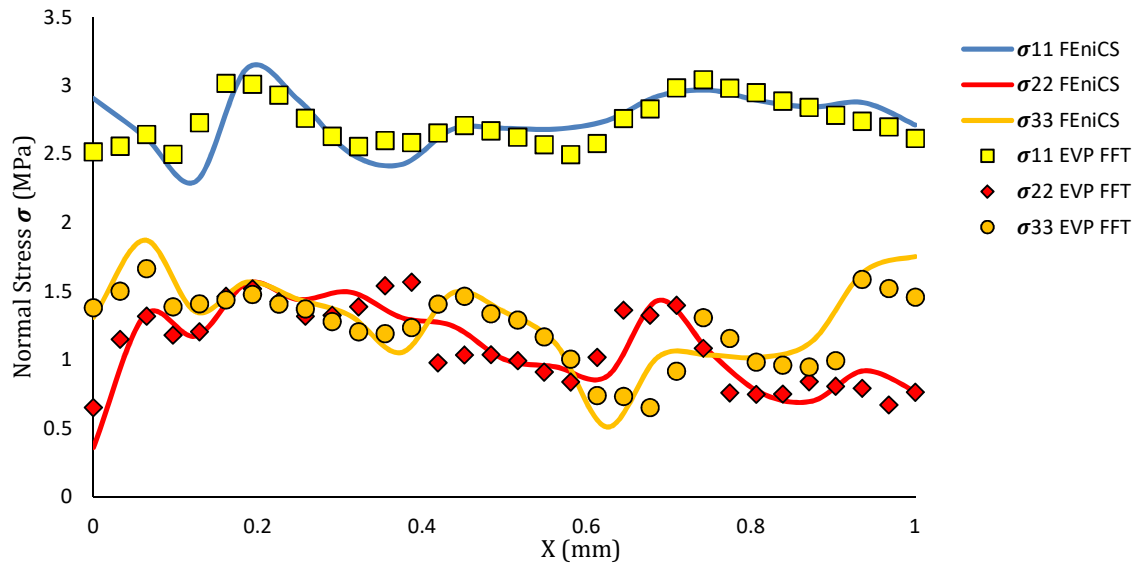


Figure (6.5) Normal Stress (σ) values plotted over the loading direction along X_A and X_B

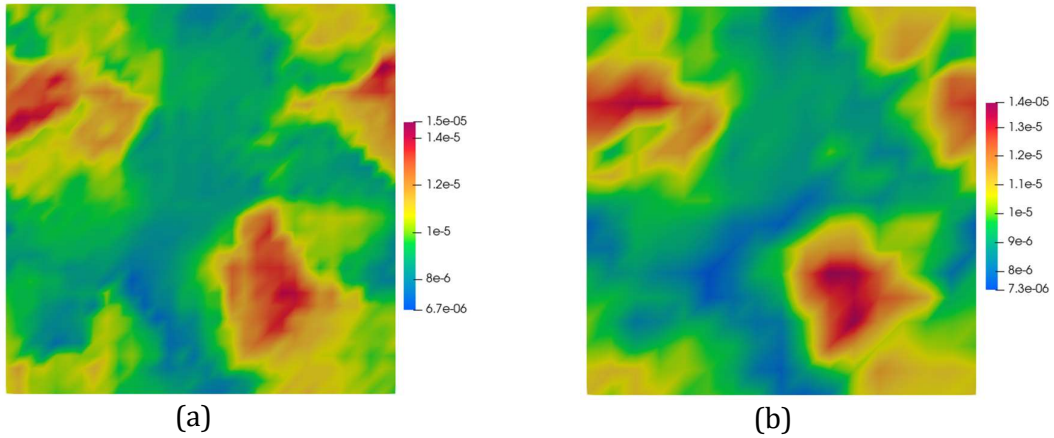


Figure (6.6) ε_{11} values plotted over the cross-section of geometry under consideration (a) EVP FFT (b) FEniCS FEM for a microstructure with 100 grains

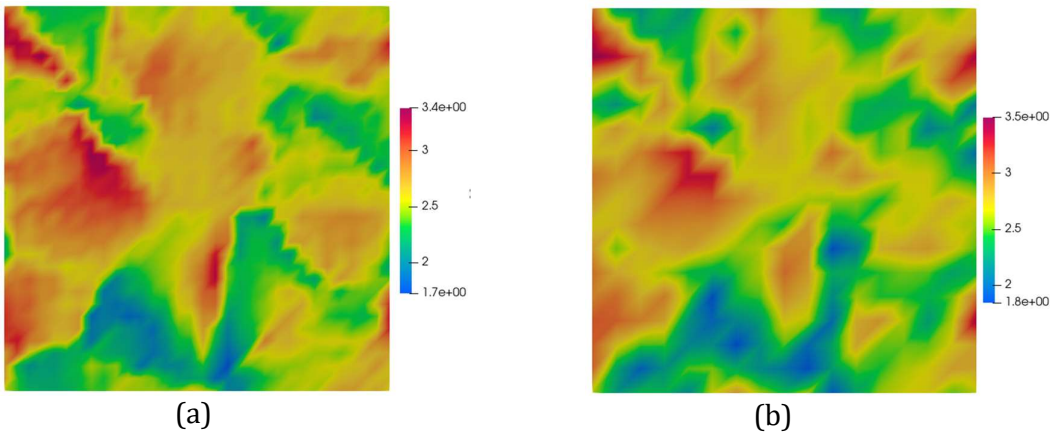


Figure (6.7) σ_{11} values plotted over the cross-section of geometry under consideration (a) EVP FFT (b) FEniCS FEM for a microstructure with 100 grains

Further, from the Figure 6.6) and Figure (6.7), we also see that the stress and strain profiles of the FEM along the mid XZ cross-section plane are in good agreement with those of the EVP FFT. Thus, we consider our heterogeneous elasticity model validated.

CHAPTER 7: HETEROGENEOUS THERMO-ELASTODYNAMICS

At this point, we have successfully validated and combined each of the constituent components of this model. The following is a summary of the complete set of equations, and solution methods, used within the model:

7.1 GOVERNING EQUATIONS

7.1.1 THE COUPLED DISPLACEMENT FIELD

The Governing Equation for Coupled Elastodynamics:

$$\text{div } \boldsymbol{\sigma} + \mathbf{f} = \rho \ddot{\mathbf{u}}, \quad \text{in } \Omega \quad (7.1)$$

The definition of elastic strain tensor:

$$\boldsymbol{\varepsilon}^E = \boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T \quad (7.2)$$

The constitutive law with heterogeneous local stiffness tensor now reads:

$$\boldsymbol{\sigma} = \boldsymbol{\mathfrak{C}} : \boldsymbol{\varepsilon} \quad (7.3)$$

$$\boldsymbol{\sigma} = \boldsymbol{\mathfrak{C}} : (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T) \quad (7.4)$$

The Variational Formulation of the Coupled Elastodynamic Equation

$$\begin{aligned} \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u \, dx &= \int_{\partial\Omega_{Su}} [\mathbf{T}_S]_{n+1} \cdot \mathbf{v}_u \, ds(u) \\ &- \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \boldsymbol{\mathfrak{C}} : \nabla \mathbf{v}_u \, dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u \, dx \end{aligned} \quad (7.5)$$

With this we move to the Coupled Temperature field definition.

7.1.2 THE COUPLED TEMPERATURE FIELD

The Governing Equation for Coupled Heat Conduction Equation:

$$-\text{div } \mathbf{q} - \theta_{Ref} \boldsymbol{\mathfrak{B}} : \dot{\boldsymbol{\varepsilon}} + \mathbf{r} = \rho C_v \dot{\theta}, \quad \text{in } \Omega \quad (7.6)$$

The Fourier law :

$$\mathbf{q} = -\mathbf{k} \cdot \nabla \theta \quad (7.7)$$

The Heterogeneous Coefficient of Thermal Stress Tensor ($\boldsymbol{\mathfrak{B}}$) :

$$\boldsymbol{\mathfrak{B}} = \boldsymbol{\mathfrak{C}} : \boldsymbol{\alpha} \quad (7.8)$$

The Variational Formulation of the Coupled Heat Conduction Equation:

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T \, dx \\ = - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T \, ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T \, dx \\ - \theta_{Ref} \int_{\Omega} \boldsymbol{\mathfrak{B}} : \dot{\boldsymbol{\varepsilon}}_n v_T \, dx + \int_{\Omega} r_{n+1} v_T \, dx \end{aligned} \quad (7.9)$$

7.2 SOLUTION METHODS

7.2.1 THE STAGGERED ALGORITHM

As discussed in Section 5.3, with the existence of strong coupling between the dynamic displacement field and the temperature field, we use the unconditionally stable staggered algorithm using the TR time integration as proposed by Farhat et al.[47]. Thus, we have the following solution algorithm:

Step 1: Solve the Coupled Heat Equation for Temperature Field (θ_{n+1})

$$\begin{aligned} \rho C_v \int_{\Omega} \left(\frac{2}{\Delta t} (\theta_{n+1} - \theta_n) - \dot{\theta}_n \right) v_T dx \\ = - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx \\ - \theta_{Ref} \int_{\Omega} \mathfrak{B} : \dot{\boldsymbol{\varepsilon}}_n v_T dx + \int_{\Omega} \mathbf{r}_{n+1} \cdot v_T dx \end{aligned} \quad (5.23)$$

Step 2: Solve the Coupled Elastodynamic Equation for Displacement Field (\mathbf{u}_{n+1})

$$\begin{aligned} \rho \int_{\Omega} \left(\frac{4}{\Delta t^2} (\mathbf{u}_{n+1} - \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n) - \ddot{\mathbf{u}}_n \right) \cdot \mathbf{v}_u dx = \int_{\partial\Omega_{Su}} [T_S]_{n+1} \cdot \mathbf{v}_u ds(u) \\ - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \mathfrak{C} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u dx \end{aligned} \quad (5.22)$$

Step 3: Solve for the Coupled Acceleration Field ($\ddot{\mathbf{u}}_{n+1}$)

$$\begin{aligned} \rho \int_{\Omega} \ddot{\mathbf{u}}_n \cdot \mathbf{v}_u dx = \int_{\partial\Omega_{Su}} [T_S]_{n+1} \cdot \mathbf{v}_u ds(u) \\ - \int_{\Omega} (\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^T)_{n+1} : \mathfrak{C} : \nabla \mathbf{v}_u dx + \int_{\Omega} \mathbf{f}_{n+1} \cdot \mathbf{v}_u dx \end{aligned} \quad (5.19)$$

Step 4: Update the Velocity Field ($\dot{\mathbf{u}}_{n+1}$)

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n + \frac{\Delta t}{2} (\ddot{\mathbf{u}}_n + \ddot{\mathbf{u}}_{n+1}) \quad (3.13)$$

Step 5: Solve for the Coupled Temperature Rate ($\dot{\theta}_{n+1}$)

$$\begin{aligned} \rho C_v \int_{\Omega} \dot{\theta}_{n+1} v_T dx = - \int_{\partial\Omega_{ST}} [Q_S]_{n+1} v_T ds(T) + \int_{\Omega} \mathbf{q}_{n+1} \cdot \nabla v_T dx \\ - \theta_{Ref} \int_{\Omega} \mathfrak{B} : \dot{\boldsymbol{\varepsilon}}_{n+1} v_T dx + \int_{\Omega} \mathbf{r}_{n+1} \cdot v_T dx \end{aligned} \quad (5.21)$$

Step 6: Assign Previous Timestep Variables:

Temperature:	$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n+1}$	
Temperature Rate:	$\dot{\boldsymbol{\theta}}_n = \dot{\boldsymbol{\theta}}_{n+1}$	
Displacement:	$\mathbf{u}_n = \mathbf{u}_{n+1}$	(5.24)
Velocity:	$\dot{\mathbf{u}}_n = \dot{\mathbf{u}}_{n+1}$	
Acceleration:	$\ddot{\mathbf{u}}_n = \ddot{\mathbf{u}}_{n+1}$	

Step 7: Continue Time Increment

7.2.2 INTERPOLATION DEGREE AND MESH ELEMENT

In addition, it is also worth to mention the preferred interpolation degrees and element types for each functional space variables:

The mechanical field variables:

\mathfrak{C}	DG0 : Discontinuous Galerkin	– Degree = 0	Shape: (3,3,3,3)
σ	CG2 : Continuous Galerkin	– Degree = 2	Shape: (3,3)
ε	CG2 : Continuous Galerkin	– Degree = 2	Shape: (3,3)
u	CG3 : Continuous Galerkin	– Degree = 3	Shape: (3)

The thermal field variables:

\mathfrak{B}	DG0 : Discontinuous Galerkin	– Degree = 0	Shape: (3,3)
k	DG0 : Discontinuous Galerkin	– Degree = 0	Shape: (3,3)
α	DG0 : Discontinuous Galerkin	– Degree = 0	Shape: (3,3)
q	CG2 : Continuous Galerkin	– Degree = 2	Shape: (3)
θ	CG2 : Continuous Galerkin	– Degree = 2	Shape: (1)

7.2.3 MESH AND TIMESTEP SELECTION

In the case of mesh, when modeling a 1mm^3 microstructure, a mesh size of 0.05 mm or lower is recommended in order to acquire a mesh insensitive solution. If using a microstructure of a different length scale, then, it is required to undertake the mesh convergence study before moving to solve the problem.

In the case of timestep selection, if one needs to study the displacement wave propagation at grain scale, it is recommended to use a timestep corresponding to CFL Number of $1\text{e-}5$ or lower. Otherwise, if one only needs to understand the time averaged behavior, one should choose a timestep corresponding to a CFL Number $\geq 1\text{e-}1$.

7.3 MODEL DEMONSTRATION

Consider a heterogeneous cubic microstructure of 1mm^3 size centered at (0.5,0.5,0.5), initially at 300K. Let us consider the body at rest and with constrained displacements on all the surfaces to their respective normal directions.

To simulate a cool support plate, let us consider this surface ($x = 1.0$) at 300 K. Now, to simulate the SSTC, consider a cyclic introduction of a heat flux providing $500\text{W}/\text{mm}^2$ for 0.5 ms and an air convective heat transfer $9\text{W}/\text{mm}^2/\text{K}$ for a period of 5.5 ms.

Thus, the problem definition is as follows:

The displacement and temperature fields be defined as:

$$\begin{aligned} \mathbf{u} &= \mathbf{u}((x, y, z), t) \\ \theta &= \theta((x, y, z), t) \end{aligned} \tag{7.10}$$

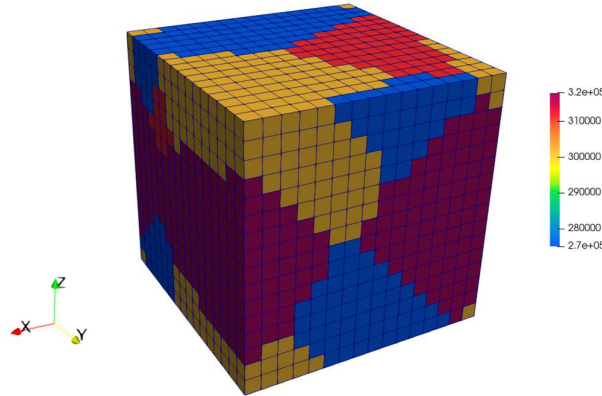
The initial conditions for the problem are:

$$\begin{aligned} u((x, y, z), 0) &= (0, 0, 0) \\ \theta((x, y, z), 0) &= 300.0 \text{ K} \end{aligned} \quad (7.11)$$

The boundary conditions for the problem are:

$$\begin{aligned} u((1.0, y, z), t) &= u((0.0, y, z), t) = (0, \text{free}, \text{free}) \\ u((x, 1.0, z), t) &= u((x, 0.0, z), t) = (\text{free}, 0, \text{free}) \\ u((x, y, 1.0), t) &= u((x, y, 0.0), t) = (\text{free}, \text{free}, 0) \\ \theta((1.0, y, z), t) &= 300.0 \text{ K} \\ \mathbf{q}((0.0, y, z), t) \cdot \mathbf{n} &= 500 \text{ W/mm}^2 \quad t < 0.5 \text{ ms} \\ \mathbf{q}((0.0, y, z), t) \cdot \mathbf{n} &= h(\theta - 300.0) \text{ W/mm}^2/\text{K} \quad 5.0 \text{ ms} < t < 6.0 \text{ ms} \end{aligned} \quad (1.12)$$

Let the microstructure be composed of 4 grains, meshed into 16x16x16 hexahedral elements as shown in Figure (7.1)



Figure(7.1) A map of ϵ_{1111} on a 4 Grain 16 Cube microstructure

7.4 SAMPLE SOLUTION

Concerning, the model demonstration problem we choose to compare the solution for the fully coupled(FC) T-ED, weakly coupled(WC) – T-ED and weakly coupled(WC) T-ES models.

From figure (7.2) we see that the temperature vs time plot is the same for all cases, and since the conductivity is presently isotropic, the homogeneous and heterogeneous cases give the same solution for the temperature profile. While, the full coupling play an important role, the time specified for cooling is too less to find considerable difference.

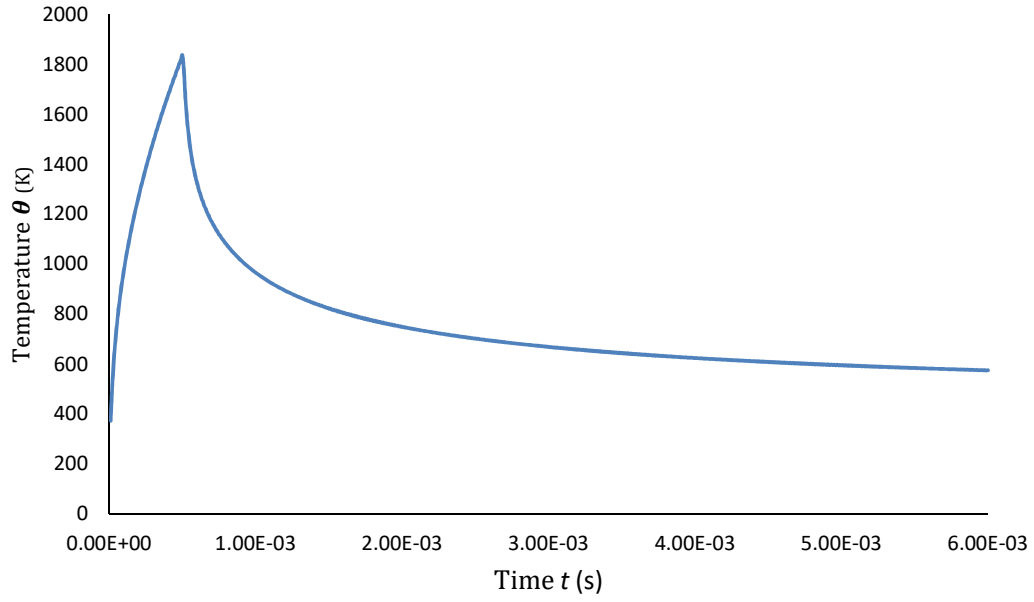


Figure (7.2) Temperature θ (°) Vs Time t (s) at surface $x = 0$

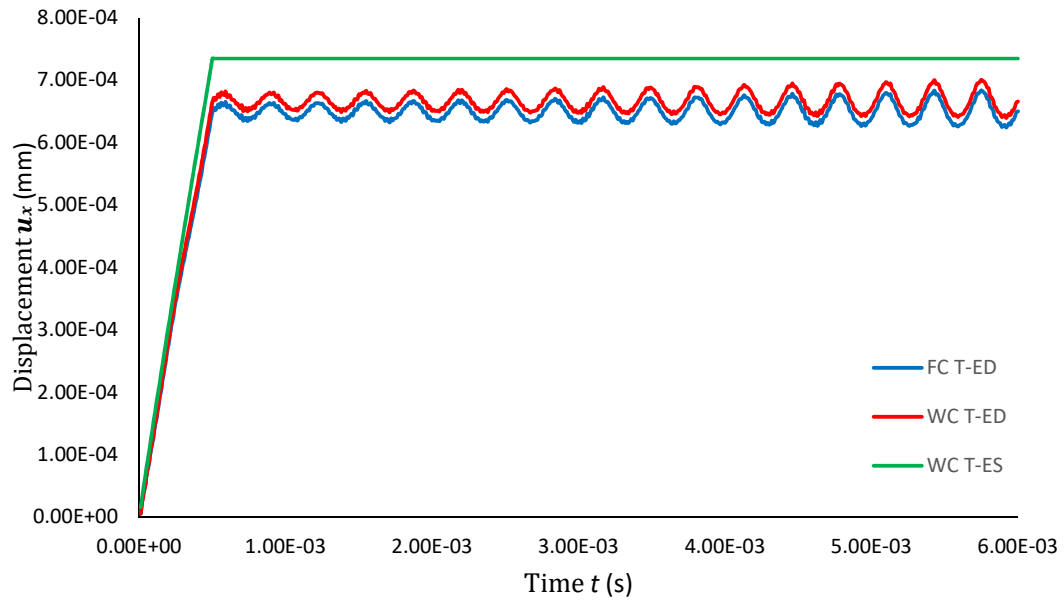


Figure (7.3) Displacement (u_x) Vs Time t (s) at cube centroid $(x, y, z) = (0.5, 0.5, 0.5)$

In figure (7.3) we observe that the WC T-ES model is quite different from the other models. The EB Time integration scheme applied for the T-ES is of only first order accuracy, thus for a problem with high gradient input loads, the solver tends to deviate from the real solution. One possible solution to this problem is to use a staggered scheme with a higher order time integration scheme.

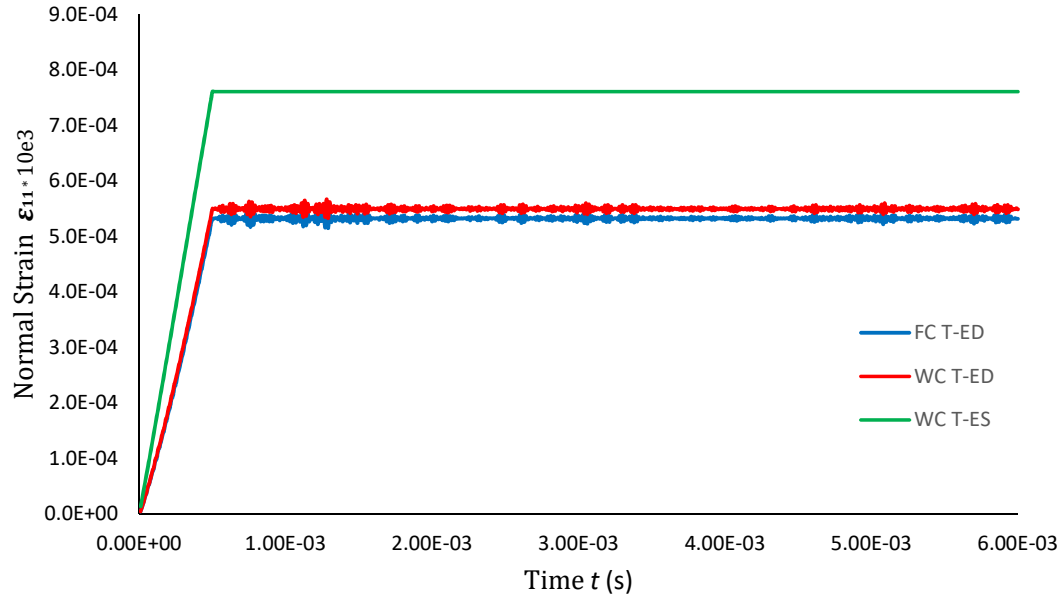


Figure (7.4) Strain (ϵ_{11}) Vs Time t (s) at cube centroid $(x, y, z) = (0.5, 0.5, 0.5)$

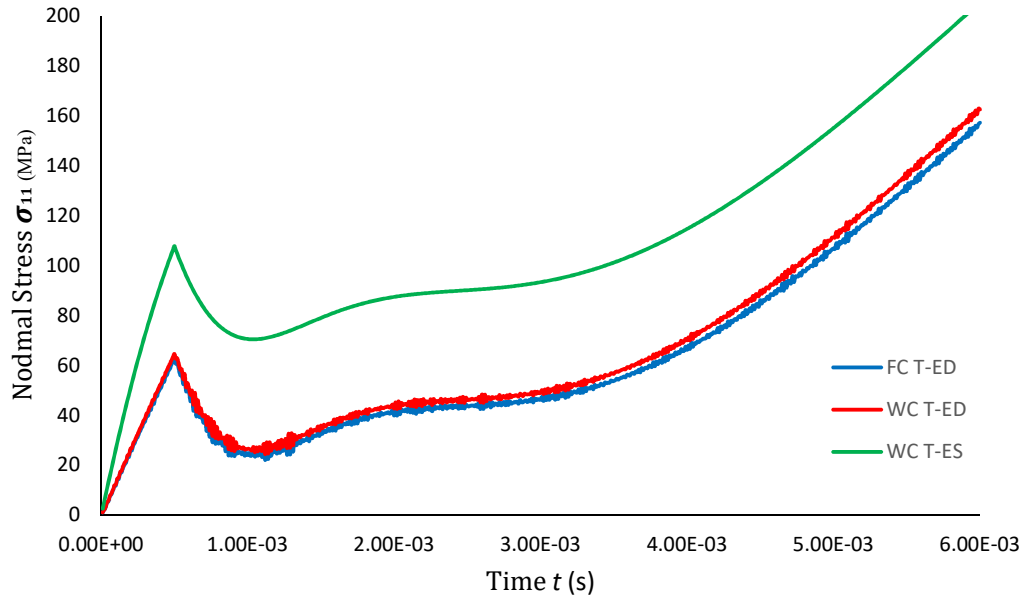


Figure (7.5) Stress (σ_{11}) Vs Time t (s) at cube centroid $(x, y, z) = (0.5, 0.5, 0.5)$

From figure (7.4) and (7.5), it is evident that the stress evolves during SSTC process. Furthermore, it is important to notice that the FC T-ED model has the least amount of stress generated, as observed from the solution of the Second Danilovskaya problem[58].

In addition, we also see that the strains have clearly gone beyond the elasticity limits, it is important to include a plasticity model to have a better understanding of the mechanics.

Conclusively, we have a working model for heterogeneous coupled thermo-elastodynamics modeling.

CHAPTER 8: CONCLUSION AND PERSPECTIVES

8.1 CONCLUSION

In this thesis, a fully coupled Thermo-elastodynamic (TED) solver for modeling and simulating solid-state thermal cycling (SSTC) on a heterogeneous microstructure has been developed using FEniCS. It is aimed at laying foundations to understand the underlying mechanics behind microstructure evolution during AM.

The model has been successfully tested and validated with analytical solutions and implementation. Furthermore, the final model thus developed, has been briefly tested using an application test case and found able to simulate SSTC.

During the study it has been observed that the timescale at which the AM process occurs i.e., greater than 10^{-3} seconds, the effects of dynamics, and the current temperature dependency of the coupling term Φ , do not play a significant role in the T-ED process. However, to properly see and understand the effects of dynamics and the coupling term, one needs to consider lower timescales of the order $10^{-6} - 10^{-8}$, which becomes more relevant when modeling plasticity.

It is also worth mentioning that the T-ES model which uses EB time integration, overestimates both the temperature and displacement response when compared to the T-ED models which uses TR time integration. This is likely to be a result of the involved EB scheme which only provides first order accuracy compared to TR's second order accuracy. Thus, an implementation with alternate time integration schemes is also recommended for T-ES.

In summary, a validated heterogeneous T-ED model for simulating solid-state thermal load over a heterogeneous microstructure has been successfully established.

8.2 PERSPECTIVE

Based on the foundation laid during this thesis the following extensions are recommended:

1. Implementation of Viscous models leading to Thermo-Visco-Elastodynamics(T-VED) model.
2. Implementation of phenomenological Plasticity Models leading to Thermo-Elasto-Visco-Plasticity(T-EVP) model.
3. Implementation of Phase Transformation models.
4. Implementation of Crystal Plasticity models.
5. Verification with experimental results.

This model is proposed to form the basis for a novel dislocation dynamics model.

APPENDIX I :

9.1 NOTATIONS

div	Divergence Operator
∇	Gradient Operator
$:$	Tensorial inner product
\cdot	Dot product
\blacksquare^T	Transpose Operator
\blacksquare^{\cdot}	1st order time derivative
$\blacksquare^{\ddot{\cdot}}$	2nd order time derivative
v_u	Vector Test Function
v_T	Scalar Test Function
n	Normal Vector
C_v	Specific Heat
ρ	Material Density
t	Time Variable
Δt	Timestep Increment
Φ	Thermal Coupling Term
dx	Finite element over domain
ds	Finite element over surface
A	Rotation Tensor
σ	Cauchy Stress Tensor
ε	Total Strain Tensor
ε^E	Elastic Strain Tensor
ε^T	Thermal Strain Tensor
\mathbb{C}	Fourth Order Stiffness Tensor
\mathfrak{C}	Reoriented Fourth Order Stiffness Tensor
u	Displacement Field Vector
f	Volumetric Body Force Density Vector
T_s	Surface Force Density on surface
q	Heat Flux Vector
k	Orthotropic Thermal Conductivity Tensor
θ	Scalar Temperature Field
α	Orthotropic Thermal Expansions Tensor
β	Coefficient of Thermal Stress Tensor
\mathfrak{B}	Reoriented Coefficient of Thermal Stress Tensor
r	Scalar Volumetric Heat Generation Rate
Q_s	Surface Heat Flux Density
θ, ϕ, ψ	Euler angles

APPENDIX II :

9.2 IMPLEMENTATION CODE: COUPLED THERMOELASTODYNAMICS

```
from __future__ import print_function
from ufl import shape
from ufl import indices
from dolfin import *
import numpy as np
import os
import sys
sys.path.insert(1, 'Sources')
from init_elasticity import init_elasticity
from init_thermal import init_thermal
from init_loads import init_loads
from init_ICs import init_ICs
from init_gen_mech import init_gen_mech
from init_datagrains import init_datagrains
from solversettings import solversettings
from voigt import voigt
import time
from math import floor

start_time = time.perf_counter()
set_log_level(50)
# Interval for writing out Stress-Strain Data
StressOutput = 10
os.system('clear')

#Read Time Step and other control variables first
#####
T_Time, T_Steps, Dynamics, Delta, SolverType, Peridic, SolverParameters = solversettings("Input/Solver_Settings.dat")
#####
OUTPUT_FOLDER = 'SOLUTIONS'
if(Delta == 0.0):
    OUTPUT_FILE = 'WeaklyCoupled_ThermoElastoDynamics'
elif(Delta == 1.0):
    OUTPUT_FILE = 'FullyCoupled_ThermoElastoDynamics'
else:
    OUTPUT_FILE = 'PartiallyCoupled_ThermoElastoDynamics'
print("*****")
print("*****")

if(Delta == 0.0):
    print("Program Type : Weakly Coupled ThermoElastoDynamics")
else:
    print("Program Type : Coupled ThermoElastoDynamics with Coupling Factor",Delta)

print("*****")
print("*****")

#####
##### MESH HAS TO BE DEFINED BEFORE READING MATERIAL DATA #####
#####
# Create the Mesh and Domain
X_Len = 1.0
Y_Len = 1.0
Z_Len = 1.0
```

```

EX          = 8
EY          = 8
EZ          = 8

MX          = EX
MY          = EY
MZ          = EZ

mesh        = UnitCubeMesh.create(MX, MY, MZ, CellType.Type.hexahedron)
#####
mesh.coordinates[:,0] = mesh.coordinates[:,0]*X_Len
mesh.coordinates[:,1] = mesh.coordinates[:,1]*Y_Len
mesh.coordinates[:,2] = mesh.coordinates[:,2]*Z_Len
#####

#####
##### IF USING TETRA HEDRAL CUBOIDAL BOX MESH #####
#####
# Create the Mesh and Domain
# Length = 6.
# Width = 2.
# mesh = BoxMesh(Point(0, 0, 0), Point(Length, Width, Width), 48, 4, 4)
#####

#####
##### FUNCTION TO READ HETEROGENEOUS MATERIAL PROPERTIES #####
#####
# Material Tensors are presently being read from the Files in order to their respective variables

CC          = init_elasticity("Input/Phase1/elastic_input_aniso.dat")
CG_init     = init_datagrain("Input/Microstructure/microstr-8cube-4gr.dat",EX,EY,EZ,CC)
K,Alpha     = init_thermal("Input/Phase1/therm_input.dat")
f,Traction,Flux = init_loads("Input/BoundaryConditions/loads_input.dat")
T_init,T_Ref = init_ICs("Input/InitialConditions/ICs_input.dat")
rho,Cv      = init_gen_mech("Input/Phase1/gen_mech_input.dat")

if(Flux != 0.0):
    print("Program Type: Flux Input")

#####
##### FUNCTION CLASS TO BUILD HETEROGENEOUS STIFFNESS MATRIX #####
#####

def ERoundX(x, base=X_Len/EX):
    return floor(x/base)
def ERoundY(x, base=Y_Len/EY):
    return floor(x/base)
def ERoundZ(x, base=Z_Len/EZ):
    return floor(x/base)
class BuildGlobalStiffnessMatrixE(UserExpression):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
    def eval(self, values, x):
        values[:] = CG_init[int(ERoundX(x[0])+ERoundY(x[1])*EY+ERoundZ(x[2])*EZ*EZ)].flatten()
    def value_shape(self):
        return ((3,3,3,3))

```



```
#####
##### PERIODIC BOUNDARY CONDITIONS
#####
class PeriodicDomain(SubDomain):
    def inside(self, x, on_boundary):
        return bool((near(x[0], 0.) or near(x[1], 0.) or near(x[2], 0.)) and
                    not(near(x[0], 0) and (near(x[1], Y_Len) or near(x[2], Z_Len))) and
                    not(near(x[1], 0) and (near(x[0], X_Len) or near(x[2], Z_Len))) and
                    not(near(x[2], 0) and (near(x[0], X_Len) or near(x[1], Y_Len)))
                    and on_boundary)

    def map(self, x, y):
        if near(x[0], X_Len, DOLFIN_EPS) and near(x[1], Y_Len, DOLFIN_EPS) and
            near(x[2], Z_Len, DOLFIN_EPS):
            y[0] = x[0] - X_Len
            y[1] = x[1] - Y_Len
            y[2] = x[2] - Z_Len
        elif near(x[0], X_Len, DOLFIN_EPS) and near(x[1], Y_Len, DOLFIN_EPS):
            y[0] = x[0] - X_Len
            y[1] = x[1] - Y_Len
            y[2] = x[2]
        elif near(x[1], Y_Len, DOLFIN_EPS) and near(x[2], Z_Len, DOLFIN_EPS):
            y[0] = x[0]
            y[1] = x[1] - Y_Len
            y[2] = x[2] - Z_Len
        elif near(x[0], X_Len, DOLFIN_EPS) and near(x[2], Z_Len, DOLFIN_EPS):
            y[0] = x[0] - X_Len
            y[1] = x[1]
            y[2] = x[2] - Z_Len
        elif near(x[0], X_Len, DOLFIN_EPS): # Map the point at X = X_Len to X = 0
            y[0] = x[0] - X_Len
            y[1] = x[1]
            y[2] = x[2]
        elif near(x[1], Y_Len, DOLFIN_EPS): # Map the point at Y = Y_Len to Y = 0
            y[0] = x[0]
            y[1] = x[1] - Y_Len
            y[2] = x[2]
        elif near(x[2], Z_Len, DOLFIN_EPS): # Map the point at Z = Z_Len to Z = 0
            y[0] = x[0]
            y[1] = x[1]
            y[2] = x[2] - Z_Len
        else:
            y[0] = x[0]
            y[1] = x[1]
            y[2] = x[2]

pbc = PeriodicDomain()
#####
class PeriodicDomainYZ(SubDomain):
    #MAPS Y1 => Y0 and Z1 => Z0 TWO BOUNDARIES

    def inside(self, x, on_boundary):
        return bool((near(x[1], 0) or near(x[2], 0.)) and
                    (not ((near(x[1], Y_Len) and near(x[2], 0.)) or
                    (near(x[1], 0) and near(x[2], Z_Len)))) and on_boundary)

    def map(self, x, y):
        if near(x[1], Y_Len) and near(x[2], Z_Len):
            y[0] = x[0]
            y[1] = x[1] - Y_Len
            y[2] = x[2] - Z_Len
        elif near(x[1], Y_Len):
            y[0] = x[0]
            y[1] = x[1] - Y_Len
            y[2] = x[2]

```

```

elif near(x[2], Z_Len):
    y[0] = x[0]
    y[1] = x[1]
    y[2] = x[2] - Z_Len
else:
    y[0] = -1000
    y[1] = -1000
    y[2] = -1000
pbcyz = PeriodicDomainYZ()
#####

#Convective Heat transfer for Metal to Liquid Metal 40,000 (W/(m2K)) = 4e10 mW/m^2K = 4e4 mW/mm^2K
H          = Constant(4.0e4)
Ha         = Constant(9.1)
T_inf      = T_init

#Calculate the time Step
dt = Constant(T_Time/T_Steps)

# H = Constant(1000.0)

print("CFL : ", 5e6*float(dt)/(X_Len/MX))

# Specify the time for boundary condition Change
CoolIT = int(T_Steps/12)

# Set Quadrature degree for the Mesh Intergal
q_degree = 3
dx = dx(metadata={'quadrature_degree': q_degree})

# Create Function Spaces
if(Peridic == 1):
    ScalarSpace      = FunctionSpace(mesh, 'CG', 2)
    VectorSpace      = VectorFunctionSpace(mesh, 'CG', 2)
    TensorSpace      = TensorFunctionSpace(mesh, "CG", 1)
    StiffnessSpace    = TensorFunctionSpace(mesh, 'DG', 0, (3,3,3,3))
elif(Peridic == 2):
    ScalarSpace      = FunctionSpace(mesh, 'CG', 2)
    VectorSpace      = VectorFunctionSpace(mesh, 'CG', 2, constrained_domain=pbcyz)
    TensorSpace      = TensorFunctionSpace(mesh, "CG", 1)
    StiffnessSpace    = TensorFunctionSpace(mesh, 'DG', 0, (3,3,3,3))
elif(Peridic == 3):
    ScalarSpace      = FunctionSpace(mesh, 'CG', 2, constrained_domain=pbcyz)
    VectorSpace      = VectorFunctionSpace(mesh, 'CG', 2, constrained_domain=pbcyz)
    TensorSpace      = TensorFunctionSpace(mesh, "CG", 1)
    StiffnessSpace    = TensorFunctionSpace(mesh, 'DG', 0, (3,3,3,3))

# Create Location for Boundary Conditions
def X0(x, on_boundary):      return near(x[0],0,DOLFIN_EPS) and on_boundary
def X1(x, on_boundary):      return near(x[0],X_Len,DOLFIN_EPS) and on_boundary

def Y0(x, on_boundary):      return near(x[1],0,DOLFIN_EPS) and on_boundary
def Y1(x, on_boundary):      return near(x[1],Y_Len,DOLFIN_EPS) and on_boundary

def Z0(x, on_boundary):      return near(x[2],0,DOLFIN_EPS) and on_boundary
def Z1(x, on_boundary):      return near(x[2],Z_Len,DOLFIN_EPS) and on_boundary

def CenterXYZ(x):            return (near(x[0],X_Len/2,DOLFIN_EPS) and near(x[1],Y_Len/2,DOLFIN_EPS) and
                                near(x[2],Z_Len/2,DOLFIN_EPS))
def CenterX1YZ(x):           return (near(x[0],X_Len,DOLFIN_EPS) and near(x[1],Y_Len/2,DOLFIN_EPS) and
                                near(x[2],Z_Len/2,DOLFIN_EPS))

```

```

# Mark Boundary For Surface Integrals
boundary_subdomains = MeshFunction("size_t", mesh, mesh.topology().dim() - 1)
boundary_subdomains.set_all(0)
AutoSubDomain(X0).mark(boundary_subdomains, 1)
AutoSubDomain(X1).mark(boundary_subdomains, 2)
AutoSubDomain(Y0).mark(boundary_subdomains, 3)
AutoSubDomain(Y1).mark(boundary_subdomains, 4)
AutoSubDomain(Z0).mark(boundary_subdomains, 5)
AutoSubDomain(Z1).mark(boundary_subdomains, 6)
dss      = ds(subdomain_data=boundary_subdomains)

#####
##### BOUNDARY CONDITION SPECIFICATION
#####

bcX1      = DirichletBC(VectorSpace.sub(0), Constant(0.0), X1)
bcC1      = DirichletBC(VectorSpace.sub(1), Constant(0.0), CenterX1YZ, method="pointwise")
bcC2      = DirichletBC(VectorSpace.sub(2), Constant(0.0), CenterX1YZ, method="pointwise")
bcT1      = DirichletBC(ScalarSpace, T_init, X1)
bcTd1     = DirichletBC(ScalarSpace, Constant(0.0), X1)

# Create Boundary Condition List

# Primary Variables Boundary Conditions
bcT        = [bcT1]
bcU        = [bcC1, bcC2, bcX1]

bcTa       = [bcTd1]
bcUa       = [bcC1, bcC2, bcX1]

# Create Reoriented Stiffness Matrix
CG         = Function(StiffnessSpace)
CG         = BuildGlobalStiffnessMatrixE(degree=0)
CC         = as_tensor(CC)

#####
# Define Functions for Form Equations

# Define total strain tensor as a function of gradient u
def epsilon(u):
    return sym(grad(u))

# Define Thermal strain tensor as a function of gradient Alpha and Temperature Difference
def epsilon_T(T):
    return as_tensor(Alpha[i,j]*(T - T_Ref),(i,j))

# Define the Stress Tensor as a function of the Mechanical Stresses and Thermal Strains
def sigma(u,T):
    return as_tensor(CG[i,j,k,l]*(epsilon(u)[k,l]-epsilon_T(T)[k,l]),(i,j))

# Define Internal Heat Transfer Flux as a function of Thermal Conductivity Tensor and Gradient of Temperature
def q(T):
    return (-dot(K,grad(T)))

# Define the Coefficient of Thermal Stress Tensor Beta()
def Beta():
    return as_tensor(CG[i,j,k,l]*Alpha[k,l],(i,j))

# Create Indices for iteration within Ternsors
i,j,k,l    = indices(4)

```

```
#Create the Trial/Test Function
```

```
T_dot = TrialFunction(ScalarSpace)
vT     = TestFunction(ScalarSpace)
u       = Function(VectorSpace)
ud_dot = Function(VectorSpace)
vu      = TestFunction(VectorSpace)
```

```
#Create Previous Step Functional Variables
```

```
T_n     = Function(ScalarSpace) #Previous Step Temperature
T_dotn  = Function(ScalarSpace) #Previous Step Temperature Rate
```

```
un      = Function(VectorSpace) #Previous Step Displacement
u_dot   = Function(VectorSpace) #Current Step Velocity
u_dotn  = Function(VectorSpace) #Previous Step Velocity
ud_dotn = Function(VectorSpace) #Previous Step Acceleration
```

```
T_n     = project(T_init,ScalarSpace) # Initialize Variable
```

```
#Create the Weak/Variational Formulation
```

```
#Primary Solver for Heat Equations
```

```
#####
T = TrialFunction(ScalarSpace)
#####
if(Delta == 0.0):
    ThermalForm = rho*Cv*((2*(T-T_n)/dt)-T_dotn)*vT*dx + Flux*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                - dot(q(T),grad(vT))*dx
```

```
else:
```

```
    ThermalForm = rho*Cv*((2*(T-T_n)/dt)-T_dotn)*vT*dx + Flux*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                - dot(q(T),grad(vT))*dx + Delta*(T_Ref*inner(Beta(),epsilon(u_dotn)))*vT*dx
```

```
#####
```

```
aT, LT = lhs(ThermalForm), rhs(ThermalForm)
```

```
T = Function(ScalarSpace)
```

```
thermalproblem = LinearVariationalProblem(aT, LT, T, bcT)
```

```
thermsolver = LinearVariationalSolver(thermalproblem)
```

```
#####
```

```
#Primary Solver for Elasticity Equations
```

```
#####
```

```
u = TrialFunction(VectorSpace)
```

```
#####
```

```
MechanicalForm = rho*dot(((2/dt)*((2/dt)*(u - un) - 2*u_dotn) - ud_dotn),vu)*dx - dot(Traction,vu)*dss(1)
                + inner(sigma(u,T),epsilon(vu))*dx
```

```
#####
```

```
aU, LU = lhs(MechanicalForm), rhs(MechanicalForm)
```

```
u = Function(VectorSpace)
```

```
mechproblem = LinearVariationalProblem(aU, LU, u, bcU)
```

```
mechsolver = LinearVariationalSolver(mechproblem)
```

```
#####
```

```
#Secondary Solver for Calculating Acceleration
```

```
#####
```

```
ud_dot = TrialFunction(VectorSpace)
```

```
#####
```

```
AccForm = rho*dot((ud_dot),vu)*dx - dot(Traction,vu)*dss(1) + inner(sigma(u,T),grad(vu))*dx
```

```
#####
```

```
aUdd, LUdd = lhs(AccForm), rhs(AccForm)
```

```
ud_dot = Function(VectorSpace)
```

```
acc_problem = LinearVariationalProblem(aUdd, LUdd, ud_dot, bcUa)
```

```
acc_solver = LinearVariationalSolver(acc_problem)
```

```

#Secondary Solver to Calculate Temperature Rate
#####
T_dot = TrialFunction(ScalarSpace)
#####
if(Delta == 0.0):
    T_dot_Form          = rho*Cv*(T_dot)*vT*dx + Flux*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                        - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                        - dot(q(T),grad(vT))*dx
else:
    T_dot_Form          = rho*Cv*(T_dot)*vT*dx + Flux*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                        - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                        - dot(q(T),grad(vT))*dx + Delta*(T_Ref*inner(Beta(),epsilon(u_dot)))*vT*dx
#####
aTD, LTD              = lhs(T_dot_Form), rhs(T_dot_Form)
T_dot                 = Function(ScalarSpace)
T_dot_problem         = LinearVariationalProblem(aTD, LTD, T_dot, bcTa)
T_dot_solver          = LinearVariationalSolver(T_dot_problem)
#####

# Create Output File
fileResults = XDMFFile(OUTPUT_FOLDER+'/'+OUTPUT_FILE+'.xdmf')
fileResults.parameters["flush_output"] = True
fileResults.parameters["functions_share_mesh"] = True

StrfileResults = XDMFFile(OUTPUT_FOLDER+'/'+OUTPUT_FILE+'_str_str.xdmf')
StrfileResults.parameters["flush_output"] = True
StrfileResults.parameters["functions_share_mesh"] = True

#Rename Output Variables for Writing
u.rename("Displacement", "label")
T.rename("Temperature", "label")

#Specify the Cumulative Time Variable and Assign as Zero
t = 0.

# Write Output at Time equals Zero to check for initial errors if any.
# fileResults.write(T, t)
# fileResults.write(u, t)

#Run the time loop for Steps Specified
for ii in range(T_Steps):

    #Create Timer for Iteration Time
    iter_time = time.perf_counter()

    #Cumulative Time Increment for Output
    t += float(dt)

    #####
    # BOUNDARY CONDITION UPDATE DURING RUN
    #####
    if (ii == CoolIT):
        #Primary Solver for Heat Equations
        #####
        T                = TrialFunction(ScalarSpace)

```

```

if(Delta == 0.0):
    ThermalForm = rho*Cv*((2*(T-T_n)/dt)-T_dotn)*vT*dx - Ha*(T - T_init)*vT*dss(1)
                - Ha*(T - T_init)*vT*dss(3) - Ha*(T - T_init)*vT*dss(4)
                - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                - dot(q(T),grad(vT))*dx
else:
    ThermalForm = rho*Cv*((2*(T-T_n)/dt)-T_dotn)*vT*dx - Ha*(T - T_init)*vT*dss(1)
                - Ha*(T - T_init)*vT*dss(3) - Ha*(T - T_init)*vT*dss(4)
                - Ha*(T - T_init)*vT*dss(5) - Ha*(T - T_init)*vT*dss(6)
                - dot(q(T),grad(vT))*dx + Delta*(T_Ref*inner(Beta(),epsilon(u_dotn)))*vT*dx

aT, LT = lhs(ThermalForm), rhs(ThermalForm)
T = Function(ScalarSpace)
thermalproblem = LinearVariationalProblem(aT, LT, T, bcT)
thermsolver = LinearVariationalSolver(thermalproblem)

#Secondary Solver to Calculate Temperature Rate
T_dot = TrialFunction(ScalarSpace)

if(Delta == 0.0):
    T_dot_Form = rho*Cv*(T_dot)*vT*dx - Ha*(T - T_init)*vT*dss(1)
                - Ha*(T - T_init)*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5)
                - Ha*(T - T_init)*vT*dss(6) - dot(q(T),grad(vT))*dx
else:
    T_dot_Form = rho*Cv*(T_dot)*vT*dx - Ha*(T - T_init)*vT*dss(1)
                - Ha*(T - T_init)*vT*dss(1) - Ha*(T - T_init)*vT*dss(3)
                - Ha*(T - T_init)*vT*dss(4) - Ha*(T - T_init)*vT*dss(5)
                - Ha*(T - T_init)*vT*dss(6) - dot(q(T),grad(vT))*dx
                + Delta*(T_Ref*inner(Beta(),epsilon(u_dot)))*vT*dx

aTD, LTD = lhs(T_dot_Form), rhs(T_dot_Form)
T_dot = Function(ScalarSpace)
T_dot_problem = LinearVariationalProblem(aTD, LTD, T_dot, bcTa)
T_dot_solver = LinearVariationalSolver(T_dot_problem)

T.rename("Temperature", "label")

#####

#Print Iteration Count
print("Increment:" + str(ii+1),"Time Increment:",float(dt),"Cumulative Time:",t)

#Step 1: Solve Heat Equation at n+1 with Coupling term velocity at n
print("Step 1: Solving Heat Equation")
thermsolver.solve()

#Step 2: Solve Elasticity Equation at n+1
print("Step 2: Solving Elasticity Equation")
mechsolver.solve()

#Step 3: Solve and Update Acceleration at n+1
print("Step 3: Solving Acceleration Update")
acc_solver.solve()

#Step 4: Update Velocity at n+1
print("Step 4: Update Velocity")
u_dot.assign(u_dotn + float(dt/2)*(ud_dot + ud_dotn))

print("Step 5: Solving Temperature Rate Update")
T_dot_solver.solve()

```

```

print("Step 6: Update Temperature and Displacement Field Variables")
un.assign(u)
u_dotn.assign(u_dot)
ud_dotn.assign(ud_dot)
T_n.assign(T)
T_dotn.assign(T_dot)

print("Step 7: Displacement-Temperature Output")
fileResults.write(u, t)
fileResults.write(T, t)
if((ii+1)%StressOutput==0):
    print("Step 8: Stress-Strain Output")
    stress = project(sigma(u,T),TensorSpace)
    strain = project(epsilon(u),TensorSpace)
    stress.rename("Stress","label")
    strain.rename("Strain","label")
    StrfileResults.write(stress, t)
    StrfileResults.write(strain, t)

print("\n\n-----> Iteration Time: ", time.perf_counter() - iter_time,"seconds\n\n")

print('Program Complete')
print("\n\n-----> Total Calculation Time: ", time.perf_counter() - start_time,"seconds\n\n")

```

BIBLIOGRAPHY

- [1] X. Li, W. Tan, (2018). Numerical investigation of effects of nucleation mechanisms on grain structure in metal additive manufacturing, *Computational Materials Science* 153 (2018) 159-169.
- [2] J.A. Koepf, M.R. Gotterbarm, M. Markl, C. Körner, (2018). 3D multi-layer grain structure simulation of powder bed fusion additive manufacturing, *Acta Materialia* 152 (2018) 119-126.
- [3] O. Zinovieva, A. Zinoviev, V. Ploshikhin, (2018). Three-dimensional modeling of the microstructure evolution during metal additive manufacturing, *Computational Materials Science* 141 (2018) 207-220
- [4] Z. Yang, S. Sista, J. Elmer, T. DebRoy, (2000). Three dimensional Monte Carlo simulation of grain growth during GTA welding of titanium, *Acta Materialia* 48(20) (2000) 4813-4825.
- [5] H.L. Wei, J.W. Elmer, T. DebRoy, (2017). Three-dimensional modeling of grain structure evolution during welding of an aluminum alloy, *Acta Materialia* 126 (2017) 413-425.
- [6] H. Wei, J. Elmer, T. DebRoy, (2017). Crystal growth during keyhole mode laser welding, *Acta Materialia* 133 (2017) 10-20
- [7] Theron M. Rodgers, Jonathan D. Madison, Veena Tikare, (2017). Simulation of metal additive manufacturing microstructures using kinetic Monte Carlo, *Computational Materials Science*, Volume 135, July 2017, Pages 78-89
- [8] C.K. Newman, M.M. Francois, (2016). An Implicit Approach to Phase Field Modeling of Solidification for Additively Manufactured Materials, Los Alamos Technical Report, LA-UR-16-24310, 2016.
- [9] A.J. Trainer, C.K. Newman, M.M. Francois, (2016). Overview of the Tusas Code for Simulation of Dendritic Solidification, Los Alamos Technical Report, LA-UR-16- 20078, 2016.
- [10] M.R. Dorr, J.L. Fattebert, M.E. Wickett, J.F. Belak, P.E.A. Turchi, (2010). A numerical algorithm for the solution of a phase-field model of polycrystalline materials *Journal of Computational Physics*, Volume 229, Issue 3, 1 February 2010, Pages 626-641
- [11] J.L. Fattebert, M.E. Wickett, P.E.A. Turchi, (2014). Phase-field modeling of coring during solidification of Au–Ni alloy using quaternions and CALPHAD input, *Acta Mater.* 62 (2014) 89.
- [12] T. Pusztai, G. Bortel, L. Gránásy, (2005). Phase field theory of polycrystalline solidification in three dimensions, *EPL (Europhys. Lett.)* 71 (1) (2005) 131.
- [13] Takuya Fujinaga, Yoshimi Watanabe, Yasushi Shibuta, (2020). Nucleation dynamics in Al solidification with Al-Ti refiners by molecular dynamics simulation, *Computational Materials Science*, Volume 182, September 2020, 109763
- [14] Avik Mahata, Mohsen Asle Zaeema, (2019). Effects of solidification defects on nanoscale mechanical properties of rapid directionally solidified Al-Cu Alloy: A large scale molecular dynamics study, *Journal of Crystal Growth*, Volume 527, 1 December 2019, 125255

- [15] C.A. Bronkhorst, S.R. Kalidindi, L. Anand, (1992). Polycrystalline plasticity and the evolution of crystallographic texture in FCC Metals, *Philos. Trans. R. Soc. Lond. A* 341 (1992) 443–477.
- [16] S.R. Kalidindi, C.A. Bronkhorst, L. Anand, (1992). Crystallographic texture evolution in bulk deformation processing of FCC metals, *J. Mech. Phys. Solids* 40 (1992) 537–569.
- [17] C.A. Bronkhorst, B.L. Hansen, E.K. Cerreta, J.F. Bingert, (2007). Modeling the microstructural evolution of metallic polycrystalline materials under localization conditions, *J. Mech. Phys. Solids* 55 (2007) 2351–2383.
- [18] B.L. Hansen, C.A. Bronkhorst, M. Ortiz, (2010). Dislocation subgrain structures and modeling the plastic hardening of metallic single crystals, *Modell. Simul. Mater. Sci. Eng.* 18 (2010) 055001.
- [19] C.A. Bronkhorst, A.R. Ross, B.L. Hansen, E.K. Cerreta, J.F. Bingert, (2010). Modeling and characterization of grain scale strain distribution in polycrystalline tantalum, *Comput. Mater. Continua* 17 (2010) 149–174.
- [20] B.L. Hansen, I.J. Beyerlein, C.A. Bronkhorst, E.K. Cerreta, D. Dennis-Koller, (2013). A dislocation-based multi-rate single crystal plasticity model, *Int. J. Plasticity* 44 (2013) 129–146.
- [21] H Moulinec, P Suquet, (1994). A fast numerical method for computing the linear and nonlinear mechanical properties of composites, *Comptes rendus de l'Académie des sciences. Série II, Mécanique, physique, chimie, astronomie*, Volume 318-1, 1994, pp1417-1423
- [22] PM Suquet, (1987). Elements of Homogenization for Inelastic Solid Mechanics, *Homogenization techniques for composite media*, 193-278
- [23] R.A. Lebensohn, (2001). N-site modelling of a 3D visco-plastic polycrystal using Fast Fourier Transform, *Acta Materialia* 49, pp 2723-2737 (2001). 173 citations. (PDF)
- [24] R A. Lebensohn and C. N. Tome, (2009). Visco-Plastic Self-Consistent (VPSC) Crystal Plasticity Model, 2009, (https://public.lanl.gov/lebenso/VPSC7c_manual.pdf)
- [25] R A. Lebensohn and C. N. Tome, (1993). A self-consistent anisotropic approach for the simulation of plastic deformation and texture development of polycrystals: Application to zirconium alloys, *Acta Metallurgica et Materialia* 41, pp 2611-2624 (1993)
- [26] R. A. Lebensohn, Anand K. Kanjarla, Philip Eisenlohr, (2012). An elasto-viscoplastic formulation based on fast Fourier transforms for the prediction of micromechanical fields in polycrystalline materials, *International Journal of Plasticity*, Volumes 32–33, May 2012, Pages 59-69
- [27] Kohar, C.P., Bassani, J.L., Mishra, R.K., Inal, K., (2018). A Multiscale Model to Incorporate Texture Evolution into Phenomenological Plasticity Models, *Journal of Physics: Conference Series*, 2018
- [28] Kohar, C.P., Bassani, J.L. Rahme, A., Muhammad, W., Mishra, R.K., Inal, K , (2017). A new multi-scale framework to incorporate microstructure evolution in phenomenological plasticity: Theory, explicit finite element formulation, implementation and validation, *International Journal of Plasticity*, 2017

- [29] Philipp Kürsteiner, Markus B. Wilms, Andreas Weisheit, Pere Barriobero-Vila, Eric A. Jägle, Dierk Raabe, (2017). Massive nanoprecipitation in an Fe-19Ni-xAl maraging steel triggered by the intrinsic heat treatment during laser metal deposition, *Acta Materialia*, Volume 129, 1 May 2017, Pages 52-60
- [30] Jan Haubrich, Joachim Gussone, Pere Barriobero-Vila, Philipp Kürsteiner, Eric A. Jägle, Dierk Raabe, Norbert Schell, Guillermo Requena, (2019). The role of lattice defects, element partitioning and intrinsic heat effects on the microstructure in selective laser melted Ti-6Al-4V, *Acta Materialia*, Volume 167, 1 April 2019, Pages 136-148
- [31] Tiago A. Rodrigues, V. Duarte, Julian A. Avila, Telmo G. Santos, R. M. Miranda, J. P. Oliveiraa, (2019). Wire and arc additive manufacturing of HSLA steel: Effect of thermal cycles on microstructure and mechanical properties, *Additive Manufacturing*, Volume 27, May 2019, Pages 440-450
- [32] Jingjing Yang, Hanchen Yu, Jie Yin, Ming Gao, Zemin Wang, Xiaoyan Zeng, (2016). Formation and control of martensite in Ti-6Al-4V alloy produced by selective laser melting, *Materials & Design*, Volume 108, 15 October 2016, Pages 308-318
- [33] Yang Liu, Jian Zhang, Zhicong Pang, (2018). Numerical and experimental investigation into the subsequent thermal cycling during selective laser melting of multi-layer 316L stainless steel, *Optics & Laser Technology*, Volume 98, 1 January 2018, Pages 23-32
- [34] H. Z. Zhong, C. G. Li, X. Y. Zhang, J. F. Gu., (2018). The graded microstructures evolving with thermal cycles in pure copper processed by laser metal deposition, *Materials Letters*, Volume 230, 1 November 2018, Pages 215-218
- [35] B. Zheng, Y. Zhou, J. E. Smugeresky, J. M. Schoenung, E. J. Lavernia, (2008). Thermal Behavior and Microstructure Evolution during Laser Deposition with LENS: Part I Numerical Calculation, *Metallurgical and Materials Transactions A* Vol 39 No 9 Sept 2008 pp 228-2236; doi:10.1007/s11661-008-9557-7.
- [36] B. Zheng, Y. Zhou, J. E. Smugeresky, J. M. Schoenung, E. J. Lavernia, (2008). Thermal Behavior and Microstructure Evolution during Laser Deposition with LENS: Part II Experimental Investigation and Discussion, *Metallurgical and Materials Transactions A* Vol 39 No 9 Sept 2008 pp 2237-2245; doi:10.1007/s11661-008-9566-6.
- [37] Hua Tan, Zhiyu Wang, Yiming Jiang, Yanze Yang, Bo Deng, Hongmei Song, Jin Li, (2012). Influence of welding thermal cycles on microstructure and pitting corrosion resistance of 2304 duplex stainless steels, *Corrosion Science*, Volume 55, February 2012, Pages 368-377
- [38] Xiaoqing Wang, Kevin Chou, (2017). Effects of thermal cycles on the microstructure evolution of Inconel 718 during selective laser melting process, *Additive Manufacturing*, Volume 18, December 2017, Pages 1-14
- [39] Yu Wu, Shuquan Zhang, Xu Cheng, Huaming Wang, (2019). Investigation on solid-state phase transformation in a Ti-47Al-2Cr-2V alloy due to thermal cycling during laser additive manufacturing process, *Journal of Alloys and Compounds*, Volume 799, 30 August 2019, Pages 325-333

- [40] Adel Belkharchouche, Achraf Boudiaf, Mohamed El Amine Belouchrani,(2017). A comparison between 2017A and 4047A aluminum alloys microstructure changes under thermal fatigue loading, *Materials Science and Engineering: A*, Volume 689, 24 March 2017, Pages 96-102
- [41] Xinyu Yang, Richard A.Barrett, MingmingTong, Noel M.Harrison, Sean B.Leen, (2020). Prediction of Microstructure Evolution for Additive Manufacturing of Ti-6Al-4V, *Procedia Manufacturing*, Volume 47, 2020, Pages 1178-1183
- [42] Utkarsh Ayachit,(2015). *The ParaView Guide: A Parallel Visualization Application*, 2015, Kitware Incorporated, ISBN-10: 1930934297
- [43] KitwarePublic, Paraview (https://www.paraview.org/Wiki/The_ParaView_Tutorial)
- [44] J. Bonet, & R. Wood, (2008) *Stress and Equilibrium*. In *Nonlinear Continuum Mechanics for Finite Element Analysis* (pp. 134-154). Cambridge: Cambridge University Press. doi:10.1017/CBO9780511755446.006
- [45] J. Lemaitre,& J. Chaboche, (1990). *Linear elasticity, thermoelasticity and viscoelasticity*. In *Mechanics of Solid Materials* (pp. 121-160). Cambridge: Cambridge University Press. doi:10.1017/CBO9781139167970.008
- [46] O.C. Zienkiewicz, R.L. Taylor, J.Z. Zhu, (2005). 6 - Problems in linear elasticity, *The Finite Element Method Set (Sixth Edition)*, Volume 1, 2005, Pages 187-228, <https://doi.org/10.1016/B978-075066431-8.50174-0>
- [47] C. Farhat, K. C. Park, & Dubois-Pelerin, Y. (1991). An unconditionally stable staggered algorithm for transient finite element analysis of coupled thermoelastic problems. *Computer Methods in Applied Mechanics and Engineering*, 85(3), 349-365. [https://doi.org/10.1016/0045-7825\(91\)90102-C](https://doi.org/10.1016/0045-7825(91)90102-C)
- [48] M. A. Kulesh, V. P. Matveenko & I. N. Shardakov, (2001). Exact Analytical Solution of the Kirsch Problem within the Framework of the Cosserat Continuum and Pseudocontinuum, *Journal of Applied Mechanics and Technical Physics* volume 42, pages687–695
- [49] Eran Grosu, Isaac Harari, (2001). Stability of semidiscrete formulations for elastodynamics at small time steps. *Finite Elements in Analysis and Design*, Elsevier, 2007, 43 (6-7), pp.533 - 542. doi:10.1016/j.finel.2006.12.006. (hal-01634271)
- [50] A. Idesman,(2011). *Accurate Time Integration of Linear Elastodynamics Problems*. *Cmes-Computer Modeling in Engineering & Sciences*, vol. 71, no. 2, Jan. 2011, pp. 111–48.
- [51] M. A. Biot,(1956). Thermoelasticity and Irreversible Thermodynamics, *Journal of Applied Physics* 27, 240 (1956); doi: 10.1063/1.1722351
- [52] M Bonnet, Attilio Frangi, Christian Reym (2014). *The finite element method in solid mechanics*. McGraw Hill Education, pp.365, 2014, 97888838674464
- [53] V. I. Danilovskaya, (1950). Thermal Stresses in an Elastic Half Space Arising After a Sudden Heating of Its Boundary (in Russian), *Prikladnaya Matematika i Mekhanika*, Vol. 14, No. 3, May-June 1950, pp.316-318.

- [54] V. I. Danilovskaya, (1952). On a Dynamical Problem of Thermoelasticity (in Russian), *Prikladnaya Matematika iMechanika*, Vol. 16, No. 3, May-June 1952, pp. 341-344.
- [55] L. Dubrovinsky, (2002). Thermal Expansion and Equation of State, *Encyclopedia of Materials: Science and Technology*, 2002, DOI: 10.1016/B0-08-043152-6/01817-9
- [56] B. A. Boley, I. S. Tolins, (1962). Transient Coupled Thermoelastic Boundary Value Problems in the Half-Space, *J. Appl. Mech.* Dec 1962, 29(4): 637-646
- [57] Muki, R. and Breuer, S., (1962). Coupling Effects in a Transient Thermoelastic Problem, *Osterreichisches Ingenieur-Archiv*, Vol. 16, No. 4, May 1962, pp. 346-36S.
- [58] R. E. Nickel, J. L. Sackman, (1968). Approximate Solutions in Linear, Coupled Thermoelasticity, *J. Appl. Mech.* Jun 1968, 35(2): 255-266 (12 pages) <https://doi.org/10.1115/1.3601189>
- [59] Sternberg, Eli, and J. G Chakravorty, (1958). On Inertia Effects in a Transient Thermoelastic Problem. Providence, R.I.: Division of Applied Mathematics, Brown University, 1958.
- [60] R. A. Lebensohn, Anthony D. Rollett, (2020). Spectral methods for full-field micromechanical modelling of polycrystalline materials, *Computational Materials Science*, Volume 173, 15 February 2020, 109336
- [61] D. Gonzalez, J.F. Kelleher, J. Quinta da Fonseca, P.J. Withers, (2012). Macro and intergranular stress responses of austenitic stainless steel to 90° strain path changes, *Materials Science and Engineering: A*, V 546, 2012, p263-271, <https://doi.org/10.1016/j.msea.2012.03.064>.