Department of Civil Engineering and Architecture
University of Pavia

Building, Architecture and Town Planning
Université libre de Bruxelles

Optimized Material Deposition for Extrusion-Based Additive Manufacturing of Structural Components

Author:
Varun Murugan

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

at the

University of Pavia, Italy
and
Université libre de Bruxelles, Belgium

Supervisors:
Prof. Ferdinando Auricchio
Università degli Studi di Pavia
Prof. Thierry J. Massart
Université libre de Bruxelles

Co-Advisors:
Prof. Peter Berke
Université libre de Bruxelles
Prof. Stefania Marconi
Università degli Studi di Pavia
Dr. Gianluca Alaimo
Università degli Studi di Pavia

XXXIV Cycle (2018 - 2021)

Abstract

Doctor of Philosophy

Optimized Material Deposition for Extrusion-Based Additive Manufacturing of Structural Components

by Varun MURUGAN

Fused Filament Fabrication (FFF) is an extrusion-based Additive Manufacturing method widely used in various industries for prototyping applications. Since FFF's invention in the 1980s, this technology has undergone numerous advancements in terms of available material, software and equipment. Nevertheless, FFF parts are often limited to non-critical, non-load-bearing applications, partly because the existing design and manufacturing tools do not capitalize on FFF's high design freedom. In particular, the slicing software that generates printing instructions prints filaments in predefined patterns and cannot realize filament paths that are locally optimized to enhance the structural performance, thus, hindering the best utilization of FFF parts for load-bearing purposes.

The current thesis aims to develop dedicated tools for designing and printing optimized filament paths for improved structural performance from FFF parts. First, an algorithmic framework that optimizes the in-plane filament paths for minimum compliance of FFF structures is presented, particularly focusing on obtaining production-ready design solutions by including the manufacturing constraints in the optimization process. Then, a new filament deposition algorithm is proposed to address the manufacturability issues observed in existing optimization strategies. The proposed algorithm accepts the point-wise orientation fields resulting from the optimization procedures and deploy filaments along the orientation fields to realize optimized designs.

Both the contributions facilitate the seamless production of structural FFF parts, which we prove by applying them to design structurally-informed filament trajectories and printing the parts.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska Curie grant Agreement No 764636.

I gratefully acknowledge the continuous support of all my supervisors and co-advisors in completing this project.

First of all, I would like to thank Prof. Ferdinando Auricchio for offering me the chance to work on this project. Prof. Auricchio was always approachable and found time to advise me despite his highly hectic schedule. I enjoyed working with him and especially thank him for his research, writing and presenting tips.

Then, I would like to convey my deepest gratitude to Prof. Thierry J. Massart, whose valuable insights helped shape the course of this project. Prof. Massart was also incredibly supportive when I moved to Belgium and faced solitary confinement during the COVID breakouts.

I also want to convey my special thanks to Prof. Peter Berke at ULB, Belgium, for showing a particular interest in my work. Discussions with Prof. Berke were valuable and helped me to advance the project in a systematic way. Besides, I am grateful for his efforts in quickly and effectively revising the manuscripts.

Many thanks to Prof. Stefania Marconi and Dr. Gianluca Alaimo for introducing me to the field of 3D printing. Without their support, I could not have quickly learnt the practicalities of 3D-printing technology. I am also thankful for their consistent involvement in the project and their countless revisions to the manuscript to make it high-quality work.

My special thanks to Stefano Steve Serioli at the ProtoLab in University of Pavia for his help in 3D printing the specimens. He has immense knowledge of 3D printing techniques, which I greatly admire.

I want to convey my special thanks to Sonia Padovan and Dr. Alberto Cattenone at the University of Pavia for their assistance in bureaucratic activities.

Finally, I thank my parents, wife, and friends for supporting and encouraging me to pass the last three years of hardship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Additive Manufacturing (AM) refers to a group of manufacturing technologies that became popular in the early 1980s for rapid prototyping of industrial components. Since then, AM has witnessed many advancements in its range of materials, software, and equipment to evolve into one of the notable classes of manufacturing technologies. Today, AM enables rapid manufacturing of end-use products in various industries, including automotive, aerospace, energy, medical, consumer, fashion, culinary etc., and holds a market worth more than $12 billion (Figure 1.1). Recognizing the growing potential of AM in the near future (Varotsis, 2019; Roberts, 2020; Roberts, 2021), this thesis constitutes an effort to widen the applicability of AM parts in structural applications, especially focusing upon a very common AM technology called, *Fused Filament Fabrication*.

The present chapter gives an overview of the subject dealt with in the thesis. Section 1.1 introduces the concept of Additive Manufacturing, its working principle and classification. Section 1.2 explains the working of a specific AM technology, called Fused Filament Fabrication (FFF), and lists the characteristics of its printed parts. Then, Section 1.3 presents the central theme of this thesis, i.e., the need for optimized material deposition in FFF parts for superior structural performance. Section 1.4 presents a literature study on the existing techniques to optimize the filament paths. Finally, Section 1.5 explains the objectives of this work, introducing two new contributions of the thesis.

1.1 Additive Manufacturing: an overview

Additive Manufacturing is the official term used in ISO/ASTM standards to describe 3D printing. It is defined as "the process of joining materials to make parts from 3D model data, usually layer upon layer, as opposed to subtractive manufacturing and formative manufacturing methodologies"

(ASTM, 2012). AM offers several advantages that the traditional manufacturing methods do not. Some of them are listed below.

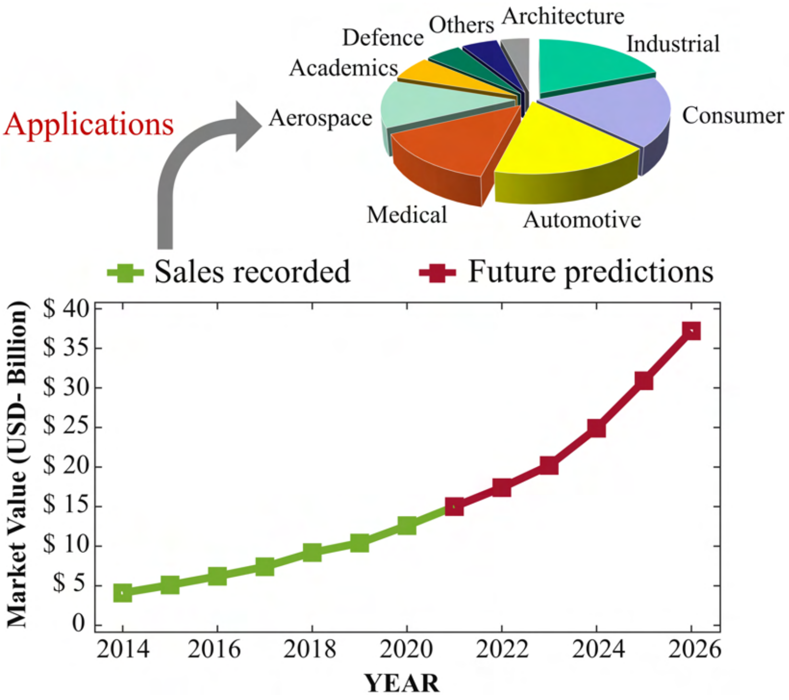


FIGURE 1.1: Additive manufacturing market growth and applications (Cotteleer and Joyce, 2014; Roberts, 2021)

- High design freedom: AM can produce almost any complex shape that cannot be produced by other methods, for example, lattice structures (Klahn, Leutenecker, and Meboldt, 2015; Jared et al., 2017; Ngo et al., 2018).
- No assembly: parts that require assembling with traditional methods can be produced as a single component (Cuellar et al., 2018; Ngo et al., 2018).
- Reduced lead times: the transition from design to product is rapid (Attaran, 2017). AM does not require any intermediate step between design and production thanks to its digital nature.
- Reduced material usage: due to the addition of material, the material waste is minimal (Klahn, Leutenecker, and Meboldt, 2015; Peng et al., 2018; Ngo et al., 2018).
- Environmental impact: in comparison to the traditional methods, AM is a greener technology. It is less polluting and consumes lesser resources (Peng et al., 2018).

- On demand production: since designs can be quickly converted into physical products, the need for inventory can be reduced or eliminated (Klahn, Leutenecker, and Meboldt, 2015; Attaran, 2017).

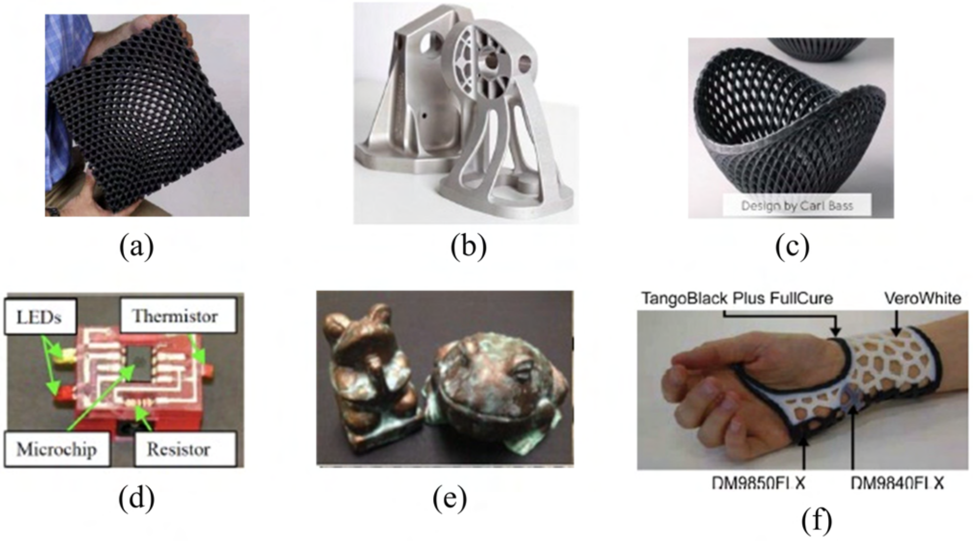


FIGURE 1.2: Parts produced with different AM technologies. (a) Fused Filament Fabricated lattice with self-collimation properties (ability to force light beams in specific direction) , (b) Topology-optimized brackets from Direct Metal Laser Sintering, (c) Decorative bowl from Binder Jetting, (d) Timer circuit from Stereolithography, (e) Frog and toad models from Sheet lamination, (f) Customized splint from Material Jetting (Thompson et al., 2016)

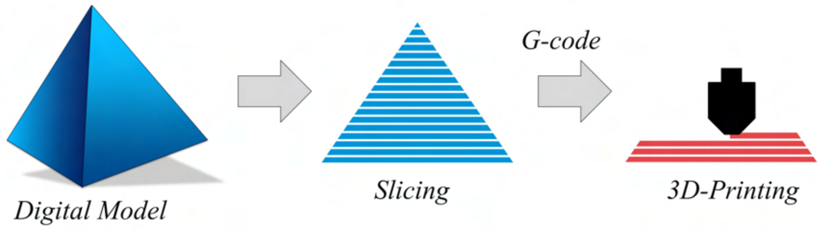


FIGURE 1.3: 3D-Printing basic workflow. A 3D model is sent to a slicing software that slices the model into layers and generate printing instructions to print the part (Pandey, Reddy, and Dhande, 2003).

AM does not refer to a single technique; it is a group of technologies working through a common principle of material addition. Table 1.1 lists seven different methods of AM, each having its own process, range of materials and application (Calignano et al., 2017). Figure 1.2 shows some of the parts printed using different AM technologies. Nevertheless, all AM methods follow the *Model-Slice-Print* strategy described in Figure 1.3 to produce components (Pandey, Reddy, and Dhande, 2003). The workflow of AM starts with the digital model of the part to be produced. The model, created in a 3D-modeling software, is imported into a slicing software and converted into layers of toolpath instructions in the form of a G-code. The G-code is a widely used programming format for controlling computer aided manufacturing process (for e.g., CNC machines and 3D printers). Once the G-code is available, it is supplied to an AM machine for carrying out the toolpath instructions that finally yield a physical replica of the digital model. Appendix A provides a sample G-code for better understanding.

TABLE 1.1: Classification of Additive Manufacturing (Calignano et al., 2017)

Methods	Process	Materials
Binder Jetting	Binding liquid selectively deposited to join powder	plastic, ceramics, metal
Direct Energy Deposition	Metal wire or powder melted and fused using laser beam	metals
Material Extrusion	Material deposited as strands and fused with each other	polymers, concrete
Material Jetting	Droplets of material deposited and cooled	wax, polymers, concrete, wax
Powder Bed Fusion	Thermal energy sources fuses regions of powder bed	metals and ceramics
Sheet Lamination	Thin sheets of material stacked and bonded	paper, polymer, composite, metals
Vat photo-polymerisation	Liquid photopolymers deposited and cured with UV lights	photopolymers

1.2 Fused Filament Fabrication: process and part characteristics

Among the various methods listed in Table 1.1, this thesis focuses upon Fused Filament Fabrication (FFF), which is an extrusion-based technology widely adopted in industries for prototyping applications (Singh et al., 2020). FFF works with thermoplastic materials and is far less expensive than other technologies due to its relatively simple process.

Multiple brands of FFF machines exist, ranging from simple desktop-operated to highly sophisticated industrial machines. A typical FFF machine comprises a heating unit, an extruder/nozzle and a platform. The heating unit and the nozzle are mounted in a frame that exhibits planar motion (X-Y axis), while the platform moves in an up-and-down (Z axis) fashion. For printing material, a thermoplastic filament is fed to the heating unit that partially melts the material and extrudes it via the nozzle onto the platform. The predetermined movement instructions provided by the G-Code are processed by the microcontroller, which signals the servo-motors to move the nozzle, extruding filaments and creating a 2D layer. Once a layer is finished, the platform moves down, and the nozzle extrudes another layer of filaments supported by the previously deposited layer. The layer-wise stacking repeats and all the extruded filaments cool down and bond to produce a solid part resembling the virtual input model. Figure 1.4 shows the construction of FFF machine, the extrusion of filaments and their cross-section.

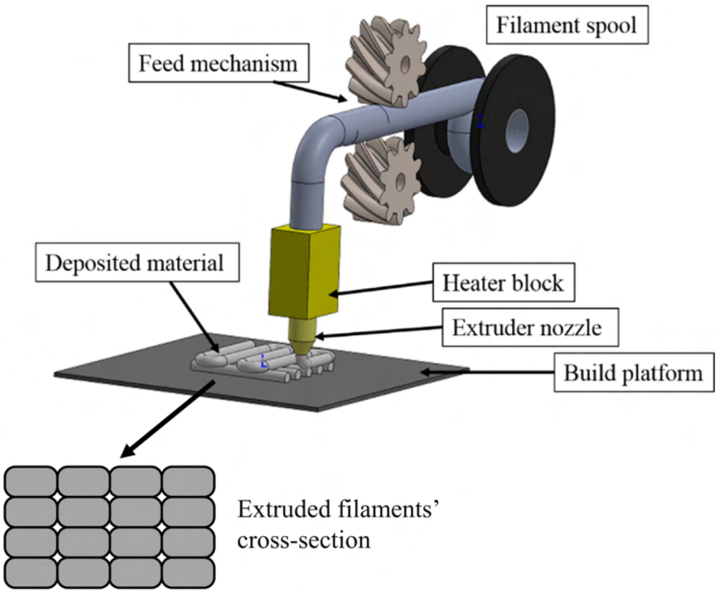


FIGURE 1.4: Working Principle of Fused Filament Fabrication (Cuan-Urquizo et al., 2019)

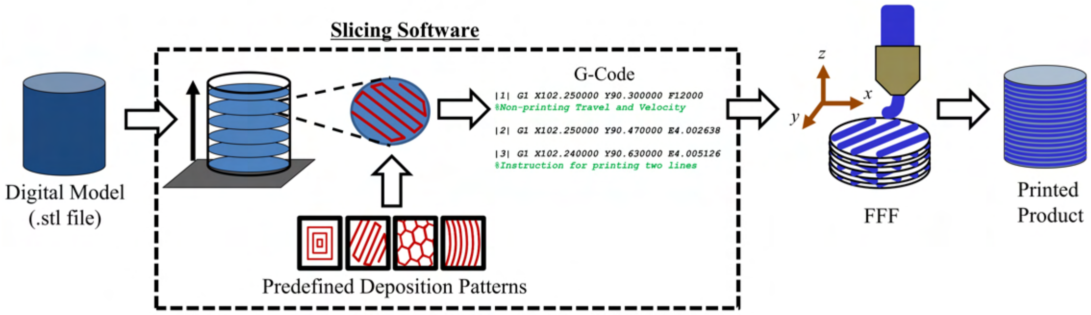
The complex multi-physical filament extrusion process produces parts with anisotropy (Ahn et al., 2002) and inter-filament voids (Li et al., 2002) that degrade the parts' properties relative to the isotropic feedstock material. Thus, an FFF part has stronger mechanical properties (stiffness and strength) along the filament direction and weaker properties along its transverse directions due to the weak bonding between filaments. However, the degree of anisotropy varies depending upon the material used. For example, a pure thermoplastic ABS is weakly anisotropic with respect to stiffness. The ratio of its Young's modulii along the filament direction (E_L) to the transverse direction (E_T) is around $E_L/E_T \approx 1.05$ to 1.2 (Alaimo et al., 2017). In contrast, when reinforced with short fibers, 3D printed ABS parts become strongly anisotropic and can reach up to $E_L/E_T \approx 8$ (Tekinalp et al., 2014). Besides the direction of filament extrusion, its density, and the printing material, an FFF part behaviour is also influenced by an array of building and process parameters (Fayazbakhsh, Movahedi, and Kalman, 2019), the individual effects of which are still being understood.

1.3 Recent trends and challenges in FFF

Until the end of the last decade, the patent rights for the FFF technology (then called Fused Deposition Modeling/FDM) was held by a single company, the printers of which could print only two materials (Mohamed, Masood, and Bhowmik, 2015; Creusen, 2019). These materials, thermoplastic ABS and PLA, did not provide properties suitable for structural applications. After the FDM patent expiry in 2009 (Mueller, 2017), new players emerged in the market who introduced a new breed of cheaper, faster and more reliable printers. The range of printable materials has also extended to include some high-performance materials like Polyether ether ketone (Rinaldi et al., 2018), Ultem (Bruijn, Gómez-Gras, and Pérez, 2020) and fiber-reinforced polymers (Tekinalp et al., 2014; Brenken et al., 2018; Blok et al., 2018). The advent of high-performance feedstock materials is opening up new prospects in FFF. It may now be possible to extend the FFF applications from prototyping to producing structural components. The unparalleled design freedom of FFF, when combined with such novel high performance materials, might pave the way for new products with tailored properties and a high strength-to-weight ratio.

High Design Freedom
High Performance Materials
Tailored Products for Target Applications

Still, there are challenges to overcome, like the lack of dedicated tools for optimal design and manufacturing of FFF products (Bourell, Leu, and Rosen, 2009; Oropallo and Piegl, 2016). As mentioned before, FFF parts are produced by modeling the parts in a 3D-modeling software and slicing the part into layers of toolpath instructions using a slicing software. However, this strategy is not necessarily the best choice for producing end-use components since the existing software tools do not thoroughly exploit FFF's high design freedom and compromise on the component functionality. For example, despite the technological capability of FFF to freely deposit filaments, the conventional slicing software prints filaments only in a set of standard predefined patterns (like +45°/-45° shown in Figure 1.5). Such a printing practice, regardless of the loading conditions on a printed part, fails to capitalize on the effect of anisotropy and results in a sub-optimal structural behaviour. Instead, what is desirable, yet lacking in a general setting, is an optimized material deposition that fine-tunes the filament orientation in a point-wise manner to yield the best structural performance from FFF parts. The current work, therefore, addresses the optimization and printing of in-plane filaments for improving the structural performance of FFF parts.



```
|1| G1 X102.250000 Y90.300000 F12000
%Non-printing Travel and Velocity

|2| G1 X102.250000 Y90.470000 E4.002638

|3| G1 X102.240000 Y90.630000 E4.005126
%Instruction for printing two lines
```

FIGURE 1.5: Slicing process with the predefined deposition patterns.

1.4 Filament path optimization: state-of-the-art

Optimization approaches have already been used in the literature to design AM parts with optimal topologies and mesostructures (Ngim, Liu, and Soar, 2007; Gaynor, 2015; Zegard and Paulino, 2016; Liu et al., 2018). Surprisingly, only a limited attention has been given to optimize the in-plane material placement in AM parts that is clearly advantageous for extrusion-based technologies (Tam and Mueller, 2017, Zhang, Liu, and To, 2017). Such optimization can help to capitalize the effect of in-plane anisotropy and tailor the mechanical properties in different directions to obtain the required structural behaviour (Lopes et al., 2007; Ghiasi et al., 2010; Giorgio, Ciallella, and Scerrato, 2020).

The concept of varying the in-plane material orientations to reach a superior structural performance has been well explored for laminated fiber-reinforced composites (FRC). Hyer and Lee, 1991 were among the first to demonstrate that the point-wise variation of the fiber paths reaches superior structural performance. Their approach involved discretizing a structure into finite elements (FE) and using the material orientation in each FE as a design variable. The result is a material orientation map that must be post-processed into a set of manufacturable fiber path trajectories. Many works were inspired by this approach, including Duvaut et al., 2000, Stegmann and Lund, 2005, Huang and Haftka, 2005, Legrand et al., 2006, Malakhov and Polilov, 2016, Kiyono, Silva, and Reddy, 2017, etc. In another popular design approach, the fiber paths are represented as curvilinear functions, and their coefficients become the design variables. Such a concept was first introduced by Nagendra et al., 1995 using Non-Uniform Rational B-Splines and was later extended to linearly varying fiber angles (Tatting, Gürdal, and Jegley, 2002; Lopes, Gürdal, and Camanho, 2008), Bezier curves (Kim, Potter, and Weaver, 2012), B-Splines (Honda, Narita, and Sasaki, 2009; Honda and Narita, 2012), Lagrange polynomials (Wu et al., 2012) etc. The main advantage of this method is the reduced number of design variables and the continuity of the fiber paths designed. Besides, multi-level methods have been also been proposed, in which the composite design problem is split into subproblems to deal with the non-convex relations between fiber orientations and the physical responses in the optimization problem (Izzi, Catapano, and Montemurro, 2021). At the macroscopic scale, the anisotropic response is described using lamination (Setoodeh et al., 2009) or polar parameters (Catapano, Desmorat, and Vannucci, 2015; Montemurro and Catapano, 2017), and the determination of optimal stacking sequence is done at the second stage.

The above techniques have been applied to multiple fiber-path optimization problems involving compliance (Stegmann and Lund, 2005; Kiyono, Silva, and Reddy, 2017), buckling loads (Wu et al., 2012), fundamental frequency (Blom et al., 2008), failure criteria (Lopes, Gürdal, and Camanho, 2008), stress concentration (Malakhov and Polilov, 2016) etc., using both gradient-based (Stegmann and Lund, 2005; Blom et al., 2008; Lemaire, Zein, and Bruyneel, 2015) and evolutionary (Legrand et al., 2006; Wu et al., 2012; Huang, Wang, and Li, 2016) optimization algorithms. Nevertheless, most of the works overlooked the manufacturability of the designs (Lozano, Tiwari, and Turner, 2018), thus, making it challenging to transform the optimized designs into a finished product. For detailed reviews on the design of laminated composites, the reader is referred to Ghiasi, Pasini, and Lessard, 2009 and Ghiasi et al., 2010.

To the author's best knowledge, only a few works sought optimized material deposition paths to improve the structural performance of AM parts. Using finite element simulations, Hoglund and Smith, 2016 computed the preferred fiber angles in each finite element of the FFF part's computational model to minimize its compliance. Yamanaka et al., 2016 used the analogy between the fiber paths and streamlines to optimize the fiber courses that maximized the fracture strength of a 3D-printed continuous-fiber composite plate. Liu and Yu, 2017 used the level-set method for optimizing the deposition paths in both fixed and flexible geometries (subject to topology changes), which was later extended by Liu and To, 2017 for three-dimensional structures, including support constraints. Roberge and Norato, 2018 optimized the material orientation and spacing to design curvilinear scaffolds with maximum stiffness for Direct Ink Writing (an extrusion-based AM method). Fernandez et al., 2019 optimized the toolpaths for minimizing the compliance of Direct Ink Written parts while considering the manufacturing constraints. Finally, the fiber direction in a 3D-printed composite was optimized by Hou et al., 2021 based on the stress-gradients to improve its ultimate strength.

Most works mentioned above result in an optimized orientation-field along which filament material must be deployed to create the optimized designs. However, since the slicing software is incapable of achieving this, some post-processing techniques are usually adopted to convert the design solutions into manufacturable filament paths. For example, a common strategy for printing along an orientation map involves dividing the geometry into many auxiliary volumes and filling each one with some standard filament pattern (like linear or concentric infill) that approximates the local orientations (Catapano et al., 2019). Naturally, such procedures do not promise conformity between the designed solutions and the printed parts, leading to a perfectible final structural performance.

1.5 Objectives of the thesis

Existing strategies to design and print optimized filament paths leave space for improvement, mainly due to the prevailing limitations of the slicing software in printing customized filament patterns. This thesis aims to lift some limitations by proposing two new contributions that benefit both the design and printing of optimized filaments in extrusion-based AM systems (regardless of whether pure or fiber-reinforced thermoplastic polymers are used).

First, *an algorithmic framework that yields optimized, readily manufacturable filament paths* is developed. The aim is to minimize the compliance of two-dimensional FFF structures subject to prescribed forces, and operating under linear elastic and small strain regimes. Compliance is a good indicator of the global response from elastostatic structures. It is the work done by external forces that is also understood to be inverse of the global structural stiffness in case the forces acting on the structures are prescribed. Therefore, the optimized filaments that minimize the compliance of a structure also make it stiffest.

Figure 1.6 illustrates the concept of the proposed optimization framework. The framework accepts design specifications such as geometry, loads, material properties, filament dimensions etc., and directly imposes the FFF printing constraints into the optimization formulation to design filament paths optimized for minimum compliance. The resulting solutions have the advantage of being easily translatable into G-Code instructions without undergoing any post-processing techniques that may decrease their expected structural performance.

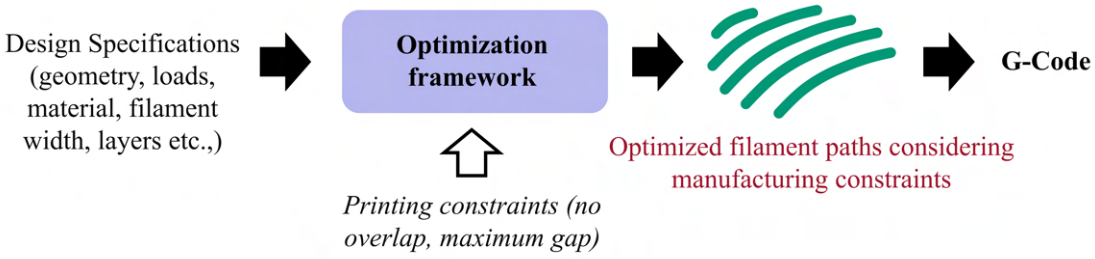


FIGURE 1.6: Contribution 1 - Proposed optimization of filament deposition paths including the manufacturing constraints. The solution can be directly converted into G-Code.

Second, *a new filament deposition algorithm that prints filaments along any given in-plane orientation-field* is developed. The algorithm ensures the manufacturability of the solutions from the commonly used point-wise optimization strategies that otherwise need approximations to a filament pattern in slicing software. Figure 1.7 depicts the working principle of the new filament deposition algorithm. The input to the algorithm is a vectorized orientation field available at a set of discrete points (for e.g., finite element nodes) that is then converted into a set of filament trajectories using streamline visualization techniques. The highlight of the proposed algorithm is that it takes into account the majority of the process constraints and design requirements crucial for producing structural FFF parts, also providing as output a ready-to-use G-Code.

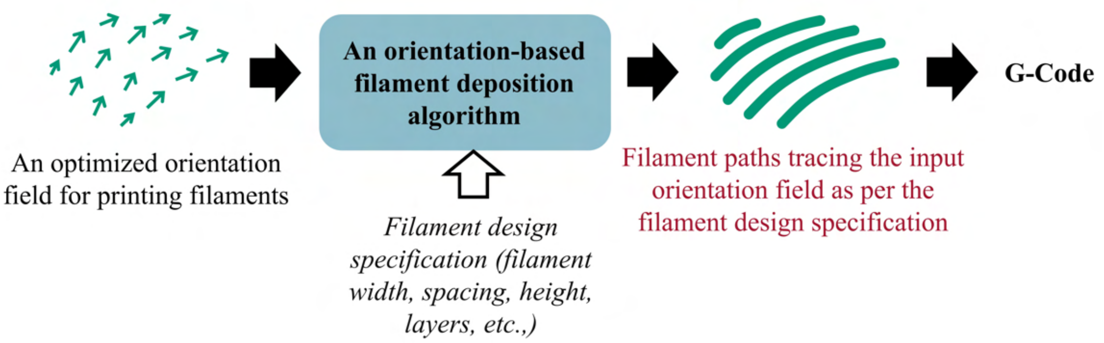

An orientation-based filament deposition algorithm
G-Code
An optimized orientation field for printing filaments
Filament paths tracing the input orientation field as per the filament design specification
Filament design specification (filament width, spacing, height, layers, etc.,)

FIGURE 1.7: Contribution 2 - Proposed filament deposition algorithm that generates filament paths from orientation fields.

Organization of thesis

The thesis is organized as given below:
Chapter 2 outlines the structural optimization concept used to optimize the in-plane filament paths, detailing the different choices for its essential constituents: the minimization algorithm, the prediction of the structural response, the design parametrization and the constraint enforcement.
Chapter 3 develops the algorithmic framework for obtaining optimized, production ready in-plane filament paths for FFF parts. The developed framework is applied to design the filament paths for some two-dimensional FFF structures using different minimization algorithms and a new phenomenological anisotropic material model that includes the effect of inter-filament spacing.
Chapter 4 presents a new filament deposition algorithm for printing customized filament paths based on the orientation maps from optimization

procedures. Capabilities of the algorithm are presented and analyzed by producing parts with both partial and maximum infill, and parts with a multi-oriented layer stacking sequence.

Chapter 5 summarizes the contributions of the thesis and provides insights on further areas to research.

Chapter 2

Elements of In-Plane Filament Path Optimization

This chapter identifies the essential elements required for optimizing the filament paths, referring to the general technique of *Structural Optimization (SO)*.

SO is a design approach in which the best parameters of the structure that meet the desired objectives are found through an iterative process (Christensen and Klarbring, 2008; Parkinson, Balling, and Hedengren, 2013; Rao, 2019). Figure 2.1 depicts the typical working of SO. An optimization algorithm evaluates the structural response, for instance, using the finite element method and changes its design variables iteratively until the design variables that satisfy the desired structural objective are found. Different types of SO are shown in Figure 2.2.

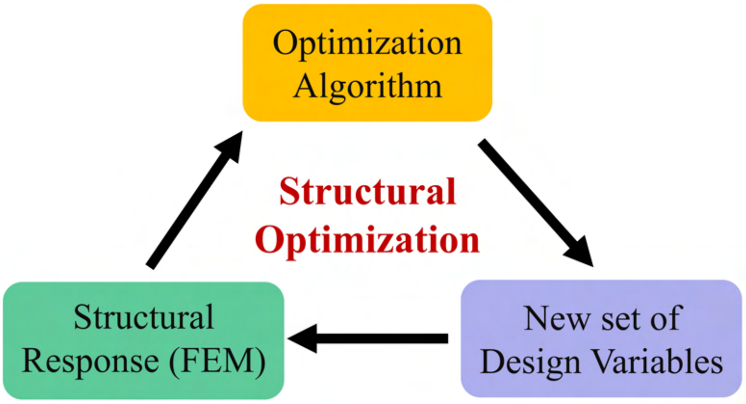
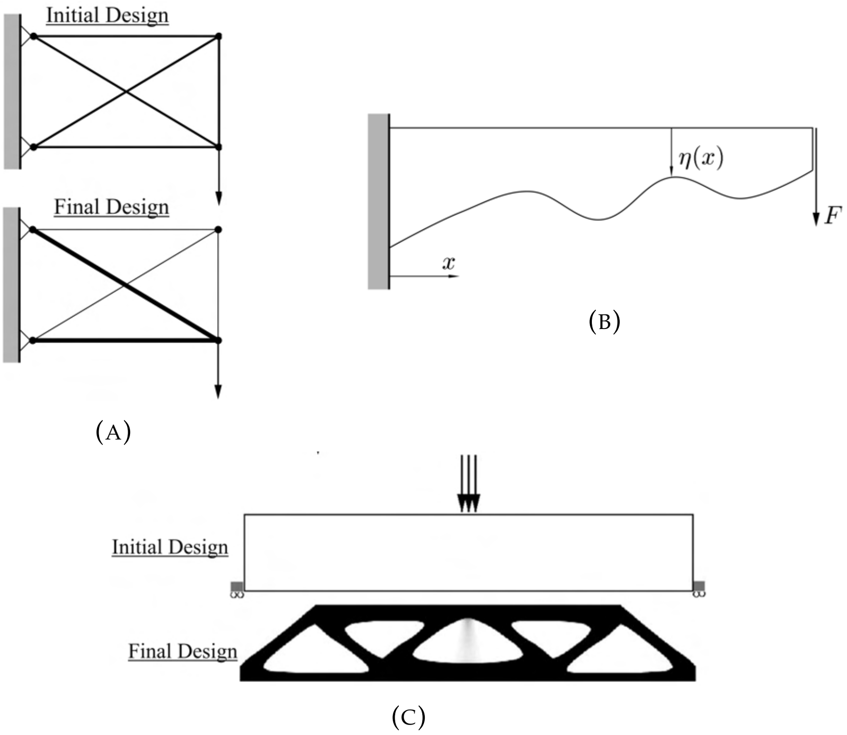


FIGURE 2.1: Workflow in structural optimization.

Thus, three essential elements constitute an SO problem: the prediction of the structural response, the design parametrization, and the optimization algorithm. Besides, the imposition and the treatment of the manufacturing constraints pose separate challenges to the optimization process. The present chapter discusses various choices for each of these elements in the context of filament path optimization.



(A)

(B)

(C)

FIGURE 2.2: Different types of structural optimization. (A) In Sizing Optimization, some type of structural thickness is optimized, i.e., area of cross-section in trusses or the thickness distribution of a sheet. (B) Shape Optimization optimizes the shape of a structural domain (shape). (C) Topology Optimization optimizes the material connectivity or its layout within a given design space (Christensen and Klarbring, 2008).

First, Section 2.1 describes the mechanical characteristics of FFF parts and explains the state-of-the-art methods to predict their behaviour. Section 2.2 describes the direct and indirect strategies for parametrizing in-plane material paths and explains the merits in each method. Section 2.3 defines, in a general setting, the filament path optimization problem to be solved and explains two different approaches to solve the problem. Section 2.4 discusses the penalty method for constraint enforcement along with its characteristics. Finally, Section 2.5 concludes this chapter with a brief note on the software employed to optimize the filament paths.

2.1 Structural response of FFF parts

Predicting an FFF part behaviour is non-trivial since the additively built parts are not homogeneous and possess properties different from the raw material used to print the part. While the raw material material is isotropic, the final FFF part exhibits anisotropy due to the filament or the building orientation

(Figure 2.3), and contains inter/intra-layer voids that deteriorate its mechanical properties. Thus, the knowledge of the raw material properties does not suffice to predict the part's performance like in subtractive or formative technologies. Instead, the mechanical properties of the printed part, and hence its behaviour, are functions of an array of building and process parameters (Figure. 2.4) used to print the part.

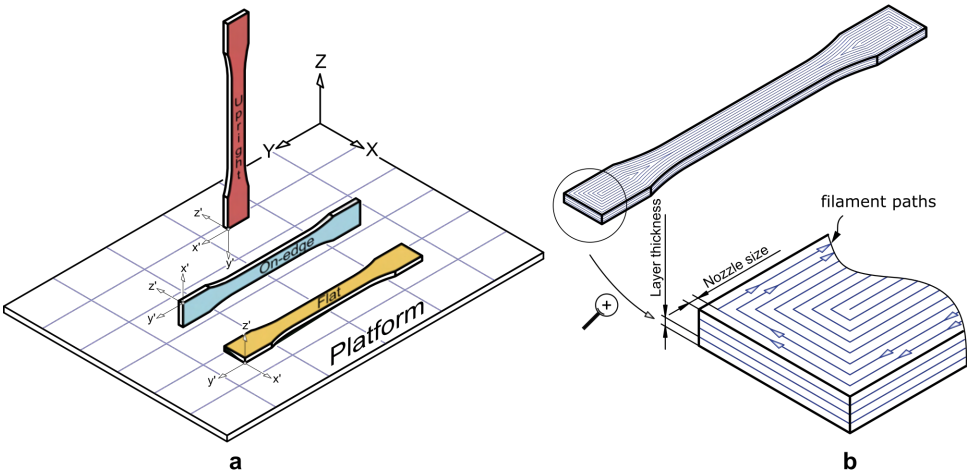


FIGURE 2.3: Building orientations and filament orientation in FFF parts (Chacón et al., 2017)

Consequently, the literature shows a large volume of experimental works that characterize the tensile (Onwubolu and Rayegani, 2014; Deng et al., 2018), bending (Chacón et al., 2017; Gebisa and Lemu, 2018), fracture (Aliheidari et al., 2017; Hart and Wetzel, 2017), fatigue (Ziemian, Ziemian, and Haile, 2016; Puigoriol-Forcada et al., 2018) and impact behaviours (Es-Said et al., 2000; Roberson et al., 2015) of the FFF parts in terms of a multitude of influencing parameters. Still, there are only a few works that built predictive models for the behaviour of FFF parts, some of which are outlined below.

Straight filament paths

The Classical Laminate Theory (CLT) can be used to predict the properties of the FFF parts under restricted scenarios. CLT allows computing the elastic behaviour of multi-layered orthotropic material using the elastic constants of a single layer (Casavola et al., 2016). First, unidirectional samples are experimentally tested to characterize individual filaments. Then, the results are used to build the predictive model for multi-layered FFF parts using CLT. Figure 2.6 illustrates this concept. Kulkarni and Dutta, 1999 and Magalhães, Volpato, and Luersen, 2014 accounted for the influence of voids in CLT models and found that the correlation between the experimental and predicted results got better as the fraction of voids decrease. Thus, CLT can be applied only if the FFF parts are quasi-solid (infill density ≈ 100%) and printed in flat orientation.

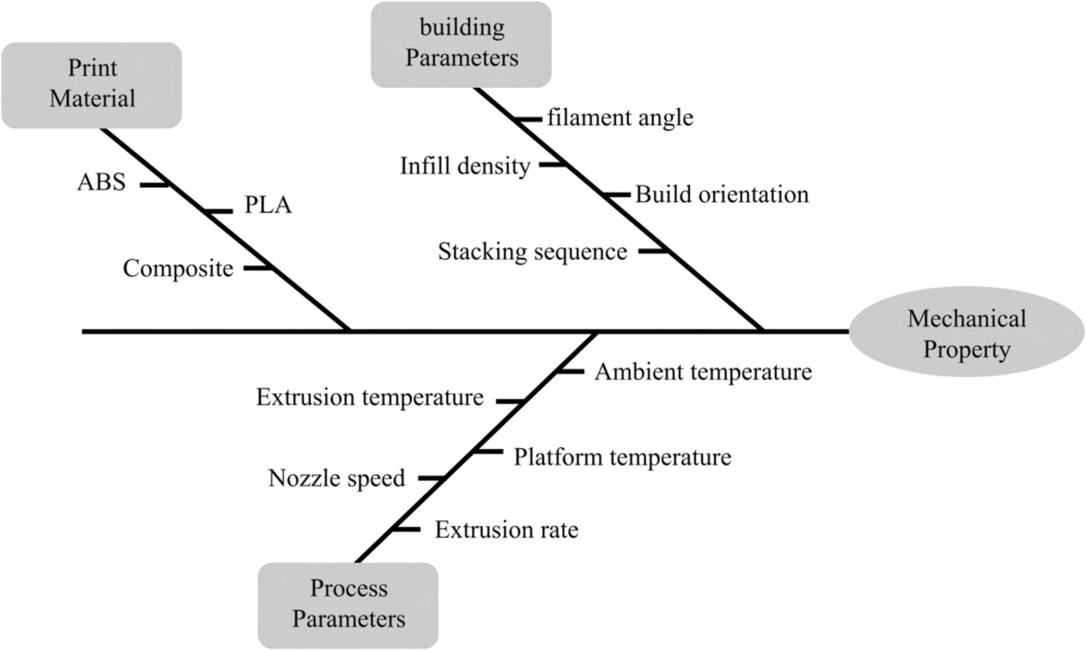


FIGURE 2.4: FFF parameters influencing the material properties

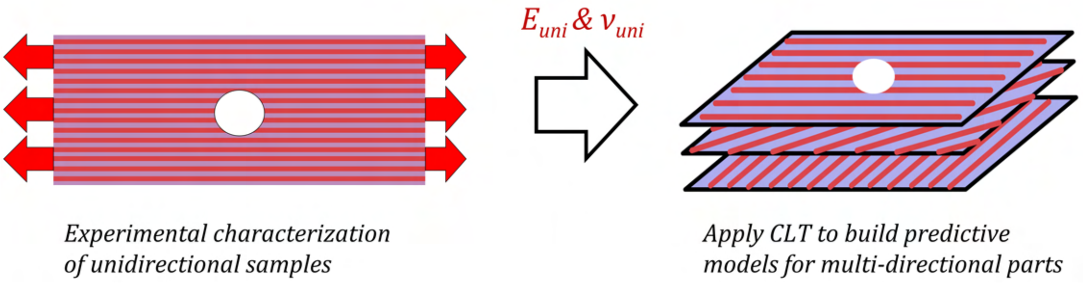

E_{uni} & ν_{uni}
Experimental characterization of unidirectional samples
Apply CLT to build predictive models for multi-directional parts

FIGURE 2.5: Application of CLT to predict FFF part properties. E_{uni} and ν_{uni} are, respectively, Young's modulus and Poisson's ratio for unidirectional parts.

For predicting the partially-filled lattice-type parts, micromechanics based approaches are more appropriate. Such methods focus on the mechanics of the unit-cells instead of layers, wherein each cell represents a building block that is repeated to complete the lattice-structure (Figure 2.6). For example, Cuan-Urquizo, Yang, and Bhaskar, 2015 predicted the elastic response of PLA-based lattice structures using the rule of mixtures given below,

$$E_L = (aN_f/A)E_f \tag{2.1}$$

where a denotes the area of cross-section of a single filament, E_f is the corresponding experimental Young's modulus, A is the area of cross-section of lattice structure with N_f filaments, and E_L is the Young's modulus of lattice. In a similar work, Cuan-Urquizo and Bhaskar, 2018 derived the Young's modulus and effective shear modulus as functions of infill density for the bending of a 0°/90° lattice structure, assuming that the filaments behave like individual beams.

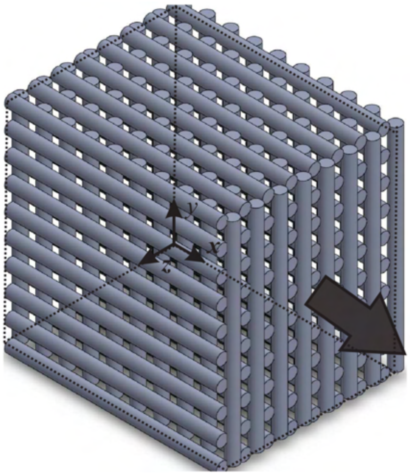


FIGURE 2.6: Lattice-type FFF part (Cuan-Urquizo and Bhaskar, 2018)

Somireddy and Czekanski, 2017 assumed isotropic material properties for the individual filaments and numerically simulated tensile tests on unidirectional FFF parts. The obtained results were applied with CLT to predict multi-directional FFF parts that showed a very good match to the experimental results. Similarly, Cuan-Urquizo and Bhaskar, 2018 modelled the lattice-type FFF structure as PLA-based cylinders sharing patches of volume. Under bending loads, the numerical results were in good agreement with the analytical predictions from the beam theory. Somireddy, Czekanski, and Singh, 2018 used the technique of numerical homogenization to characterize an L-shaped part. The strain energy of heterogeneous RVEs were computed using FEA and equated to homogenized RVEs for computing the constitutive matrix of the printed part.

Curvilinear filament paths

All the above methods assume either quasi-solidity or lattice-type arrangement with straight filaments, which narrows design possibilities. Ideally, load-bearing FFF parts should be tailored by varying their in-plane filament orientations and density for specific applications. Nevertheless, this comes at a cost that the prediction of their mechanical behaviour becomes much more challenging. To the author's best knowledge, very few strategies have been proposed to characterize freely deposited FFF parts.

Roberge and Norato, 2018 predicted the behaviour of curvilinear bone scaffolds that have a lattice-type construction. Using the micromechanics of beams, functional relationships were established between the elastic constants of the scaffolds and the geometric parameters of unit cells. To compute the calibration constants of this approach, least square regression was used to fit analytical models to the data resulting from computations. Numerical homogenization was applied on RVEs for varying air gaps and overlap values to obtain their effective elastic properties. Once the elasticity tensor was assembled, the response of the scaffold with curvilinear filaments was predicted by rotating the elasticity tensor by the local filament orientation in each finite element.

Most recently, Fernandez et al., 2019 proposed a phenomenological material model for FFF parts with varying in-plane orientations and density. For an FFF part with n layers, the elasticity tensor $\mathbb{C}^*(\mathbf{x})$ at any point $\mathbf{x}$ in the i^{th} layer was modelled as a function of the local filament orientation, $\alpha_i(\mathbf{x})$, and volume-ratio (density), $\nu_i(\mathbf{x})$:

$$\mathbb{C}^*(\mathbf{x}) = \nu_i(\mathbf{x})\mathbb{C}(\alpha_i(\mathbf{x})) \tag{2.2}$$

where $\mathbb{C}(\alpha_i(\mathbf{x}))$ is the elasticity tensor of a full-volume transversely isotropic material that is rotated by the local filament orientation.

Similar to Fernandez et al., 2019, the present work also develops a phenomenological material model accounting for filament orientation and density. The newly developed material model incorporates a realistic effect of inter-filament gaps on the stiffness and produces more realistic designs from filament path optimization, which is proven by comparison with the model of Fernandez et al., 2019 in Chapter 3. However, it is emphasized that the proposed model does not predict the failure of FFF parts (due to weak inter-filament bonding), or the local deformation of the unit-cells when they are partially filled. The model only gives a qualitative response from the FFF parts, which is sufficient for optimizing filament paths that minimize compliance (explained in section 2.3).

2.2 In-plane filament path parametrization

In any structural optimization problem, the procedure to parametrize the structural elements, such as size, shape or material paths, into the variables that intervene in the optimization process is called design parametrization. Here, the two commonly used parametrization strategies of fiber-reinforced composites (FRC) are discussed, considering the similarities between the FFF parts and FRCs (Figure 2.7).

2.2.1 Direct parametrization

The direct parametrization strategy involves discretizing the studied structure into finite elements and directly considering the material orientation at each FE as a design variable (Hyer and Charette, 1991; Stegmann and Lund, 2005; Bruyneel, 2011; Malakhov and Polilov, 2016; Acar et al., 2016; Kiyono, Silva, and Reddy, 2017). The result is an optimal material orientation map over the FE discretized structure that must be post-processed to generate the set of toolpath trajectories (Figure 2.8).

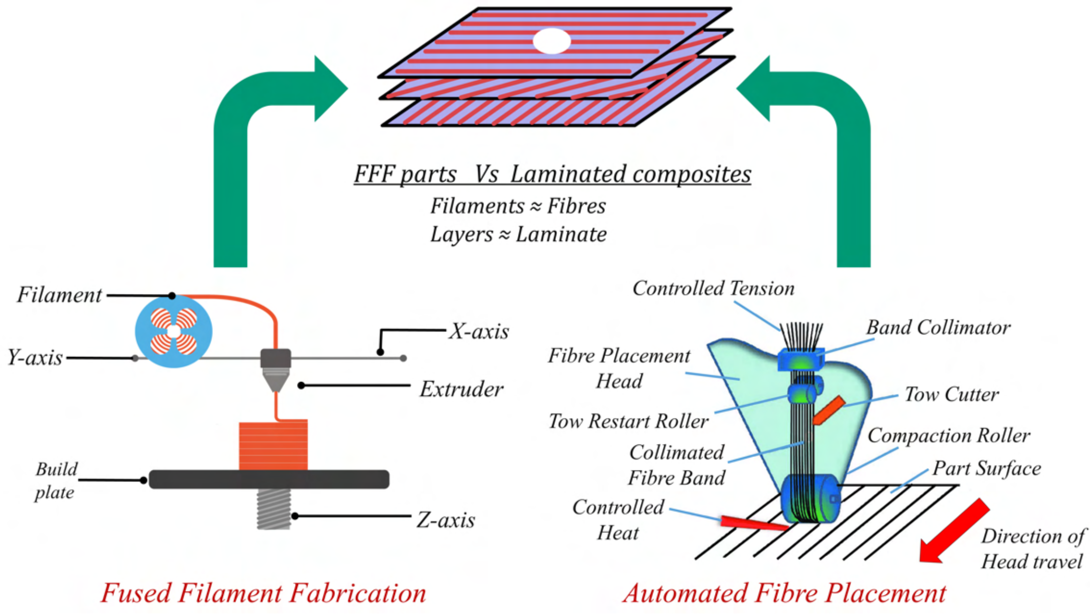


FIGURE 2.7: Similarity of FFF parts and laminated fiber-reinforced composite (FRC). Laminated FRC are produced by stacking layers of plies, which have fibers along a direction to provide anisotropic properties to the composite. Such composites are produced using the Automated Fiber Placement technique, which uses a robotic arm to layup groups of preimpregnated fibers (tows) onto a tool (Druckwege, 2022; Ngo, 2020; Rousseau et al., 2019).

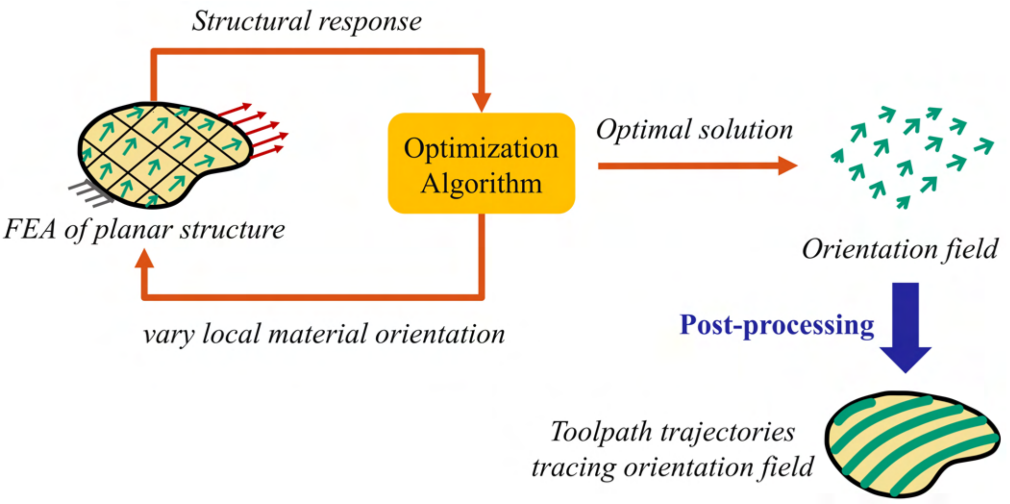


FIGURE 2.8: Optimization of in-plane material orientation in each Finite Element and post-processing to derive toolpath trajectories.

Being relatively simple to implement, this method offers maximum design freedom and enables the point-wise optimization of the filament (or fiber) orientation. However, this approach suffers from a high computational cost due to the inter-dependency between the finite element discretization and the number of design variables. To circumvent this issue, a few works have adopted patch designs where a structure is divided into subregions or patches each having a constant material orientation, as shown in Figure 2.9 (Hyer and Lee, 1991; Catapano et al., 2019). Still, a major disadvantage in the direct parametrization technique is that it does not allow the imposition of manufacturing constraints required to transform the results directly into printable solutions (Lozano et al., 2016). Instead, the obtained optimal solutions must be approximated to the closest manufacturable solution in an additional post-processing step, which, however, compromises the optimality of the design solutions.

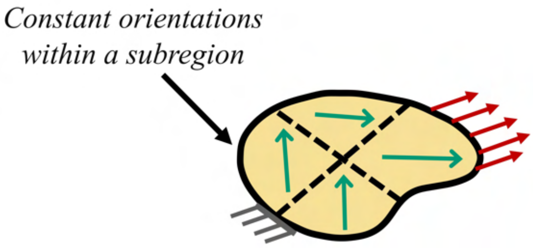


FIGURE 2.9: Reduction of design variables through the definition of subregions or patches. Such a technique introduces observable interfaces between the patches.

2.2.2 Indirect parametrization through curvilinear functions

Here, the fiber path trajectories are represented as parametrized curves and the corresponding shape parameters become the design variables. The structural response is computed by extracting the local orientations of the curves that are mapped onto finite elements. Usually, a single reference curve is optimized and then replicated to produce the fiber paths for the entire structure. Reference curves having linearly varying fiber angles have been widely used (Tatting, 1998; Tatting, Gürdal, and Jegley, 2002; Lopes, Gürdal, and Camanho, 2008; Lopes, 2009; Haldar et al., 2018), although the non-linear variations of fiber angles have also been considered to exploit a larger design space. Some examples include Lagrangian polynomials (Wu et al., 2012), Bezier curves (Kim, Potter, and Weaver, 2012), splines (Honda and Narita, 2012) and NURBS (Nagendra et al., 1995). The main advantage of using parametrized curves to represent the fiber paths is the limited number of design variables and the easy imposition of manufacturing constraints to maintain continuity of the fiber paths.

TABLE 2.1: Comparison of direct and indirect parametrizations of material paths.

Direct Parametrization	**Indirect Parametrization**
Large design space	Limited design space
Manufacturability not guaranteed	Production-ready design solutions
Non-trivial post processing	Minimal or no post processing

Table 2.1 compares the characteristics of direct and indirect parametrization. The direct parametrization technique offers a large design space, but it does not promise manufacturability of the design solutions. In contrast, the indirect parametrization technique works with a limited design space and offers readily manufacturable solutions.

Both direct and the indirect parametrization techniques have been used in this work. Aiming to obtain readily printable design solutions, Chapter 3 uses an indirect technique for optimizing the filament paths. At the same time, the manufacturability issues of the direct parametrization technique are also addressed. Chapter 4 proposes a new filament deposition algorithm that accepts the orientation fields from the direct parametrization methods and provides the G-code instructions for printing along the orientation fields without introducing any approximations.

2.3 Filament path optimization problem and solution approaches

Any optimization problem is essentially a minimization or maximization problem, in which the goal is to find the set of design variables that minimize or maximize an objective function subject to some constraints. A classical constrained minimization problem is given below,

$$\begin{aligned} &\underset{\mathbf{z}}{\text{minimize}} && f(\mathbf{z}) \\ &\text{subject to} && h_j(\mathbf{z}) \leq 0, \quad j = 1, 2...m \end{aligned} \tag{2.3}$$

where $f(\mathbf{z})$ is the objective/response function to be minimized, $\mathbf{z}$ is the vector of design variables that is changed during the optimization to minimize the objective function, and $h_j(\mathbf{z}) \leq 0$, with $j = 1, 2...m$ are the inequality constraints that must be satisfied by the optimal solution.

In a similar fashion, the filament path optimization problem is stated below. The goal is to find the optimal set of filament paths that minimize the compliance of an FFF structure subject to manufacturing constraints. Accordingly, Eq. (2.3) takes the form

$$\begin{aligned} &\underset{filament\ paths}{\text{minimize}} && \text{structural compliance} \\ &\text{subject to} && \text{manufacturing constraints,} \end{aligned} \tag{2.4}$$

where the structural compliance is the objective function to be minimized, the filament path parameters are the design variables, and the manufacturing constraints are due to the technological limitations of the FFF process.

During the minimization process, the optimization algorithm changes the filament path parameters of the structure under study (with domain Ω) and evaluates the corresponding change in the compliance. The compliance, given by the expression

$$c = \int_{\Omega} \sigma : \epsilon \, d\Omega, \tag{2.5}$$

is computed after solving the equilibrium equation for the structure through finite element analysis, with the updated Cauchy stress (σ) and infinitesimal strains (ϵ) corresponding to the new filament path parameters. Based on the new compliance value, the filament path parameters are again changed, and this process is repeated until the filament paths that minimize the compliance of the structure is found.

Assuming that the compliance is a non-convex function of the filament paths, two different approaches are used to solve the filament path optimization problem: (i) a two-step gradient-based minimization, and (ii) differential evolution. The following sections explain the characteristics of both the methods that will be applied and compared for optimizing the filament paths in Chapter 3.

For simplicity, the following sections use a generic response function f and design variables $\mathbf{z}$ in place of compliance and the filament path parameters.

2.3.1 Two-step gradient-based minimization

In general, gradient-based methods start the minimization process from a single point and iteratively proceed to better approximations along directions computed from the gradient of the objective function at each point. Although such methods ensure fast convergence to minimum solutions, they are sensitive to the starting point, and may converge to a local minimum if the function to be minimized is non-convex (such as the compliance with respect to the filament paths). In order to drive the gradient-based methods towards the global optimum, a two-step approach is set up (similar to Fernandez et al., 2019). Within the two-step approach, first a design with straight and parallel filament paths that minimize the compliance of the structure is obtained. Then, the obtained design is supplied as the starting point for solving the curvilinear filament path optimization problem in Eq. (2.4). Figure 2.10 illustrates the concept of the two-step gradient-based optimization employed in this work.

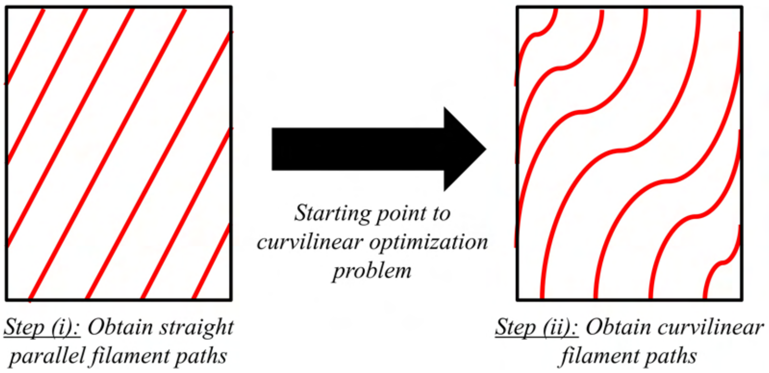

Starting point to curvilinear optimization problem
Step (i): Obtain straight parallel filament paths
Step (ii): Obtain curvilinear filament paths

FIGURE 2.10: Concept of two-step gradient-based minimization. In the first step, the straight and parallel filament paths that minimize the compliance are obtained. The obtained design is used as the starting point for the solving the curvilinear filament path optimization problem.

In the present work, the two-step gradient-based minimization is implemented using the Quasi-Newton algorithm, the working of which is detailed below.

Working of Quasi-Newton

The Quasi-Newton (QN) method was first introduced in Davidon, 1959. Like the Steepest Descent (Curry, 1944) and the Newton method (Fletcher, 2013), QN is an iterative solver that follows a *line search* strategy (Nocedal and Wright, 2006) to solve minimization problems. The minimization algorithm starts from a user-specified point and consistently lowers the objective function values along informatively chosen descent directions. Here, the starting point corresponds to the straight parallel toolpaths that minimize the compliance. If $\mathbf{z}^k$ denotes the solution at the kth iteration, then the next point $\mathbf{z}^{k+1}$ at iteration $(k+1)$ is found as,

$$\mathbf{z}^{k+1} = \mathbf{z}^k + \mathbf{p}^k \tag{2.6}$$

where $\mathbf{p}^k$ is the descent direction chosen at iteration k.

Various choices exist for the descent direction $\mathbf{p}^k$. Steepest Descent sets $\mathbf{p}^k$ to be the negative gradient of the objective function ($\mathbf{p}^k$ = -$\nabla\mathbf{f}^k$), whereas the Newton method uses the Hessian information at a point to compute $\mathbf{p}^k$, as given below.

$$\mathbf{p}^k = -\nabla^2(\mathbf{f}^k)^{-1}\nabla\mathbf{f}^k \tag{2.7}$$

Steepest Descent is relatively inefficient compared to the Newton method since it uses only the first-order information at a point (i.e., the gradient). On the other hand, although the Newton method promises a superior convergence rate, the Hessian matrix is too costly to compute, especially if the function to be minimized is multi-dimensional and not available in closed-form (as is the case of the current application). QN promises a balanced approach by avoiding the costly Hessian matrix computation while simultaneously being more efficient than the Steepest Descent.

QN approximates the Hessian $\nabla^2\mathbf{f}^k$ in Eq. (2.7) using a less expensive matrix $\mathbf{B}^k$ that is computed from the objective function value and its gradient at the previous iteration. For a downhill movement, QN requires $\mathbf{B}^k$ to be positive definite and to satisfy the following condition,

$$\mathbf{B}^{k+1}\mathbf{a}^k = \mathbf{q}^k \tag{2.8}$$

where $\mathbf{a}^k = \mathbf{z}^{k+1} - \mathbf{z}^k$ and $\mathbf{q}^k = \nabla \mathbf{f}^{k+1} - \nabla \mathbf{f}^k$. Among the different methods to compute the matrix $\mathbf{B}^k$, the BFGS (Broyden, Fletcher, Goldfarb, and Shanno) formula is employed here:

$$\mathbf{B}^{k+1} = \mathbf{B}^k - \frac{\mathbf{B}^k\mathbf{a}^k(\mathbf{a}^k)^T\mathbf{B}^k}{(\mathbf{a}^k)^T\mathbf{B}^k\mathbf{a}^k} + \frac{\mathbf{q}^k(\mathbf{q}^k)^T}{(\mathbf{q}^k)^T\mathbf{a}^k} \tag{2.9}$$

The pseudo-code for the QN algorithm with the BFGS method is given below.

```
Define starting point $\mathbf{z}^0$ and convergence tolerance $tol$>0,
Compute $\mathbf{B}^0$
$k = 0$
While $||\nabla \mathbf{f}^k||>tol$ (OR) $k \leq Iter_{max}$
    Compute $\mathbf{p}^k = -(\mathbf{B}^k)^{-1}\ \nabla \mathbf{f}^k$
    Compute $\mathbf{z}^{k+1} = \mathbf{z}^k + \mathbf{p}^k$
    Compute $\mathbf{a}^k = \mathbf{z}^{k+1} - \mathbf{z}^k$ and $\mathbf{q}^k = \nabla \mathbf{f}^{k+1} - \nabla \mathbf{f}^k$
    Update $\mathbf{B}^{k+1}$ using Eq. (2.9)
    $k = k + 1$
end While
```

Sensitivity Analysis

Gradient-based methods require the objective function gradient at every iteration, which may not be readily available. The procedure to compute the derivatives of the objective function with respect to the design variables is referred to as sensitivity analysis (Choi and Kim, 2004; Van Keulen, Haftka, and Kim, 2005).[1] Different methods to compute the sensitivities are described below.

The *forward difference* and the *central difference* methods offer the simplest means for finding sensitivities. If $f(\mathbf{z})$ is the response function of a structure under equilibrium, then the sensitivities $df/d\mathbf{z}$ are calculated as given below. Forward difference method:

$$\frac{df}{dz_j} \approx \frac{f(\mathbf{z} + \Delta\mathbf{z}_j) - f(\mathbf{z})}{\Delta\mathbf{z}_j} \tag{2.10}$$

[1] In the field of numerical optimization, sensitivity analysis refers to the process of determining the changes in the optimum to small perturbations in the parameter or constraint values (Nocedal and Wright, 2006).

Central difference method:

$$\frac{df}{dz_j} \approx \frac{f(\mathbf{z}+\Delta\mathbf{z}_j) - f(\mathbf{z}-\Delta\mathbf{z}_j)}{2\Delta\mathbf{z}_j} \tag{2.11}$$

where $j = 1, 2...N_{dv}$ denote the index of the design variable, and $\Delta\mathbf{z}_j$ denotes a small perturbation introduced to the j^{th} design variable.

The above techniques are simple to implement and are suitable for situations where the objective function is a black box (i.e., no information is available on the nature of objective function). However, despite their simplicity, such methods incur a high computational cost. For N_{dv} design variables, the forward difference method requires $N_{dv}+1$ function evaluations and the central difference method needs $2N_{dv}+1$ function evaluations. Therefore, in situations where the response function is evaluated through numerical simulations, analytical sensitivities are more preferable.

Analytical sensitivities are more accurate and incur a lower computational cost (Tortorelli, Haber, and Lu, 1991; Lee, 1996; Pandey and Bakshi, 1999; Stillman, 2000). Let us consider an elastostatic system in which the response function $f(\mathbf{z})$ depends upon the design variable $\mathbf{z}$ and the displacements $\mathbf{u}(\mathbf{z})$, i.e., $f(\mathbf{z})$=$g(\mathbf{z}, \mathbf{u}(\mathbf{z}))$. Then, the sensitivity of the response function f with respect to a design variable z_j is,

$$\frac{df}{dz_j} = \frac{\partial g(\mathbf{z}, \mathbf{u}(\mathbf{z}))}{\partial \mathbf{u}} \frac{d\mathbf{u}}{dz_j} + \frac{\partial g(\mathbf{z}, \mathbf{u}(\mathbf{z}))}{\partial z_j} \tag{2.12}$$

The above equation cannot be evaluated directly due to the term $d\mathbf{u}/dz_j$ that is implicitly defined through the equilibrium equation $\mathbf{K}(\mathbf{z})\,\mathbf{u}(\mathbf{z}) = \mathbf{P}(\mathbf{z})$, where $\mathbf{K}$ is the stiffness matrix and $\mathbf{P}$ is the loading vector. Therefore, first $d\mathbf{u}/dz_j$ is computed by differentiating the equilibrium condition and solving the resulting algebraic equation. Then, the newly computed $d\mathbf{u}/dz_j$ is substituted into the above equation to find the sensitivity of the response function. This method is called *direct analytical method*.

Another approach, called *adjoint method*, eliminates the $d\mathbf{u}/dz_j$ term using a Lagrange multiplier method. First, an augmented function is introduced:

$$\hat{f}(\mathbf{z}) = g(\mathbf{z}, \mathbf{u}(\mathbf{z})) - \boldsymbol{\lambda}(\mathbf{K}(\mathbf{z})\mathbf{u}(\mathbf{z}) - \mathbf{P}(\mathbf{z})) \tag{2.13}$$

where $\boldsymbol{\lambda}$ is an arbitrary vector of Lagrange multipliers. Note that the function $\hat{f}$ is equal to f since $\mathbf{K}(\mathbf{z})\mathbf{u}(\mathbf{z}) - \mathbf{P}(\mathbf{z})$ equals zero. Differentiation of Eq. (2.13) and subsequent rearrangement gives

$$\begin{aligned}\frac{d\hat{f}}{dz_j} = &\left[\frac{\partial g(\mathbf{z},\mathbf{u}(\mathbf{z}))}{\partial z_j} - \boldsymbol{\lambda}.\left(\frac{d\mathbf{K}(\mathbf{z})}{dz_j}\mathbf{u}(\mathbf{z}) - \frac{d\mathbf{P}(\mathbf{z})}{dz_j}\right)\right] \\ &+ \frac{d\mathbf{u}(\mathbf{z})}{dz_j}.\left[\frac{\partial g(\mathbf{z},\mathbf{u}(\mathbf{z}))}{\partial \mathbf{u}} - \mathbf{K}^T(\mathbf{z})\boldsymbol{\lambda}\right].\end{aligned} \tag{2.14}$$

For a suitable value of Lagrange multipliers $\hat{\boldsymbol{\lambda}}$, the unknown $d\mathbf{u}/dz_j$ term can be eliminated, and the sensitivity $d\hat{f}/dz_j$ (i.e., df/dz_j) is found without involving the last term in Eq. (2.14). Accordingly, the following equation is solved to compute $\hat{\boldsymbol{\lambda}}$:

$$\mathbf{K}^T(\mathbf{z})\hat{\boldsymbol{\lambda}}(\mathbf{z}) = \frac{\partial g(\mathbf{u}(\mathbf{z}),\mathbf{z})}{\partial \mathbf{u}}. \tag{2.15}$$

The resulting $\hat{\boldsymbol{\lambda}}(\mathbf{z})$ is substituted into Eq. (2.14), which is reduced to:

$$\frac{d\hat{f}}{dz_j} = \left[\frac{\partial g(\mathbf{z},\mathbf{u}(\mathbf{z}))}{\partial z_j} - \hat{\boldsymbol{\lambda}}(\mathbf{z}).\left(\frac{d\mathbf{K}(\mathbf{z})}{dz_j}\mathbf{u}(\mathbf{z}) - \frac{d\mathbf{P}(\mathbf{z})}{dz_j}\right)\right]. \tag{2.16}$$

The adjoint method is more efficient than the direct analytical method. In the direct analytical method, the $d\mathbf{u}/dz_j$ needs to be found separately for each design variable z_j. Whereas the adjoint method uses the same set of Lagrange multipliers $\hat{\boldsymbol{\lambda}}$ for computing all the sensitivities using Eq. (2.16), which is more efficient.

Besides the direct and adjoint methods, automatic differentiation techniques can also be employed to compute the sensitivities. Automatic differentiation tools exploit the analytical relations encoded between the various quantities to directly yield the required derivatives. The present work uses a combination of the automatic differentiation techniques and the adjoint method to compute sensitivities. The expression for the sensitivity of compliance with respect to the design variables is provided in Chapter 3.

2.3.2 Differential evolution

The filament path optimization problem is also solved using a population-based evolutionary algorithm called differential evolution. This algorithm was introduced as a technical report in 1995 (Storn and Price, 1995) and was later published in the Journal of Global Optimization in 1997 (Storn and Price, 1997). Since then, differential evolution has become one of the most popular global optimisers (Trunfio, 2016). Differential evolution operates by a principle that is agnostic to the gradient of the objective function. It applies evolutionary principles to a set of randomly initialized population of candidate solutions and probes the function landscape for the global minimum solution. Differential evolution does not guarantee global minima, especially if non-convex functions are to be minimized. However, it is still preferred here because it avoids getting trapped to the local minima and is also easier to implement.

Working of differential evolution

The process of differential evolution starts with *initializing* a population of candidate solutions. A single candidate solution is represented as: $\mathbf{z}_t^g = \left(z_{t,j}^g\right)$, where $t = 1, 2...SP$ is the index of the candidate solution in a population of size SP, the index $j = 1, 2...N_{dv}$ denotes design variable, and g is the generation number. During initialization (i.e., at $g = 0$), random values are assigned to the entire population of candidate solutions within some preset bounds. Then, the initialized candidate solutions undergoes a repetitive process of mutation, cross-over and selection, as explained below.

Mutation is a perturbation step that generates new candidate solutions from the existing ones to explore the function space for better solutions. In mutation, the scaled difference of any two random vectors is added to a third random vector to produce a mutation vector $\mathbf{m}_t^g$ as shown below:

$$\mathbf{m}_t^g = \mathbf{z}_{r1}^g + F(\mathbf{z}_{r2}^g - \mathbf{z}_{r3}^g), \tag{2.17}$$

where $F \in$ (0,1+) is the scale factor (explained later), and $r1$, $r2$ and $r3$ denote the random indices which must be chosen different from the current index t.

The *cross-over* operation, also referred to as recombination, is primarily done to promote the diversity (distribution in the function space) of the candidate solutions. In this operation, the components of the mutation ($\mathbf{m}_t^g$) and parent vectors ($\mathbf{z}_t^g$) are copied to generate the trial vectors ($\mathbf{r}_t^g$). The fraction of parameters copied from the mutation or the parent vector is controlled by a user-set parameter called the cross-over probability $C_r \in$ [0,1]. The cross-over operation is explained below using the pseudo-code:

$$r_{t,j}^g = \begin{cases} m_{t,j}^g & \text{if rand}(0,1) \leq C_r \\ z_{t,j}^g & \text{otherwise} \end{cases} \tag{2.18}$$

If a random number generated by the rand(0,1) is less than C_r, then the trial parameter comes from the mutant vector $\mathbf{m}_t^g$, else it comes from the parent vector $\mathbf{z}_t^g$.

In the *selection* step, the parent vectors $\mathbf{z}_t^g$ and the trial vectors $\mathbf{r}_t^g$ of the current generation are compared. If the objective function value $f(\mathbf{r}_t^g)$ is less than or equal to the objective function value $f(\mathbf{z}_t^g)$, then $\mathbf{r}_t^g$ replaces the $\mathbf{z}_t^g$ to become the parent vector in the next generation $g+1$. Otherwise, the $\mathbf{z}_t^g$ retains its place for the next generation.

$$\mathbf{z}_t^{g+1} = \begin{cases} \mathbf{r}_t^g & \text{if } f(\mathbf{r}_t^g) \leq f(\mathbf{z}_t^g) \\ \mathbf{z}_t^g & \text{otherwise} \end{cases} \tag{2.19}$$

A sequence of mutation, cross-over and selection operations complete one generation of differential evolution. Multiple generations are completed by evolving the *SP* candidate solutions until the termination criteria is satisfied. Some commonly used termination criteria are preset objective function value, maximum number of generation and maximum computational time.

For a schematic example, differential evolution is applied to minimize a two-dimensional unimodal function f. A population of SP = 5 candidate solutions (indexed *1* to *SP*) is randomly initialized within preset parameter bounds and perturbed to create new points. The perturbation is done by adding the scaled difference of two random vectors to a third random vector in the population. Between the newly produced point $\mathbf{r}_1$ and parent vector $\mathbf{z}_1$, the solution which has the lower objective function value becomes a parent vector in the next generation (Figure 2.11). This procedure is repeated until all vectors in the initial population have competed against the newly produced vectors, and the parent vectors for the next generation are chosen.

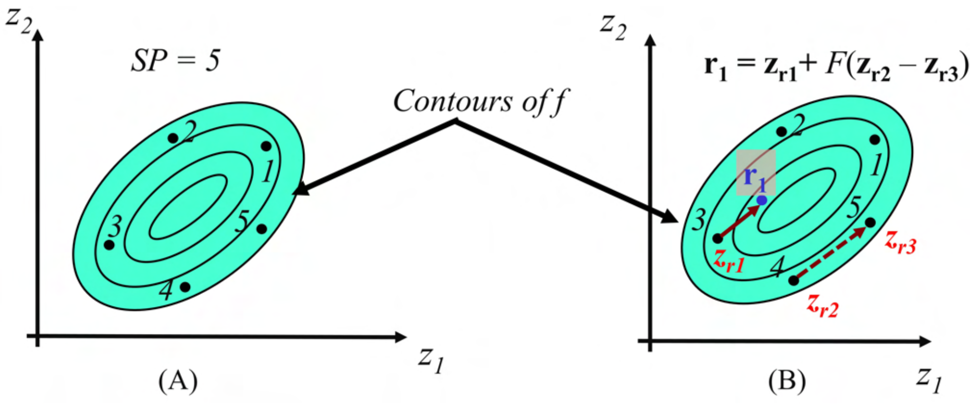


FIGURE 2.11: Schematic representation of differential evolution. (A) Initialization and (B) Perturbation

Role of control parameters in exploration and exploitation of the function space

Differential evolution is relatively simple to implement, the challenge is to find the right set of control parameters for computational efficiency. In fact, the successful application of any evolutionary algorithms requires the balancing of exploration and exploitation functions through the appropriate choice of control parameters (Gämperle, Müller, and Koumoutsakos, 2002; Sá et al., 2008). Exploration means visiting new regions of the function space to maximise the probability of finding the global minimum, whereas exploitation refers to the ability to look for better points in the vicinity of already found good points. In principle, exploration of the function space is beneficial in terms of finding new better solutions. However, the algorithm must also be capable of avoiding useless exploration and save the computational cost by refining (exploiting) an already found close-to-minimum solution. Exploration and exploitation functions are controlled through the parameters C_r, SP and F. Figure 2.12 presents a conceptual depiction of the control parameters influence on exploration and exploitation.

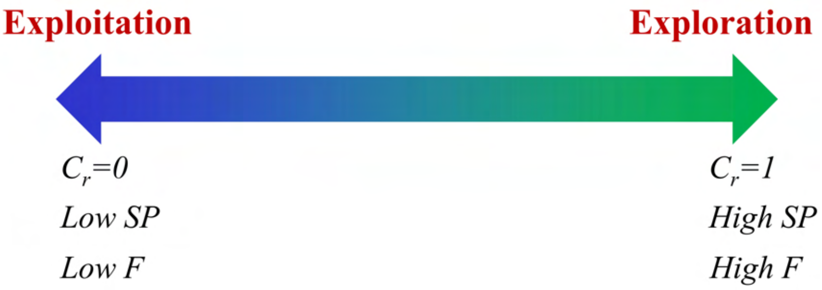


FIGURE 2.12: Exploration vs exploitation.

Among the three control parameters, the influence of the population size SP is straightforward to interpret. Larger population sizes thoroughly explore the function space and are more prone to reach the global minimum. A too small population size, despite incurring a lower computation cost, can produce an unsatisfactory (local) optimum due to insufficient exploration of the function space. The scale factor F determines the perturbation size to generate new candidate solutions from the existing ones. A small F promotes exploitation by perturbing only slightly the candidate solutions that remain close to the existing population, conversely, a large F promotes exploration. The cross-over probability C_r controls the diversity in the candidate solutions. New candidate solutions with C_r close to zero predominantly inherit from the parent vectors, which favours the exploitation of the function space. As a limiting scenario, $C_r = 0$ causes stagnation of the candidate solutions since the trial vectors become equal to the parent vectors in the crossover operation. On the other hand, the solutions from C_r close to one inherit mainly from the mutation vectors and favour exploration of the design space. When $C_r = 1$, the improvement of the existing solutions is impossible, as the trial vectors become the mutation vectors.

2.4 Enforcing constraints

The maximization or the minimization process are often performed respecting some constraints. For instance, two manufacturing constraints must be accounted during the optimal design of an FFF part: (i) the filaments cannot overlap into each other, and (ii) the filaments cannot be widely spaced to prevent sagging of filaments in above layers. Both the constraints are directly related to the filament path spacing, which is a design variable in the optimization problem studied.

The simplest, yet effective, method to impose such constraints is to reformulate the constrained minimization problem into an equivalent unconstrained form through penalty method. Let us recast the constrained minimization problem in Eq. (2.3) as given below:

$$p(\mathbf{z}, \epsilon_p) = f(\mathbf{z}) + \epsilon_p \sum_{j=1}^{m} G_j(g_j(\mathbf{z})) \tag{2.20}$$

where ϵ_p is the penalty coefficient that penalizes the objective function proportionally to the constraint violations, and G_j is some function of the constraint g_j. Usually, the general form $G_j = \{max[0, g_j]\}^q$ is used, where q is restricted to 1 or 2 in most practical problems (Rao, 2019). A larger penalty coefficient enables stricter constraint enforcement, whereas a small penalty coefficient permits larger constraint violations and produces solutions from infeasible regions. An optimal value of ϵ_p is not known a priori and is practically often chosen in a trial and error fashion by solving the unconstrained minimization problems for various ϵ_p.

Penalty methods are preferred owing to their simplicity and intuitive appeal. They can be employed for both gradient-based and differential evolution, although the numerical implications of the over- and under-penalization of the constraints are different in both the approaches. In gradient-based methods, over-penalizing constraints through large penalty coefficients prevents the solution from entering the infeasible space. But it also leads to ill-conditioning effects that increases the computational effort of the minimization process. On the other hand, being a population-based method, differential evolution does not prevent the candidate solutions from entering the infeasible space. Anyhow, by choosing a sufficiently high value of penalty coefficient, differential evolution can be discouraged from preferring the infeasible solutions for the subsequent generations. Note that the over-penalization in differential evolution can also lead to premature convergence to sub-optimal solutions. If the constraints separate the feasible function space into multiple islands (such as in the filament path optimization problem), then over-penalizing the constraints may not allow the candidate solutions to cross through and explore all the feasible islands (Price, Storn, and Lampinen, 2006). In such a case, increasing the exploratory factors can help overcome the effect of large penalty coefficients.

Besides the penalty method, there exists other popular approaches to solve constrained minimization problems, viz. the Lagrange multipliers and the augmented Lagrangian methods. The Lagrange multipliers method introduces additional variables to the minimization problem for the direct enforcement of constraints. Although this approach yields exact solutions, it poses difficulties due to the additional effort required to solve the multipliers. The augmented Lagrangian method combines the properties of the Lagrange multiplier and the penalty methods for avoiding the ill-conditioning issues seen in the latter.

Nevertheless, the present work uses the penalty method mainly because of its easy implementation and lesser computational effort.

2.5 A brief note on the software capabilities

The algorithmic framework for finding the optimal filament paths is developed in Mathematica (*Wolfram Mathematica 13*), employing its specialized packages called AceGen and AceFEM (Korelc, 2007). The minimization tasks are performed using the built-in functions of Mathematica, *FindMinimum* and *NMinimize*. The *FindMinimum* implements the gradient-based minimization, and *NMinimize* implements the differential evolution. On the other hand, the finite element tasks are handled by AceGen and AceFEM. AceGen uses a number of techniques such as, automated differentiation, symbolic and algebraic capabilities, and simultaneous optimization of expressions to efficiently code complex formulae needed in numerical procedures. It can convert the symbolic input into codes for many programming languages and has some standard modules for calculation of tangent stiffness matrix, post-processing and other complex tasks, enabling easy interfacing with the commercial FE environments. While AceGen handles the numerically intensive parts, AceFEM largely handles preprocessing and post-processing tasks such as collecting input data, mesh generation, control of solution procedures, result-visualization, and so on. Besides, MATLAB, 2018 has been used to develop the orientation-based filament deposition, generate G-code instructions and to visualize the filament paths designed.

Chapter 3

A Filament Path Optimization Framework For Production-Ready Designs

Chapter 2 presented the essential elements of filament path optimization, i.e., the prediction of FFF part behaviour, filament path parametrization, the minimization approaches and the enforcement of constraints. The present chapter implements these concepts to develop a new optimization framework that designs readily printable filament paths for minimizing the compliance of FFF parts. The developed framework is applied to optimize some two-dimensional FFF structures subject to prescribed forces and operating under linear elastic, small strain regimes. Then, the resulting optimized filament patterns are converted into G-code instructions, printed as per design specifications, and experimentally studied.

Besides, the chapter studies several aspects of the filament path optimization, particularly focusing on the following new contributions to the state-of-the-art: (i) the development and implementation of a new material model incorporating the phenomenological stiffness decrease at low filament densities that is shown to impact the resulting optimized filament patterns significantly, and (ii) the comparison of the two-step gradient-based minimization and the differential evolution in terms of efficiency, operability and compatibility with the newly proposed material model[1].

The organization of the chapter is as follows:
Section 3.1 sets up the filament path optimization problem to be solved, expressing the essential filament path quantities and the manufacturing constraints in terms of the design variables. Then, Section 3.2 develops a new material model accounting for the inter-filament gaps in an FFF layer. In

[1] This chapter is inspired from the paper published in the International Journal of Solids and Structures. DOI: 10.1016/j.ijsolstr.2022.111916

Section 3.3, the approaches to optimize the filament paths are presented, also depicting the developed algorithmic framework. Section 3.4 applies the framework to design filament paths for some example structures, and Section 3.5 shows the parts printed with the optimized filament paths. Section 3.6 presents an experimental comparison between the optimized and the standard filament patterns. Finally, Section 3.7 provides a summary of the chapter and concludes on this work.

3.1 Formulation of the optimization problem

To obtain optimized, production-ready filament paths, an indirect parametrization technique of Fernandez et al., 2019 is used. It expresses the filament path quantities in a layer and, hence, the manufacturing constraints and the material model as the contours or level-sets of a B-Surface (i.e., B-spline surface) defined over the layer (Fernandez et al., 2019). Then, the optimal filament trajectories are found as the solution to a minimization problem in which the shape parameters of the B-Surface are treated as the design variables. The series of steps inspired from the work of Fernandez et al., 2019 leading up to the formulation of the minimization problem and different solution approaches are presented first.

3.1.1 Filament path parameterization

The filament angle and spacing that describe the local filament path are formulated in terms of the contours of a B-Surface; they become continuous functions of the shape parameters of the B-Surface.

Let us assume a generic FFF component to optimize the filament paths and minimize its compliance (i.e., maximize its stiffness, assuming prescribed forces). The assumed component has layers of filaments printed on the $\mathbf{e_1}$-$\mathbf{e_2}$ plane and stacked in the $\mathbf{e_3}$ direction. Each extruded filament has a constant width w and a height h, as shown in Figure 3.1. The entire FFF component occupies a domain $\boldsymbol{\lambda}$ in three dimensional space that is sliced into individual domains λ_1, λ_2, ...λ_n representing the layers of the component. Under suitable assumptions on the component (cross-sections symmetrical to the mid-plane, loads non-varying along the $\mathbf{e_3}$, dimension in $\mathbf{e_3}$ smaller than along the other directions), the analysis is focused on a 2D domain Ω that represents the component mid-plane having superposed filament trajectories from layers $\lambda_{i=1,2..n}$ (Figure 3.1).

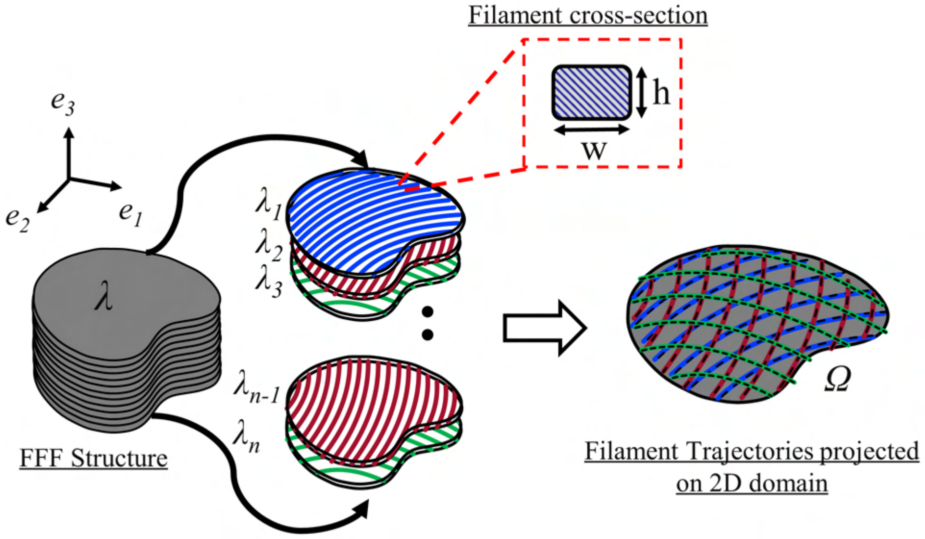


FIGURE 3.1: Two-dimensional assumption for FFF part. $\boldsymbol{\lambda}$ is composed of individual layers λ_1, $\lambda_2..\lambda_n$ having filaments of width w and height h. An equivalent 2D domain Ω is defined with filament trajectories from λ_1, $\lambda_2..\lambda_n$ projected on to it.

An ith in-plane filament layout on Ω can be described as the contours of a scalar function $\phi_i(x, y)$ defined over Ω (Honda and Narita, 2011; Huang, Wang, and Li, 2016),

$$C_i^k = \{\mathbf{x} \mid \phi_i(\mathbf{x}) = k\, w\},\ k \in \mathbb{Z},\ \mathbf{x} = (x, y) \in \Omega \tag{3.1}$$

where each contour in the set C_i^k, resulting from $\phi_i = 1w, 2w...kw$, represents a single trajectory in the ith filament layout on Ω (Figure 3.2).

Since the contours of ϕ represent filament trajectories, it follows that the local filament angle and the spacing can also be derived from the contours of ϕ. At any point $\mathbf{x} \in \Omega$, the filament angle $\alpha_i(\mathbf{x})$ is the angle between the tangent **t(x)** to the contour and the $\mathbf{e_1}$ axis (shown in Figure 3.2). Therefore, $\alpha_i(\mathbf{x})$ is expressed as,

$$\alpha_i(\mathbf{x}) = tan^{-1}\left(\frac{-\phi_{i,x}}{\phi_{i,y}}\right) \tag{3.2}$$

where $\phi_{i,x}$ and $\phi_{i,y}$ are partial derivatives of the ϕ_i with respect to x and y, respectively (Honda and Narita, 2011; Huang, Wang, and Li, 2016). In a similar fashion, the filament spacing $l_i(\mathbf{x})$ is also denoted using the contours of ϕ_i. As Figure 3.2 illustrates, the filament spacing $l_i(\mathbf{x})$ is the sum of the distances from $\mathbf{x}$ to the two closest points in the adjacent contours, where $\Delta\mathrm{\mathbf{x}_k}$ and $\Delta\mathrm{\mathbf{x}_{k\text{-}1}}$ are correspondingly the distances from $\mathbf{x}$ to the closest points $\mathbf{x}^*_\mathrm{k}$ and $\mathbf{x}^*_\mathrm{k\text{-}1}$ in the two adjacent contours C_i^k and C_i^{k-1}. Thus, the spacing $l_i(\mathbf{x})$

is related to ϕ_i as given below (refer to Fernandez et al., 2019 for derivation).

$$l_i(\mathbf{x}) \approx \frac{w}{|\nabla \phi_i(\mathbf{x})|} \tag{3.3}$$

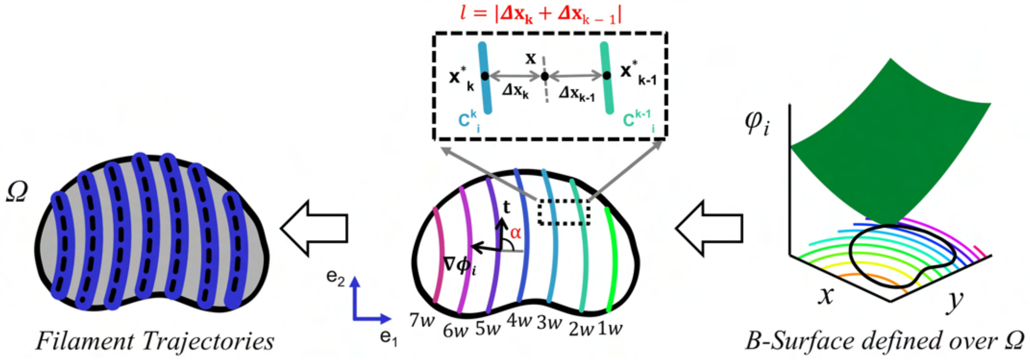


FIGURE 3.2: In-plane filament trajectories represented using contours of a B-Surface. α_i : filament angle; l_i : filament spacing; $\mathbf{x}^*_\mathrm{k}$: closest point in contour C^k_i from $\mathbf{x}$; $\mathbf{x}^*_{\mathrm{k}\text{-}1}$: closest point in contour C^{k-1}_i from $\mathbf{x}$.

A B-surface (or a B-spline surface) results from the tensor product of one-dimensional B-spline functions in two parametric domains. It is composed of individual entities called patches, the shapes of which are influenced by a set of control points, as shown in Figure 3.3. The following set of equations define a single patch of the ith B-surface defined over Ω (Bartels, Beatty, and Barsky, 1995, Marschner and Shirley, 2015):

$$x(\Xi) = \mathbf{N}(\Xi).\mathbf{X}^\mathbf{L};\ y(\Xi) = \mathbf{N}(\Xi).\mathbf{Y}^\mathbf{L};\ \phi_i(\Xi) = \mathbf{N}(\Xi).\mathbf{d}^\mathbf{L}_i \tag{3.4}$$

where $\mathbf{N}(\Xi)$ is the vector of bicubic basis functions, the symbol $\Xi = (\xi, \eta)$ represents the coordinates of the patch in parent space, while the vectors $\mathbf{X}^\mathbf{L}$, $\mathbf{Y}^\mathbf{L}$ and $\mathbf{d}^\mathbf{L}_i$ denote the 3D positions of a grid of control points that influence the patch in physical space (see Figure 3.3). The local grid of control points is, in turn, a subset of all the control points of the B-surface, the 3D positions of which are denoted as $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{d}_i$, respectively. All three vectors have the same dimensions and assume the distance units of the FFF component to be optimized (for e.g., mm).

Eq. (3.4) relates ϕ_i to the local control point heights $\mathbf{d}^\mathbf{L}_i$ of the ith B-surface, therefore enabling the contours of ϕ_i, and in turn, the filament path angle α_i (Eq. (3.2)) and spacing l_i (Eq. (3.3)) to be controlled using them. Thus, keeping the $\mathbf{X}$ and $\mathbf{Y}$ constant, the control point heights $\mathbf{d_i}$ are varied to design the filament trajectories. If 'n' sets of filament trajectories are represented using

'n' number of B-Surfaces, then the entries of the entire vector of control point heights of the B-Surfaces, $\mathbf{d} = \{\mathbf{d_1}, \mathbf{d_2}, ...\mathbf{d_n}\}$, become the design variables for the filament trajectories in the FFF structure. The total number of design variables (N_{dv}) is determined from the number of B-surface patches as given below,

$$N_{dv} = n\, CP_x\, CP_y = n(P_x + 3)(P_y + 3) \tag{3.5}$$

where CP_x and CP_y denote the number of control points used per B-Surface in $\mathbf{e_1}$ and $\mathbf{e_2}$ directions, with P_x and P_y being the corresponding patch number in each direction. For example, Figure 3.3 employs a total of $N_{dv} = 42$ design variables in a single layer ($n = 1$).

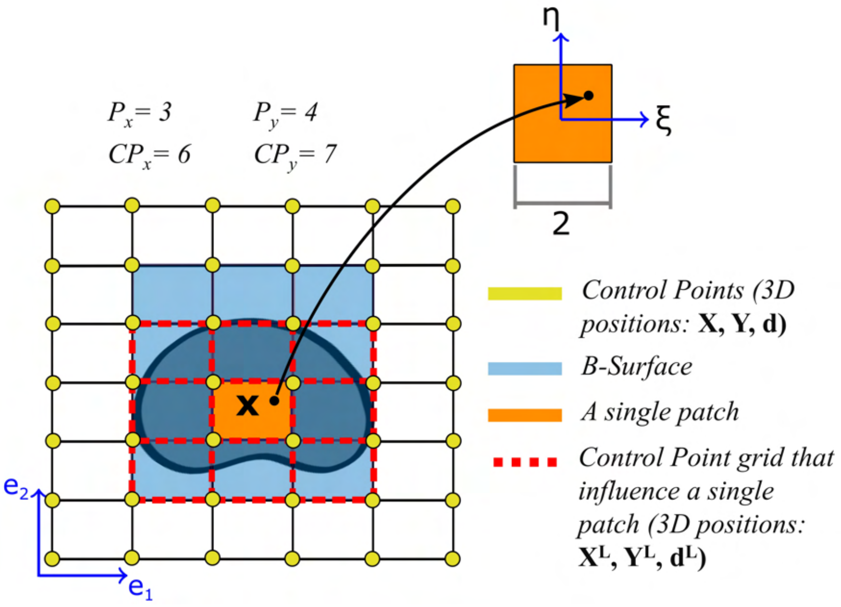

η
ξ
P_x= 3
CP_x= 6
P_y= 4
CP_y= 7
2
x
e₂
e₁
Control Points (3D positions: X, Y, d)
B-Surface
A single patch
Control Point grid that influence a single patch (3D positions: X^L, Y^L, d^L)

FIGURE 3.3: Patches of a uniform bicubic B-Surface defined over a layer. P_x : Number of patches along $\mathbf{e_1}$; P_y : Number of patches along $\mathbf{e_2}$; CP_x = P_x + 3 : Number of control points along $\mathbf{e_1}$; CP_y = P_y + 3 : Number of control points along $\mathbf{e_2}$.

3.1.2 Imposing manufacturing constraints

Production-ready solutions can be yielded from design optimization, provided proper manufacturing constraints are incorporated in the process. Here, two essential constraints are included (Figure 3.4): (i) the *no-overlap* constraint between neighbouring filaments, i.e., the path spacing l_i is set larger than or equal to the filament width ($l_i \geq w$) and (ii) the *no-sag* constraint that states that any two adjacent filaments cannot be spaced larger than a prescribed value of l_{max} ($l_i < l_{max}$) to avoid sagging of the filaments deposited on the previous layer (Figure 3.4), where l_{max} is a parameter chosen based on the printer capabilities to bridge two unsupported points.

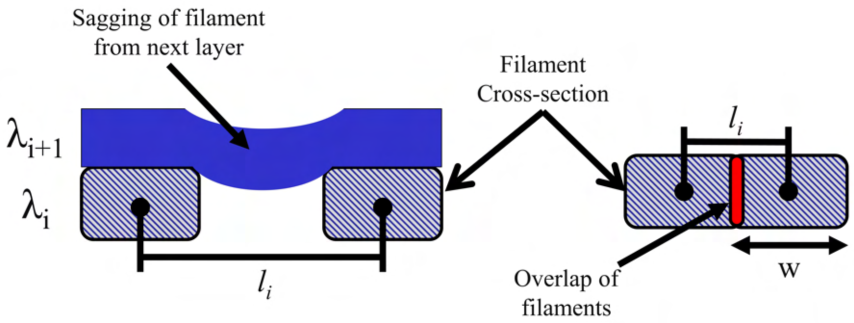


FIGURE 3.4: Manufacturing filament spacing constraint; sag and overlap, respectively, define the maximum and minimum filament spacing

The aforementioned manufacturing constraints are equivalently expressed as constraints on the gradient of ϕ_i for relating them to the design variables. The expressions for no-overlap and no-sag constraints are derived from Eq. (3.3) and stated as follows, respectively:

$$|\nabla\phi_i| \leq 1 \tag{3.6}$$

$$|\nabla\phi_i| \geq \frac{w}{l_{max}} \tag{3.7}$$

It is emphasized that Eqs. (3.6) and (3.7) express *local* nonlinear constraints, which must be respected at every point in a layer λ_i. Due to the practical challenges in implementing the local constraints (Amstutz, 2010; Amstutz and Novotny, 2010; Amstutz, Novotny, and Souza Neto, 2012), the following integral expressions are defined (Le et al., 2010; Fernandez et al., 2019):

$$G_i^a = \int_\Omega S_t(|\nabla\phi_i|^2 - 1)d\Omega \tag{3.8}$$

$$G_i^b = \int_\Omega S_t\left(1 - \left(|\nabla\phi_i|\frac{l_{max}}{w}\right)^2\right)d\Omega \tag{3.9}$$

where a step function S_t is introduced to consolidate the positive values, as given below.

$$S_t(\Theta) = \begin{cases} 0, & \text{if } \Theta \leq 0 \\ 1, & \text{if } \Theta > 0 \end{cases} \tag{3.10}$$

Note that the parameter Θ indicates $(|\nabla\phi_i|^2 - 1)$ and $(1 - (|\nabla\phi_i|l_{max}/w)^2)$ from Eqs. (3.8) and (3.9), respectively. Thus, the overlap and sagging constraints are violated if G_i^a and G_i^b are positive, and are satisfied if G_i^a and G_i^b are zero.

3.1.3 The minimization problem

Having expressed the printing trajectories (refer to Eqs. (3.2), (3.3) and (3.4)) and manufacturing constraints (refer to Eqs. (3.6) and (3.7)) in terms of the design variables, the optimization problem can now be defined. The objective is to find the optimal control point heights of the B-Surfaces that minimize the compliance of Ω, subject to no-overlap and no-sag constraint[2]. The corresponding mathematical formulation is given as,

$$\begin{aligned} \underset{\mathbf{d_1,d_2..d_n}}{\text{minimize}} \quad & \text{compliance (c)}, \\ \text{subject to:} \quad & G_i^a = 0, \;\; i = 1, 2, ..n \\ & G_i^b = 0, \;\; i = 1, 2, ..n \end{aligned} \tag{3.11}$$

where the objective/cost function to be minimized is the compliance $c = \int_\Omega \sigma : \epsilon \, d\Omega$ (twice the strain energy) computed from the Cauchy stress tensor σ and the infinitesimal (linear) strain tensor ϵ using finite element analysis (Brampton, Wu, and Kim, 2015; Esposito et al., 2019) .

The above minimization problem is recast into the following unconstrained minimization problem using the penalty method (Nocedal and Wright, 2006; Price, Storn, and Lampinen, 2006; Ali and Zhu, 2013),

$$\underset{\mathbf{d_1,d_2,..d_n}}{\text{minimize}} \qquad \mathrm{c} + \epsilon_{\mathrm{p}} \left(\sum_{i=1,2,..n} (G_i^a)^2 + (G_i^b)^2 \right) \tag{3.12}$$

where the quadratic function p = c + ϵ_{p} $((G^a_{i=1..n})^2 + (G^b_{i=1..n})^2)$ becomes the new *penalized cost function* to be minimized, and ϵ_p (> 0) is the penalty coefficient that penalizes the new cost function.

[2] The equality constraints mentioned in the optimization problem in Eq. (3.12) are practically inequality constraints since the negative values of G_i^a and G_i^a vanish by construction of Eqs. (3.8) and (3.9)).

3.2 Anisotropic material model accounting for the effect of inter-filament spacing

The mechanical performance of FFF parts relies upon a multitude of FFF parameters, among which the filament angle and density are widely recognized to play dominant roles (Fayazbakhsh, Movahedi, and Kalman, 2019; Cuan-Urquizo et al., 2019). A noteworthy highlight of this work is, in fact, a new material model that predicts the component response accounting for the local filament angle and density.

The proposed material model extends the classical approach of Fernandez et al., 2019, in which a transversely isotropic elasticity tensor is rotated by the local filament angle $\alpha_i(\mathbf{x})$ and linearly scaled by the local density/volume-ratio $\nu_i(\mathbf{x})$, where the *volume ratio*, expressed below, is the amount of filament material in a $l_i \times l_i \times h$ Representative Volume Element (Figure 3.5).

$$\nu_i(\mathbf{x}) = \frac{l_i b h}{l_i^2 h} \approx |\nabla \phi_i(\mathbf{x})| \tag{3.13}$$

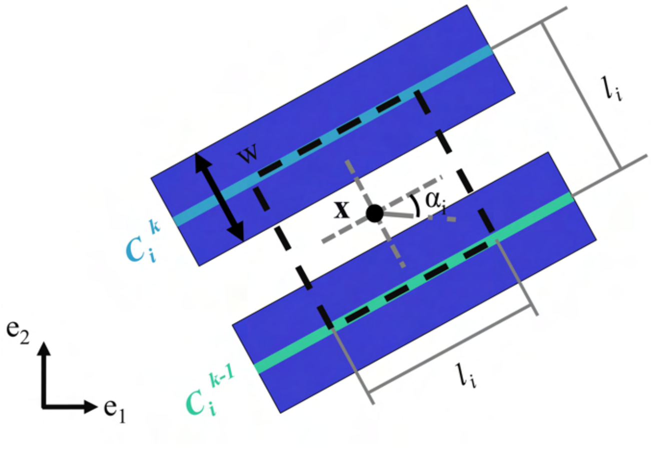


FIGURE 3.5: Definition of volume ratio. ν_i is the ratio of filament material in the $l_i \times l_i \times h$ Representative Volume Element.

The simplified technique to linearly scale all the stiffness components with the volume ratio assumes significant transverse stiffness even when lacking inter-filament fusion, thereby ignoring the filament bonding mechanism and leading to less realistic printing patterns (as will be shown in Section 3.4). Thus, to avoid such solutions, a material model incorporating the phenomenological effect of inter-filament spacing is developed below.

Let $\Psi_i(\mathbf{x})$ denote the strain energy density due to the ith filament layout. Then, the strain energy density $\Psi(\mathbf{x})$ for the whole FFF part is the sum of individual contributions from the strain energy densities due to n filament layouts on Ω (from λ_1, $\lambda_2 ... \lambda_n$), i.e.,

$$\Psi(\mathbf{x}) = \Psi_1(\mathbf{x}) + \Psi_2(\mathbf{x}) + ..\Psi_n(\mathbf{x}) \tag{3.14}$$

Taking the assumptions of linear elasticity and small-strains, and recalling $\sigma = \partial\Psi/\partial\boldsymbol{\epsilon}$, the strain energy density $\Psi_i(\mathbf{x})$ is expressed as a quadratic function of the strain tensor $\boldsymbol{\epsilon}$,

$$\Psi_i(\mathbf{x}) = \frac{1}{2}\,\boldsymbol{\epsilon}(\mathbf{x}) : \mathbf{C}_i^*(\mathbf{x}) : \boldsymbol{\epsilon}(\mathbf{x}) \tag{3.15}$$

where the elasticity tensor $\mathbf{C}_i^*(\mathbf{x})$ at a point $\mathbf{x} \in \Omega$ is a function of the filament angle α_i and the volume-ratio ν_i at that point, i.e., $\mathbf{C}_i^*(\alpha_i(\mathbf{x}), \nu_i(\mathbf{x}))$.

If transverse isotropy is assumed in layer λ_i, then the elasticity tensor $\mathbf{C}_i^*(\mathbf{x})$ has five independent components in a local coordinate system: s_{11}, s_{22}, s_{12}, s_{44} and s_{23}, where the adopted indices are consistent with the Voigt notation (Kollar and Springer, 2003). The goal is to represent that the stiffness component along the filament direction s_{11} dominates if adjacent filaments do not touch each other, i.e., for low values of volume ratio ν_i. The other components s_{22}, s_{12}, s_{44} and s_{23} come into play only when the inter-filament bonding occurs, i.e., when a given minimum threshold of ν_i has been reached. Modelling this behaviour implies that the printing and the transverse direction stiffness are treated differently with respect to ν_i, (a decoupled behaviour).

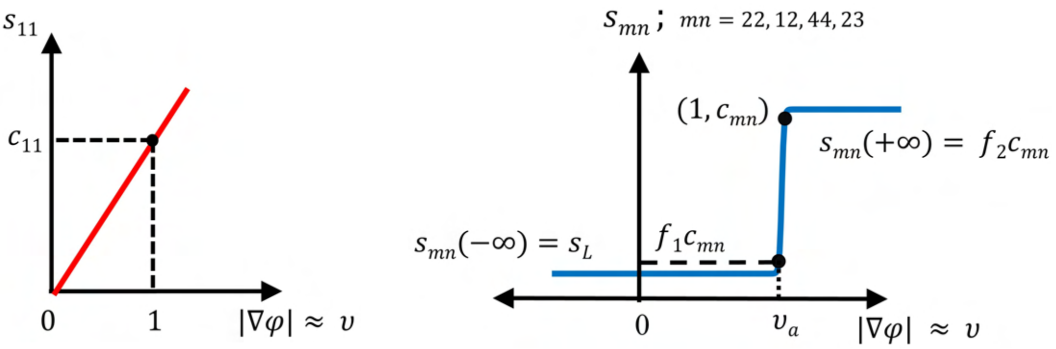
Figure 3.6 sketches the assumed evolution of the five stiffness components as functions of the volume ratio. The stiffness along the filament extrusion direction s_{11} is modelled as a linear function of ν_i as follows,

$$s_{11}(\nu_i) = c_{11}\nu_i \tag{3.16}$$

where c_{11} is the stiffness component along the filament direction at full volume ratio (i.e. corresponding to a volume filled at 100% by the filament material). On the other hand, the components s_{22}, s_{12}, s_{44} and s_{23} are almost zero for widely spaced adjacent filaments. When the filaments start to bond with each other, the four stiffness components rapidly rise and subsequently reach a saturated maximum value once the filaments are completely fused. Figure 3.6b shows a hyperbolic tangent curve that is assumed to model this behaviour. Accordingly the following expression is stated for the stiffness components,

$$s_{mn}(\nu_i) = a_3 \tanh(a_1\nu_i + a_2) + a_4 \tag{3.17}$$

where $tanh(z) = (e^z - e^{-z})/(e^z + e^{-z})$, and s_{mn} corresponds to each of the four stiffness components s_{22}, s_{12}, s_{44} and s_{23}, for which the unknown constants a_1, a_2, a_3 and a_4 must be individually determined.



(A)

(B)

FIGURE 3.6: Stiffness variation as function of the volume-ratio. (A) s_{11} is linear function of ν. (B) The components $s_{22}, s_{12}, s_{44}, s_{23}$ are hyperbolic tangent functions of ν. Only s_{11} is significant before inter-filament fusion. The other components are very low for $\nu \leq \nu_a$ and reach a saturated state once the filaments are completely fused.

To compute the unknown constants in Eq. (3.17), the following system of equations that result from the conditions at $\nu_i = \nu^l, \nu^a$, 1 and ν^u is solved.

$$s_{mn}(\nu^l) = s_L \tag{3.18a}$$

$$s_{mn}(\nu^a) = f_1 c_{mn} \tag{3.18b}$$

$$s_{mn}(1) = c_{mn} \tag{3.18c}$$

$$s_{mn}(\nu^u) = f_2 c_{mn} \tag{3.18d}$$

where ν^l and ν^u, respectively, denote the lower and upper limits of the volume ratio ν_i in a layer. In a physical sense, ν^l is equal to zero, and ν^u can be slightly more than one, provided a meagre percentage of inter-filament overlap is allowed. However, assuming that a saturated state already exists at volume ratios ν^l and ν^u, both quantities are set to $-\infty$ and $+\infty$, respectively, for the analytical solution of Eq. (3.18). The volume ratio ν^a denotes the *activation point* at which the inter-filament bonding is assumed to start. Accordingly, ν_a must be chosen close to one ($\nu_a \rightarrow 1$) for a realistic inclusion of the filament spacing effect. The parameter s_L denotes a very small user-defined value for the four stiffness components when the filaments are widely spaced (at $\nu_i = \nu^l$), whereas c_{mn} denote their values when the filaments are fused, i.e., at 100% volume ratio ($\nu_i = 1$). Finally, the quantities f_1 and f_2 are the user-defined fractional constants that scale the stiffness c_{mn} at $\nu_i = 1$ and assign them at ν^a and $\nu_i = \nu^u$, respectively.

After determining all the unknown constants in Eq. (3.17), the transverse isotropic elasticity tensor is assembled in the local coordinate system and rotated by the filament angle $\alpha_i(\mathbf{x})$ to find the final elasticity tensor $\mathbf{C}_i^*(\mathbf{x})$.

Thus, the proposed material model uses the local mesostructural data (angle and spacing) to compute the stiffness at a point. However, it cannot predict the local failure mechanisms due to weak inter-filament bonding or considers the influence of the unit-cells in partially filled parts. In such a case, dedicated micromechanical models are more suitable.

3.3 Solution methodologies and the depiction of optimization framework

The minimization problem in Eq. (3.12) is solved using the two-step gradient based minimization and differential evolution. Both methods are applied for filament path optimization in Section 3.4 and are comparatively evaluated in terms of solution quality, computational cost and operational difficulties.

For the two-step gradient-based method, first an initial optimized design with straight and parallel toolpaths having uniform orientations $\alpha^{\text{in}}_{i=1,2..n}$ is obtained. Then, the control point heights $\mathbf{d}^{\text{in}}_{i=1,2..n}$ corresponding to the uniform orientations are supplied as the starting point to solve the curvilinear filament optimization in Eq. (3.12). The initial control point heights are calculated as:

$$\mathbf{d}_{\mathbf{i}}^{\mathbf{in}} = -\sin(\alpha_i^{\text{in}})\mathbf{X} + \cos(\alpha_i^{\text{in}})\mathbf{Y} \tag{3.19}$$

where $\mathbf{X}$ and $\mathbf{Y}$ are the vectors of control point coordinates on $\mathbf{e}_1$-$\mathbf{e}_2$ plane for any ith B-surface. On solving the minimization problem in Eq. (3.12), convergence is reported at a kth step if the following criteria are satisfied,

$$||\Delta\mathbf{d}^k|| \leq 10^{-ag_1} \; \& \; ||\nabla\mathbf{p}(\mathbf{d}^k)|| \leq 10^{-ag_2} \tag{3.20}$$

where $\Delta\mathbf{d}^k$ indicates the difference in the vector of design variables at the kth step and previous steps, the quantity $\nabla\mathbf{p}(\mathbf{d}^k)$ is the gradient of the penalized objective function p at the kth step, while ag_1 and ag_2 (*AccuracyGoal*) are positive integers that specify the convergence tolerance. Note that the convergence criteria in Eq. (3.20) are affected by the dimensions/units assumed for the structure. To avoid this normalized quantities can be used in the convergence criteria. The convergence is accelerated by the use of analytical sensitivities, computed from a combination of the adjoint method and the automatic differentiation. Following is the expression for compliance sensitivity with respect to the *i*th layer design variables (Fernandez et al., 2019):

$$\frac{dc}{d\mathbf{d}_i} = -\int_{\Omega} \boldsymbol{\epsilon}^T \frac{\partial \mathbb{C}_i^*}{\partial \mathbf{d}_i} \boldsymbol{\epsilon} \, d\Omega \tag{3.21}$$

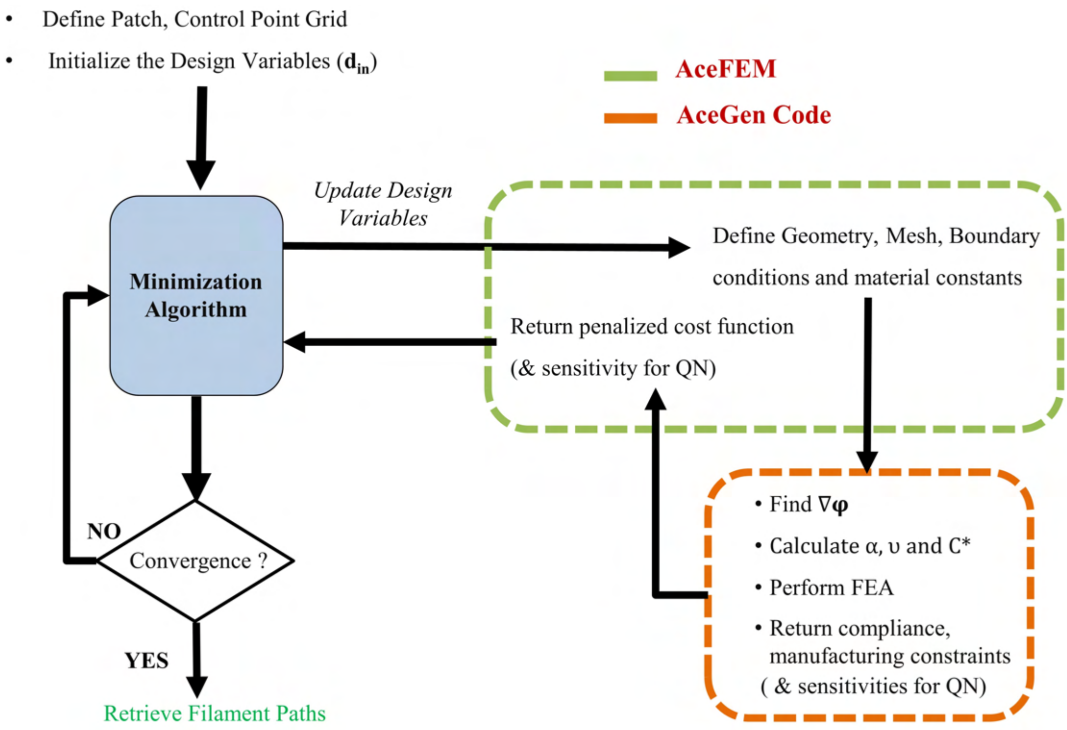

• Define Patch, Control Point Grid
• Initialize the Design Variables ($\mathbf{d}_{in}$)
AceFEM
AceGen Code
Update Design Variables
Minimization Algorithm
Define Geometry, Mesh, Boundary conditions and material constants
Return penalized cost function (& sensitivity for QN)
• Find ∇φ
• Calculate α, υ and C*
• Perform FEA
• Return compliance, manufacturing constraints (& sensitivities for QN)
NO
Convergence ?
YES
Retrieve Filament Paths

FIGURE 3.7: Optimization framework developed in Mathematica, AceGen and AceFEM. AceGen is used to develop FE codes, while AceFEM does the preprocessing and post-processing tasks.

Differential evolution that is also applied here is insensitive to the gradient of the objective function and works with a population of candidate solution to probe the function landscape for the global minimum. The search for the minimum solution continues until the preset maximum number of generations (G_{max}) is reached or if the difference between the best function values in the new and old populations, as well as the distance between the new best point and the old best point, are less than a tolerance of 10^{-ag}, with ag being a user-set positive integer (Brett Champion, 2008). Differential evolution is likely to reach (near) global minimum of complex functions without getting trapped at local solutions. However, it incurs a higher computational cost and is highly reliant on the values of control parameter (SP, C_r and F), which are found in this work using a trial and error approach.

The workflow of the entire framework developed in Mathematica summarizing each step is depicted in Figure 3.7. While differential evolution is already available as Mathematica's built-in minimiser, the two-step gradient based minimization is implemented using the Quasi-Newton algorithm in Mathematica. To evaluate the structural performance, the finite element procedures are implemented using AceGen and AceFEM packages (Korelc, 2007). Appendix B provides the AceGen and AceFEM codes developed for the filament path optimization.

3.4 Application to two-dimensional structures

The described methodology is applied to optimize the filament paths of 2D structures in three separate sections. Section 3.4.1 makes a comparative assessment of the two-step gradient-based minimization and the differential evolution. Section 3.4.2 evaluates the benefits of optimized filament paths against the standard straight filament patterns. Finally, Section 3.4.3 compares the proposed material model with the model of Fernandez et al., 2019 to investigate the effect of decoupling the printing and transverse stiffness components.

3.4.1 Two-step gradient-based minimization-Vs-Differential evolution

Prior to comparing the minimization approaches, the influences of essential parameters like number of patches (P_x, P_y), FE mesh and penalty coefficient on the optimization process are studied. Based on the insights on these user-parameters, the subsequent steps assume suitable values for the parameters to apply and compare both minimization approaches.

This section employs an MBB (Messerschmitt-Bölkow-Blohm) beam to optimize the filament paths. The corresponding geometry, boundary conditions, and an example of patch arrangement for the optimization are given in Figure 3.8. Note that arbitrary load magnitudes have been assumed for the structure, as the applied magnitude does not influence the optimized solutions under linear elastic assumptions. A total out-of-plane thickness of $t = 1$

mm, a filament width of $w = 0.6$ mm and a maximum allowable in-plane gap of l_{max}=1.6 mm are assumed (as in Fernandez et al., 2019). At full-volume fraction, the FFF parts are assumed to be strongly anisotropic with the following properties of a fiber-reinforced composite: $c_{11} = 152.47$ GPa, $c_{22} = 15.44$ GPa, $c_{12} = 7.46$ GPa, $c_{44} = 4.550$ GPa and $c_{23} = 9.410$ GPa (Kollar and Springer, 2003), and with the following material model parameters: ν^a=0.9, $s_L = 10^{-4}$ GPa, $f_1 = 10^{-3}$ and $f_2 \approx 1.03$ to 1.1.

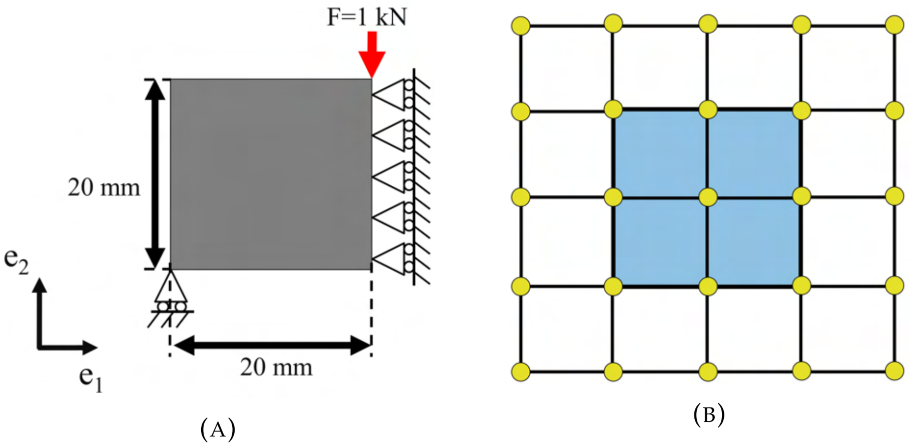


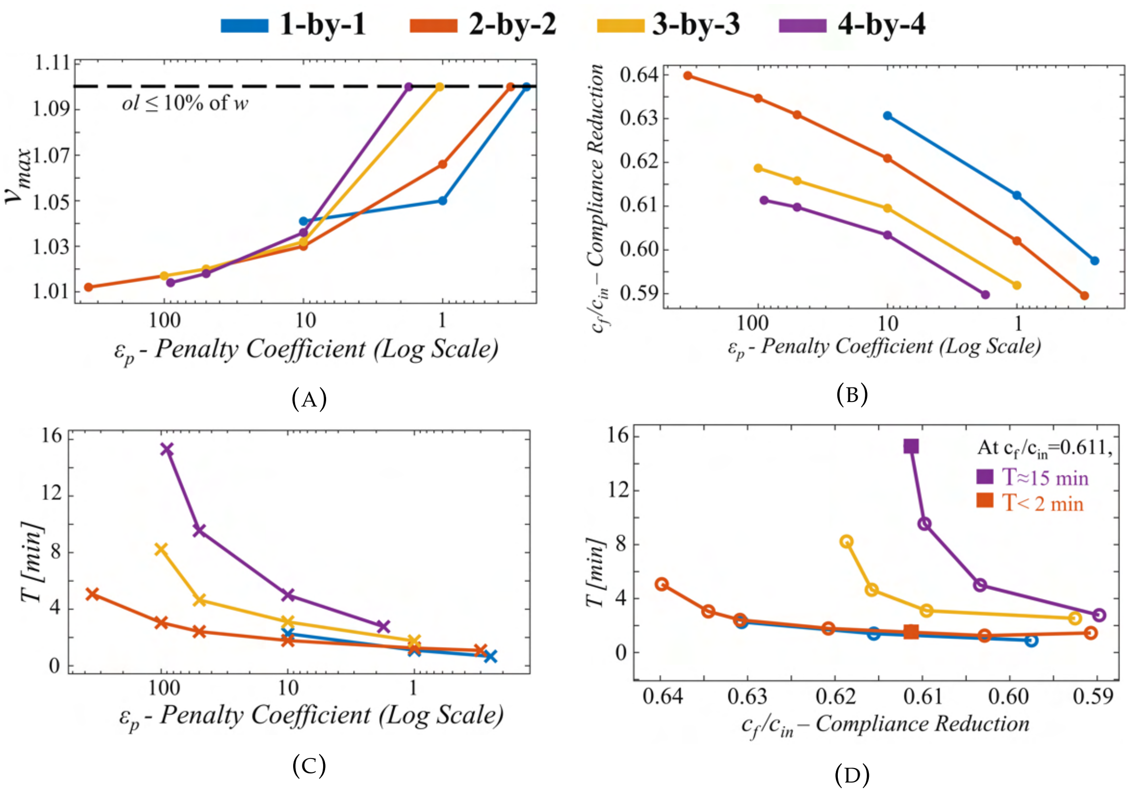
FIGURE 3.8: MBB structure and patch arrangement. (A) Geometry of MBB structure assuming symmetric boundary conditions along the midside vertical axis and a load of F = 1kN, (B) Patch arrangement with $P_x = 2$, $P_y = 2$ yields $N_{dv} = 50$ for $n = 2$.

Parametric study on patch number, FE mesh and penalty coefficient

Two parametric studies are conducted while optimizing the MBB structure: (i) the roles of the B-surface patch number and the penalty coefficient are investigated simultaneously for a given FE mesh, and (ii) the effect of refining the FE mesh is examined, keeping the number of patches constant. For simplicity, both studies are conducted using the two-step gradient based minimization, assuming a single layer and imposing only the no-overlap condition. A tolerance ag_1=2/ag_2=2 is prescribed since convergence criterion could not be satisfied for stricter tolerance.

The first parametric study uses a 32-by-32 bilinear FE mesh and conducts the filament path optimization for four different patch arrangements P_x-by-P_y: 1-by-1, 2-by-2, 3-by-3 and 4-by-4. The corresponding initial design variables $\mathbf{d}^{\text{in}}$ are computed from α^{in}=70.59° and the minimization problem in Eq. (3.12) is solved for a range of penalty coefficients ϵ_p. Starting from the highest possible value, the penalty coefficient is reduced ensuring that the overlap

$ol_i(\mathbf{x}) = w - l_i(\mathbf{x})$ in the optimum solution is kept within a reasonable limit. For example, a maximum overlap of $ol_i \leq 10\%$ of w has been used here, as higher values of overlap were found to affect the smoothness of the printed part. Besides, an overlap equivalent to 10% of w is also typically used by commercial slicing software.



(A)

(B)

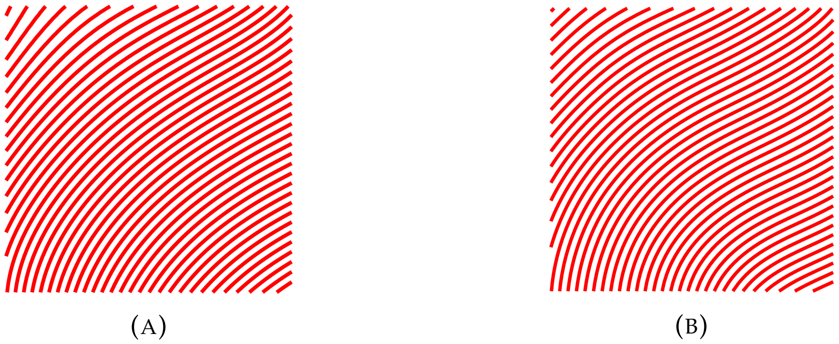
(C)

(D)

FIGURE 3.9: Parametric study on penalty coefficient and patch number. (A) Maximum volume ratio v_{max} increases with decreasing ϵ_p. Minimization process do not converge for high penalty coefficient ϵ_p and the condition, ol ≤10% of w, sets the lower limit of ϵ_p. (B) For most ϵ_p values, a higher number of patches reaches a lower compliance reduction c_f/c_{in}. (C) Computational time T increases with the number of patches, particularly at high ϵ_p. (D) For reaching a given value of compliance reduction c_f/c_{in}, the 3-by-3 and 4-by-4 patches requires a much higher computational time T.

For the assumed patch numbers, Figure 3.9a depicts the maximum volume ratio v^{max} obtained with different penalty coefficients ϵ_p. Figures 3.9b and 3.9c, respectively, depict the ratio of final compliance to the initial compliance (c_f/c_{in}) and the computational time (T) against penalty coefficients ϵ_p, while Figure 3.9d shows the computational time T, taken for achieving a given value of compliance reduction c_f/c_{in}.

Figure 3.9a indicates a trend typical to any penalty-based optimization process. Regardless of the patch number, the decrease in the penalty coefficient ϵ_p increases the maximum volume ratio v^{max}, denoting the proportional degree of constraint violation (overlap) in the optimum solutions. The limit set on the overlap *ol* dictates the lowest penalty coefficient, whereas its upper limit is due to the effect of *ill-conditioning*, inherent to the penalty methods of gradient-based approaches (Nocedal and Wright, 2006). Any attempt for a stronger constraint imposition using higher penalty coefficients than those shown in Figure 3.9a does not guarantee convergence and may prematurely terminate the minimization process. Therefore, suitable penalty coefficients are chosen by a trial and error approach, adopting a compromise between imposing constraints to the desired extent and achieving convergence quickly.



(A)

(B)

FIGURE 3.10: Comparison of the optimized filament paths from 1-by-1 and 4-by-4 patches under the same initial conditions. (A) Centerlines of the optimized filaments for 1-by-1 patch with $c_{\mathrm{f}}/c_{\mathrm{in}} = 0.63$. (B) Centerlines of the optimized filaments for 1-by-1 patch with $c_{\mathrm{f}}/c_{\mathrm{in}} = 0.60$.

Figure 3.9b illustrates that the higher patch numbers yield lower compliance reduction factors $c_{\mathrm{f}}/c_{\mathrm{in}}$ for most penalty coefficients ϵ_p. Clearly, a higher patch number reaches a reduced compliance by exploiting a larger design space. As an example, the difference between the optimized filament paths from 1-by-1 and 4-by-4 patches is shown in Figure 3.10. However, despite the improvements to the final design, the respective numerical implications must also be accounted. Figure 3.9c shows that the computational time T for the 3-by-3 and 4-by-4 patches significantly increases as the ϵ_p gets higher, a sign common to ill-conditioned systems. As a result, Figure 3.9d shows that a given compliance reduction of $c_{\mathrm{f}}/c_{\mathrm{in}}$=0.61 is attained by the 4-by-4 patch in $T \approx 15$ minutes, for which the 2-by-2 patch takes less than 2 minutes. Thus, as expected, the increased design freedom of a higher patch number is achieved at a higher computational cost that increases further when constraints are enforced stricter.

Choosing the number of B-surface patches is problem dependent. However, deducing from these results, it is recommended to initially opt for a lower patch number; more patches can be chosen to refine the results if they appear too crude or if further gain in stiffness is expected. Alternately, the solution from using a lower number of patch number can be used as a starting point for an optimization with a higher number of patches. Although this can speed up the optimization process, care must be taken to ensure that the B-surface remains uniform after refinement since the optimization framework does not work for non-uniform B-surfaces. In any case, the patch number is only a simulation parameter and does not affect the printing process.

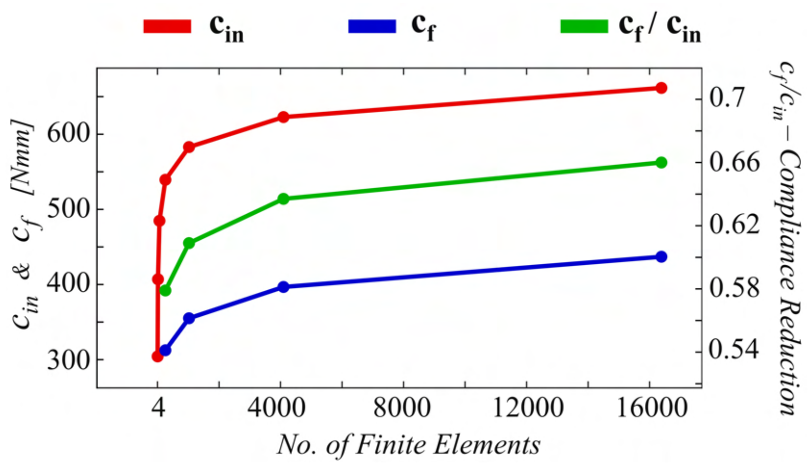


FIGURE 3.11: Parametric study on the influence of mesh size with a 2-by-2 patch arrangement. c_f and c_f/c_i increase asymptotically at higher number of finite elements. For too few finite elements, the minimization process did not converge.

The second parametric study uses a 2-by-2 patch and solves the minimization problem (Eq. (3.12)) for a series of different FE mesh, varying the element numbers as : $2^m \times 2^m$, with $m = 1, 2...7$. As previously discussed, the penalty coefficients in all cases are set through a trial and error method to maintain the same amount of constraint violation in the final solutions and facilitate their comparison. Figure 3.11 plots the initial compliance c_{in}, final compliance c_f and the compliance reduction c_f/c_{in} against the number of finite elements used.

The initial compliance c_{in} shows a sharp increase at the beginning that later becomes asymptotic as the finite element meshes are refined. As expected, coarser meshes are stiffer and increasing the number of finite elements leads to converged meshes with a lower stiffness. Note that the final compliance c_f and the compliance reduction $c_\mathrm{f}/c_\mathrm{in}$ follow trends similar to the initial compliance c_in, although both c_f and $c_\mathrm{f}/c_\mathrm{in}$ are not plotted for low number of finite elements. The reason is that the optimisation fails to converge for too few finite elements due to the poor accuracy of the solutions from coarse meshes. Anyhow, refinement of the FE mesh does not significantly alter the filament path solutions. Therefore, it is recommended to use a sufficient number of finite elements (approaching a converged mesh), mainly to ensure the accuracy of the FE results and to enable convergence at the optimum solutions.

Based on the results presented so far, the 2-by-2 patch arrangement and 32-by-32 elements FE mesh are considered suitable for the comparative study of the two minimization approaches. A 2-by-2 patch is chosen to gain a good balance between the design space, computational cost and operable range of penalty coefficient (Figure 3.9). The 32-by-32 FE mesh is chosen because it offers a good compromise between computational cost and accuracy. Although it is not a properly converged mesh, trends on the optimization approaches' performance were observed to remain valid independently of the FE mesh used with respect to further mesh refinement.

Two-Step gradient based minimization: filament path solutions

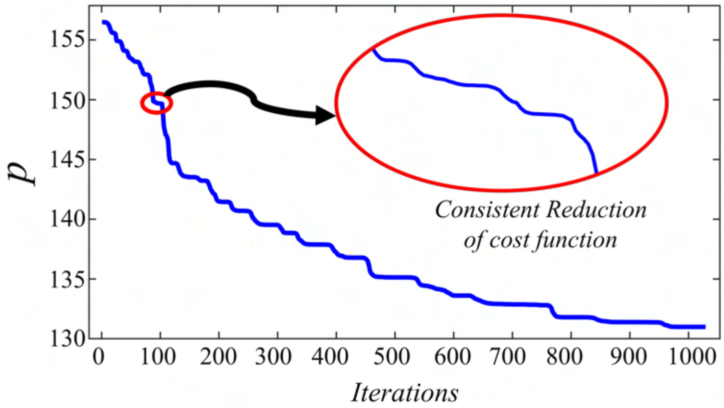
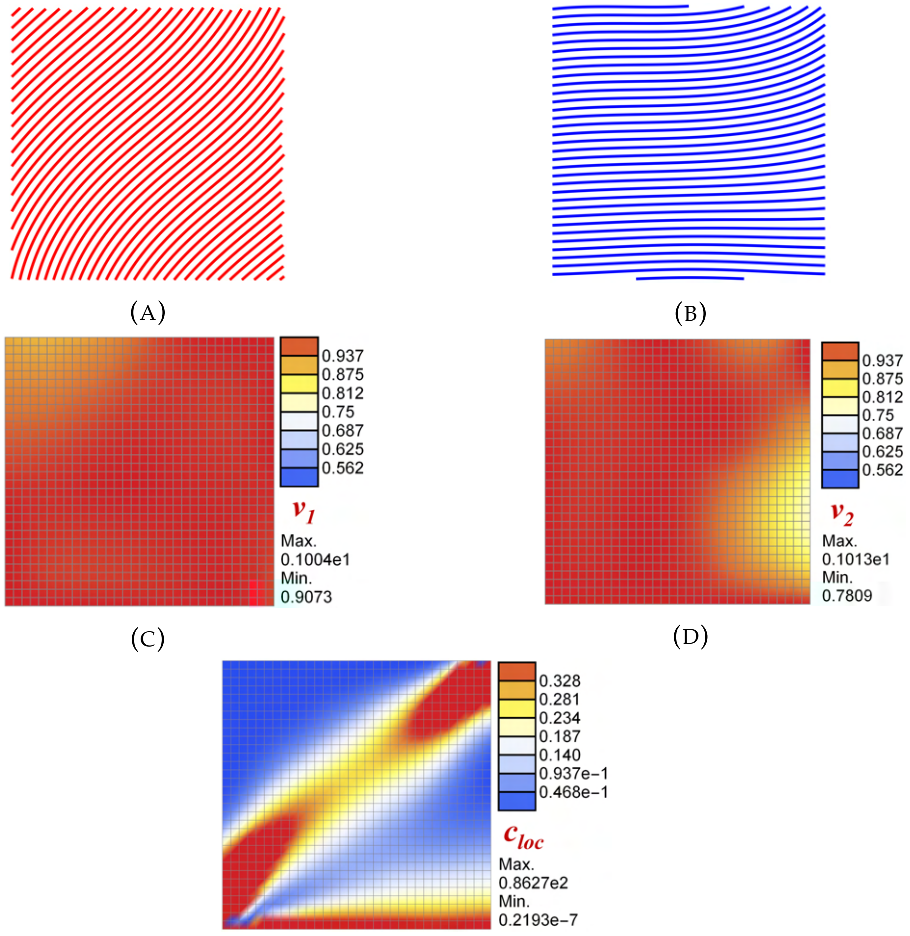


FIGURE 3.12: Convergence curve for penalized cost function in the second step of gradient-based method. The cost function consistently reduces with each iteration.

Now, the two-step gradient-based minimization is applied for optimizing a two-layer ($n = 2$) MBB structure using the 2-by-2 patch and 32-by-32 FE mesh. The initial design variables corresponding to the two layers, $\mathbf{d}_1^{\mathrm{in}}$ and $\mathbf{d}_2^{\mathrm{in}}$, are calculated from $\alpha_1^{\mathrm{in}} = 61.8°$ and $\alpha_2^{\mathrm{in}} = 14.3°$ (Eq. (3.19)), respectively.

A penalty coefficient ϵ_p=225, which is the largest value that achieves convergence, is used to penalize constraint violations. The tolerance $ag_1 = 2/ag_2 = 2$ is again prescribed and the minimization problem of Eq. (3.12) is solved.



(A) (B)

(C) (D)

(E)

FIGURE 3.13: Optimized filament paths in MBB using two-step gradient-based approach. (A) Centerlines of optimized filaments in Layer-1, (B) Centerlines of optimized filaments in Layer-2, (C) Volume-ratio in Layer-1, (D) Volume-ratio in Layer-2, (E) Plot with a local measure of compliance ($\sigma : \epsilon$) in N/mm^2.

On a PC with AMD Ryzen 5 3400G processor (3.70 GHz and 64 GB RAM), the minimization process converged in $T = 18$ minutes with a final penalized cost function of $p_{\mathrm{f}} = 130.847$. Figure 3.12 plots the convergence curve of the penalized cost function p. The final converged solution has a compliance c_{f}=130.842 Nmm and constraint quantities measuring $\epsilon_p(G_1^a)^2 = 0.002$, $\epsilon_p(G_2^a)^2 = 0.003$, $\epsilon_p(G_1^b)^2 = 0$ and $\epsilon_p(G_2^b)^2 = 0$. The results show that the no-sag constraint is trivially satisfied, since a compliance minimization problem naturally leads to a maximum-filled solution. On the other hand, the positive close-to-zero G_1^a and G_2^a indicate that the no-overlap constraint was not rigorously satisfied. The maximum overlap in the solution is observed to be $ol_{max} \approx 1\%$ of w. Figure 3.13 shows the optimized filament paths for two layers, the corresponding volume ratio plots, and the local compliance measure $c_{loc} = \boldsymbol{\sigma} : \boldsymbol{\epsilon}$ in the structure.

Differential evolution: filament path solutions

Differential evolution is applied for the filament path optimization, again using a 2-by-2 patch, a 32-by-32 mesh, and two layers ($n = 2$). For successfully applying differential evolution, the appropriate choice of control parameters were searched through manual tuning (i.e., a trial and error fashion). Table 3.1 shows four stages of applying differential evolution to the minimization problem in Eq. (3.12). In each stage, multiple trials are conducted using various combinations of cross-probability C_r, population size SP and maximum generations G_{max}, recording for each combination, the following quantities : the best penalized cost function p_b attained by the candidate solutions, the

TABLE 3.1: Tuning of control parameters in differential evolution. C_r : crossover probability; G_{max} : maximum number of generations; SP : population size of candidate solutions; p_b : best value of penalized cost function reached by candidate solutions; T : computational time; G : completed number of generations before termination

Stages	C_r	G_{max}	SP	p_b	T [min]	G
I	0.99	100	50	7.3×10^9	18	100
	0.5	100	50	1.8×10^7	20	100
	0.01	100	50	5.6×10^6	21	100
II	0.5	250	50	1.3×10^7	24	125
	0.01	250	50	3.3×10^6	59	250
III	0.01	250	80	1.1×10^6	72	250
	0.01	250	125	0.79×10^6	114	250
	0.01	250	170	4×10^6	150	250
IV	0.01	500	135	723	234	500
	0.01	1000	135	208	522	1000
	0.01	2000	135	143	960	2000
	0.01	**3000**	**135**	**136.45**	**1800**	**3000**

number of generations G completed before termination, and the corresponding computational time T. Based on the results of each trial, the control parameters C_r, SP and G_{max} are progressively tuned to reach the final set of values that fetch the optimum solution.

The entire parameter tuning process maintains a constant scale factor of $F = 0.6$ (Gämperle, Müller, and Koumoutsakos, 2002), a convergence tolerance of $ag \approx 7$ (a low tolerance avoids possible premature termination), and a penalty coefficient of ϵ_p=10^5. Here, the candidate solutions enter the infeasible design space regardless of the magnitude of penalty coefficient. However, the high cost function value assigned for the solutions with constraint violations discourages differential evolution from preferring them for subsequent generations, avoiding the convergence to infeasible design space.

Fixing the maximum generations to G_{max} = 100, and the population size to SP = 50, Stage-I applied three cross-probabilities that are representative of its entire range: C_r = 0.01, 0.5 and 0.99. On one extreme, the value C_r = 0.01 favours exploration of the design space, whereas C_r = 0.99 exploits a potential candidate solution and refines it towards the minimum point[3]. The median value C_r = 0.5 balances both the exploration and exploitation factors in equal proportion. All the three trials completed G=100 generations, yielding very high values of penalized cost function p_b that indicated severe constraint violations.

Stage-II allowed the same set of candidate solutions to evolve and explore upto maximum generation, G_{max} = 250. Only cross-probabilities C_r=0.5 and C_r=0.01 were used since $C_r = 0.99$ yielded unsatisfactory results in the previous stage. The trial with C_r=0.5 converged upon a single solution and prematurely terminated the minimization process at G = 125 generations, yielding a penalized cost function $p_b = 1.3\times10^7$. Whereas, C_r=0.01 enabled the exploration of the design space until maximum generation G_{max} was reached, attaining a significantly lower cost function p_b at a higher computation effort (T=59 minutes).

Stage-III further explored the design space using larger population sizes: SP=80, 125 and 170, while retaining G_{max} = 250 and C_r=0.01 to catalyze the design space exploration. Compared to the results from SP=50, the population size SP=80 reached a lower cost function of $p_b = 1.1\times10^6$, which was in turn surpassed by the population size SP=125 to reach $p_b = 0.79 \times10^6$. The cost function did not reduce further when the population size was increased

[3] The 0-1 scale of the C_r in Mathematica's differential evolution is reversed to the classical version mentioned in Chapter 2

to $SP = 170$. Whereas, the computational time T increased, as expected, since a larger population size needs additional FE evaluations.

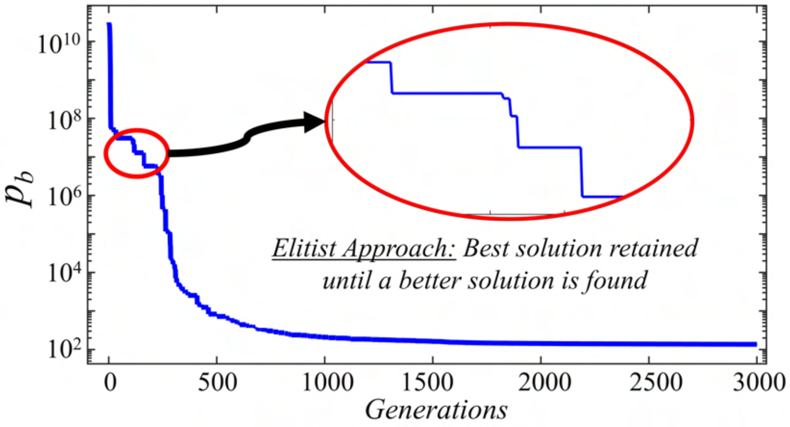
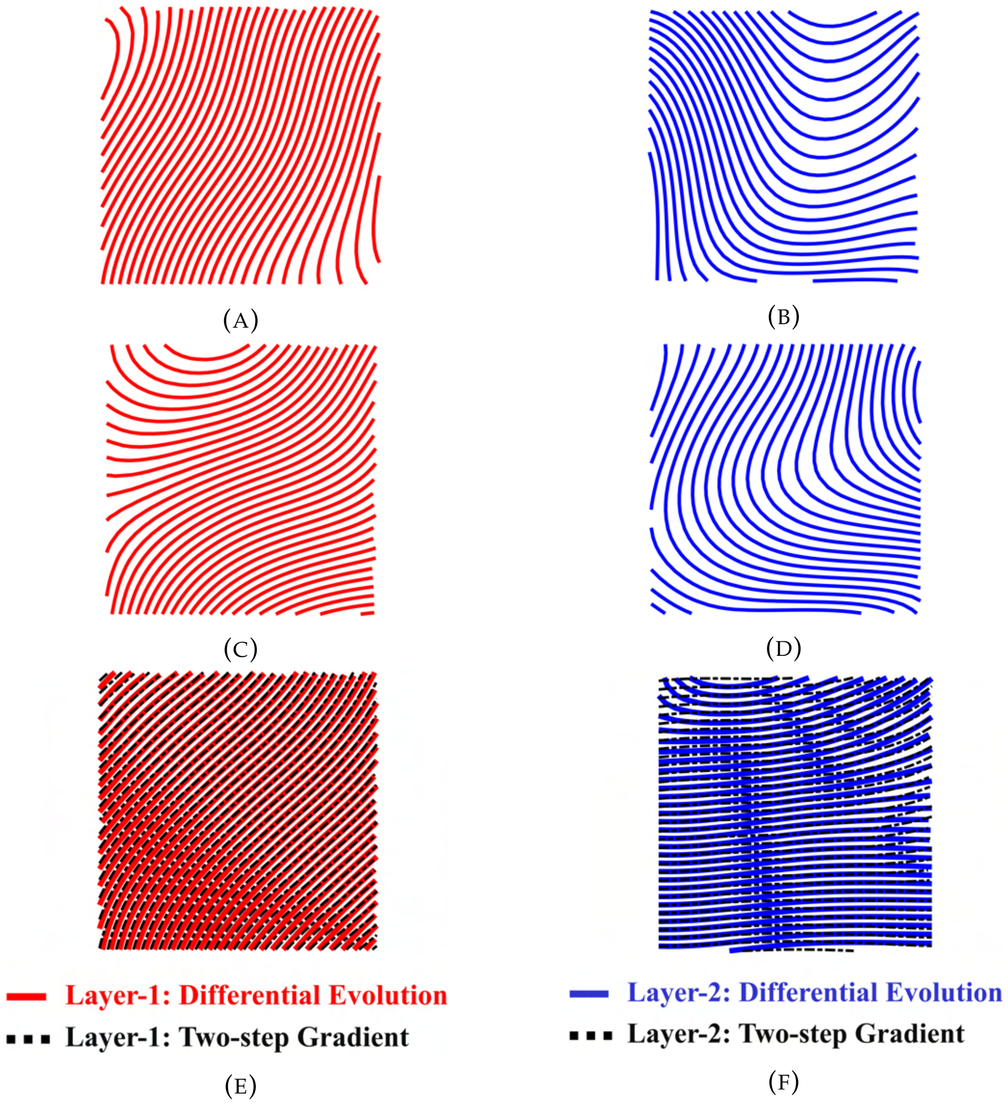


FIGURE 3.14: Decrease in the best penalized cost function p_{b} as generations increase. Solutions with best p_{b} retained for multiple generations until a better solution is found (elitist approach).

Based on the previous results, Stage IV parameters were set to $C_r = 0.01$, SP =135 and $G_{max} = 500$ to 3000. The final cost function values p_{b} dropped with an increase of maximum generations G_{max}, eventually reaching $p_{\mathrm{b}} = 136.45$ in G=3000 generations. Figure 3.14 shows the history of the best penalized cost function p_{b} against the number of generations G. The parameter tuning process was stopped here, witnessing a decreasing reduction rate in p_{b}, which indicated that a (near) minimum had been found after sufficient exploration of the design space.

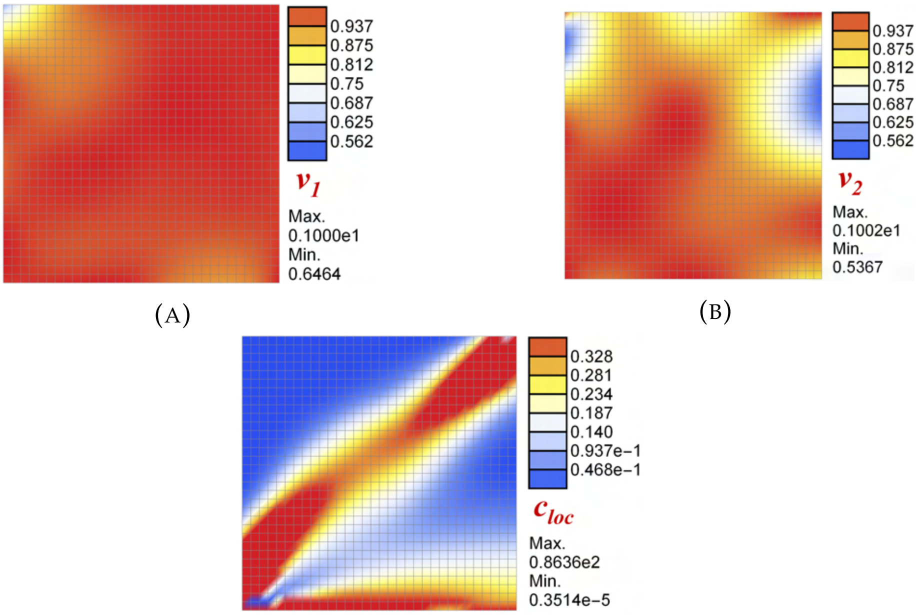
Figure 3.15 compares the the differential evolution solutions for three different set of control parameters. The filament path solution corresponding to C_r=0.01, SP=85, G_{max}=1000 (in Figures 3.15a and 3.15b), and that corresponding to C_r=0.01, SP=125, G_{max}=1000 (in Figures 3.15c and 3.15d) are entirely different from the gradient-based solutions in Figure 3.13. In contrast, the tuned set of control parameters C_r=0.01, SP=135, G_{max}=3000 yielded solutions very close to that of the gradient-based solutions (see Figures 3.15e and 3.15f). The respective best penalized cost function p_{b}=136.45 is comprised of the compliance c_{b} = 136.39 Nmm and the constraint quantities $\epsilon_p (G_1^a)^2 = 0.05$, $\epsilon_p(G_1^b)^2 = 0.01$, $\epsilon_p(G_2^a)^2 = 0$ and $\epsilon_p(G_2^b)^2 = 0$. Figure 3.16 shows the volume ratio plots, and the local compliance measure c_{loc} in the structure.



Layer-1: Differential Evolution
Layer-1: Two-step Gradient
Layer-2: Differential Evolution
Layer-2: Two-step Gradient

(A) (B) (C) (D) (E) (F)

FIGURE 3.15: Centerlines of optimized filament paths in MBB structure using differential evolution for various control parameters. (A) Layer-1, and (B) Layer-2 with $C_r = 0.01$, $SP = 85$ and $G_{max} = 1000$, yielding $p_{\text{b}} = 214$ Nmm. (C) Layer-1, and (D) Layer-2 with $C_r = 0.01$, $SP = 125$, and $G_{max} = 1000$, yielding $p_{\text{b}} = 232$ Nmm. (E) Layer-1, and (F) Layer-2 with $C_r = 0.01$, $SP = 135$ and $G_{max} = 3000$, yielding $p_{\text{b}} = 136.45$ Nmm. In (E) and (F), the centerlines of optimized filaments from differential evolution is superposed with the centerlines of optimized filaments from gradient-based method in Figure 3.13.



(A)

(B)

(C)

FIGURE 3.16: Volume ratios and local compliance measure plots corresponding to the differential evolution solution in Figures 3.15e and 3.15f. (A) Volume-ratio in Layer-1 (B) Volume-ratio in Layer-2, (C) Plot with a local measure of compliance $(\sigma : \epsilon)$ in N/mm^2.

The two-step gradient-based approach started from an informatively chosen initial point and consistently reduced the cost function at subsequent iterations to converge at a final compliance of c_f= 130.842 Nmm in $T = 18$ minutes (refer to Figure 3.12). On the other hand, differential evolution employed many candidate solutions and searched the function space for the minimum solution, retaining the best candidate solution it encountered throughout the search process (hence the staircase effect in Figure 3.14). At the end of 3000 generations completed in $T \approx 30$ hours (= 1800 minutes), the best candidate solution from differential evolution had a compliance of c_b = 136.39 Nmm. A critical comparison is next performed considering different aspects of the utilized methodologies.

(i) Efficiency: In problems where several local minima are assumed, population-based methods such as differential evolution are more likely to find the global minimum. So they may be frequently preferred over gradient based methods, despite their high computational cost. However, in the present work, the two-step gradient-based minimization could produce an acceptable solution by reaching a final compliance very near to the differential evolution $((c_b\text{-}c_f)/c_f \times 100 = 4.2\%)$ at a much lower computational cost.

(ii) *Operational Difficulties:* Although differential evolution is simple to implement, the need to tune the control parameters for each problem makes differential evolution challenging to operate.

(iii) *Influence of ν^a:* Finally, differential evolution should be avoided because it struggled with the proposed material model. Activation points ν_a larger than 0.9 often randomly terminated the differential evolution due to a severely ill-conditioned stiffness-matrix in the FE procedure. This is further discussed in section 3.4.3, where the influence of the material parameter ν^a is studied.

For the reasons mentioned above, the two-step gradient based minimization will be applied to the filament path optimization in the remainder of this chapter. However, it is emphasized that the two-step gradient-based method is still a heuristic approach that does not guarantee a global optimum for non-convex functions. Even so, the method may still be sufficient for practical application, as it has been shown to outperform a global population-based method.

3.4.2 Comparing optimized and standard straight filament patterns

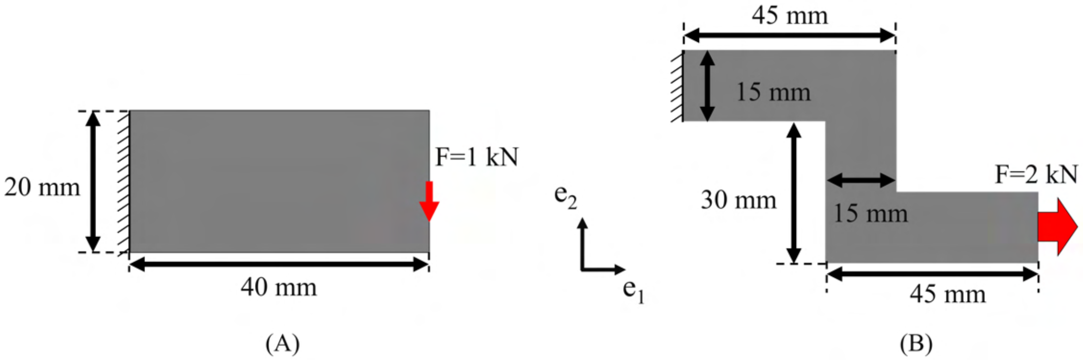


FIGURE 3.17: Example structures for filament path optimization. (A) Cantilever beam, (B) Z-shaped part

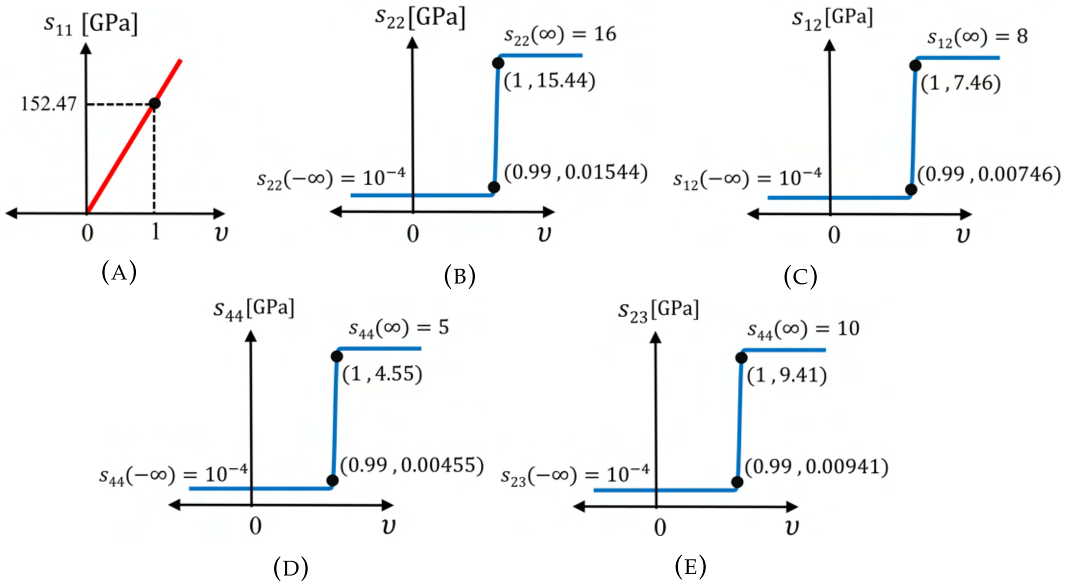
This section compares the structural compliance of optimized filament patterns against the commonly used 45°/-45° and 0°/90° straight filament patterns. A cantilever beam (Figure 3.15a), a Z-shaped part (Figure 3.15b), and the previously used MBB structure are employed for this purpose. Table 3.2 shows the different optimization parameters used in this study. Besides, the filament width (w), maximum allowed gap (l_{max}), thickness (t), two-layer

($n = 2$) assumption, convergence tolerance (ag_1, ag_2) and all the material parameters are kept from the previous section. The penalty coefficient is set to ϵ_p=10 and ν_a is set closer to one: $\nu^a = 0.99$, producing the stiffness components' evolution depicted in Figure 3.18.

The minimization problem is solved and the optimized filaments are shown in Figures 3.19 and 3.20. For clarity, the figures depict the filament paths with a constant width of w=0.6 mm. Table 3.3 reports the filament path optimization results, also facilitating a comparison between the optimized and standard straight filament patterns using ratios $c_{\mathrm{f}}/c_{\mathrm{in}}$, $c_{\mathrm{f}}/c_{45/\text{-}45}$ and $c_{\mathrm{f}}/c_{0/90}$. The ratio $c_{\mathrm{f}}/c_{\mathrm{in}}$ measures the compliance reduction due to curvilinear filaments relative to the initial straight filament design, indicating the benefit of printing curvilinear filaments. On the other hand, the ratios $c_{\mathrm{f}}/c_{45/\text{-}45}$ and $c_{\mathrm{f}}/c_{0/90}$ denote the compliance reduction by optimized filaments relative to the 45°/-45° and 0°/90° straight filaments, respectively, indicating the overall advantage of optimized filament designs over the standard filament patterns.

TABLE 3.2: Parameters for optimizing the example structures.

	MBB	**Cantilever**	**Z-Part**
Finite Elements	1024	2048	4032
Patches $\mathrm{P_x} \times \mathrm{P_y}$	$2 \times 2 = 4$	$4 \times 2 = 8$	$5 \times 3 = 15$
Control Points $\mathrm{CP_x} \times \mathrm{CP_y}$	$5 \times 5 = 25$	$7 \times 5 = 35$	$8 \times 6 = 48$
Design Variables $\mathrm{CP_x} \times \mathrm{CP_y} \times 2$	50	70	96
Initial Design $\alpha_1^{in}/\alpha_2^{in}$	61.8°/14.3°	18°/-18°	-3.79°/-61.24°



(A)

(B)

(C)

(D)

(E)

FIGURE 3.18: Evolution of the five stiffness parameters s_{11}, s_{22}, s_{12}, s_{44} and s_{23} for the assumed material properties and ν^a=0.99. Only s_{11} is significant before inter-filament fusion, the other components $s_{22}, s_{12}, s_{44}, s_{23}$ are very low for $\nu \leq \nu_a$.

TABLE 3.3: Optimized solutions and comparison to standard straight filament patterns. $\alpha_1^{in}/\alpha_2^{in}$: layer angles at starting point; c_{in} : compliance at starting point; c_{f} : final compliance; T : computational time; ν^{max} : maximum volume ratio; ν^{min} : minimum volume ratio; $c_{45/-45}$: compliance of part with two layers at 45°and -45°; $c_{0/90}$: compliance of part with two layers at 0° and 90°

		MBB	**Cantilever**	**Z-Part**
Optimized Solution	$\frac{c_{\text{f}}}{c_{\text{in}}}$	0.85	0.88	0.72
	ν_1^{max}/ν_2^{max}	1.024/1.042	1.011/1.011	1.01/1.008
	ν_1^{min}/ν_2^{min}	0.92/0.99	0.94/0.94	0.58/0.75
	T [min]	6	8	21
Optimized solutions Vs straight patterns	$\frac{c_{\text{f}}}{c_{45/\text{-}45}}$	0.45	0.25	0.25
	$\frac{c_{\text{f}}}{c_{0/90}}$	0.49	0.49	0.66

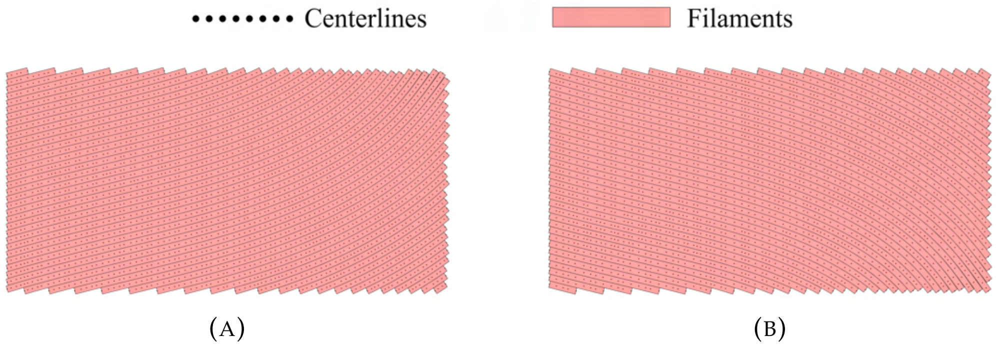


FIGURE 3.19: Optimized filament paths in cantilever. (A) Filament layout-1 (B) Filament layout-2. Filament paths depicted with a constant width of $w = 0.6$ mm

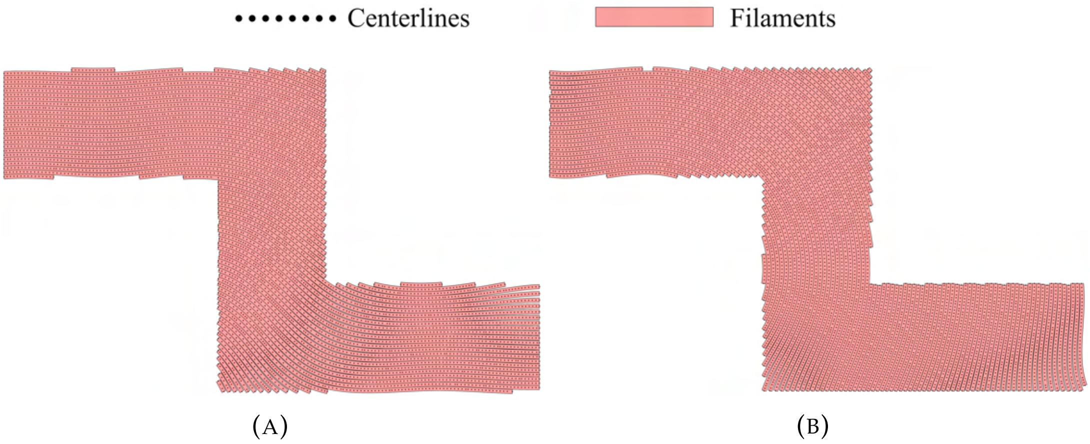


FIGURE 3.20: Optimized filament paths in Z-shaped part. (A) Filament layout-1 (B) Filament layout-2. Filament paths depicted with a constant filament width of $w = 0.6$ mm

Curvilinear filaments vary the material orientations point-wise, locally exploiting the anisotropy to tailor the mechanical properties in different directions. Their significance is better realized with complex geometries like Z-shaped part, as evident from the compliance reduction $c_{\mathrm{f}}/c_{\mathrm{in}}$ in Table 3.3. While the MBB beam and the cantilever already show 15% ($c_{\mathrm{f}}/c_{\mathrm{in}} = 0.85$) and 12% ($c_{\mathrm{f}}/c_{\mathrm{in}} = 0.88$) compliance reductions due to curvilinear filaments, the Z-shaped part reaches a higher reduction of 28%. Furthermore, the optimized filament paths offer promising benefits over the standard patterns thanks to the strong anisotropy in the assumed material. The compliance reductions relative to 45°/-45° and 0°/90° straight filament patterns are in the range of 35%-75%. On average, the compliance of the optimum solutions are at least twice as good as the compliance of the standard patterns.

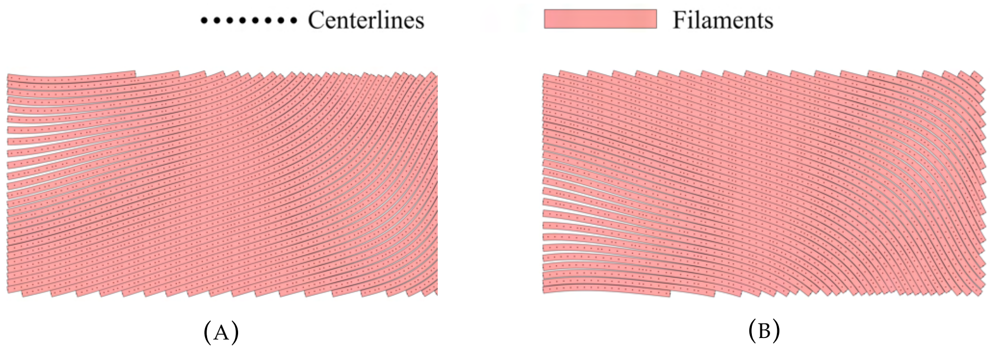
Nevertheless, the ratios $c_{\mathrm{f}}/c_{\mathrm{in}}$, $c_{\mathrm{f}}/c_{45/\text{-}45}$ and $c_{\mathrm{f}}/c_{0/90}$ offer only numerical estimations of the improvements attainable from optimized filament paths.

The exact benefits of preferring them over the straight filament patterns can be known through a more rigorous approach, in which their performances are compared after equalizing the quantity of material or the final mass in both cases (shown in Section 3.6).

3.4.3 Decoupling the filament spacing influence on printing and transverse direction stiffness

This section compares the optimized solutions from the proposed material model with the classical model of Fernandez et al., 2019 that considers homogeneously the effect of filament spacing on all stiffness components. For the sake of distinguishing both the models, the optimization involving the Fernandez et al., 2019 model is called *Linear Material Optimization (LMO)* and the one using the proposed model is called *Nonlinear Material Optimization (NMO)*, where the terms *linear* and *non-linear* indicate the corresponding transverse stiffness behaviours with filament spacing in both material models.

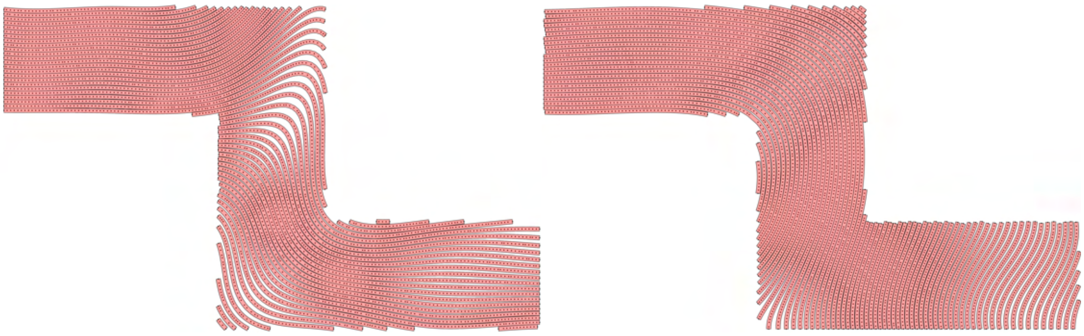
LMO is performed here, applying the same initial conditions of NMO in Section 3.4.2. Note that the initial compliance c_{in} does not change for LMO since both the classical and the new material models have the same values of stiffness components at full volume fraction (ν_i=1). Figures 3.21 and 3.22 depict the filament paths resulting from LMO. Table 3.4 reports the LMO results and compares them against the NMO results reported in Table 3.3. Next, the compliance for the LMO filament paths in Figures 3.21 and 3.22 are estimated by the proposed (nonlinear) material model. The newly evaluated compliance, called c_{nL}, are then compared against the initial compliance c_{in} and the final compliance c_{L} from LMO (refer to Table 3.4).



(A)

(B)

FIGURE 3.21: Optimized filament paths in the cantilever structure from LMO. (A) Filament layout-1 (B) Filament layout-2. Gaps appear between filaments.



(A)

(B)

FIGURE 3.22: Optimized filament paths in the Z-shaped part from LMO. (A) Filament layout-1 (B) Filament layout-2. Gaps appear between filaments.

TABLE 3.4: Optimized solutions from the LMO and comparison of compliance from the classical and the proposed material models. c_{in} : compliance at the starting point; c_{L} : final compliance from LMO; c_{f} : final compliance from NMO reported in Table 3.3; c_{nL} : compliance from the proposed nonlinear material model for the LMO filament paths in Figures 3.21 and 3.22; v^{max} : maximum volume ratio for LMO filament paths; v^{min} : minimum volume ratio for LMO filament paths

		Cantilever	**Z-Part**
Results from LMO	$\frac{c_{\mathrm{L}}}{c_{\mathrm{in}}}$	0.81	0.62
	v_1^{max}/v_2^{max}	1.05/1.05	1.048/1.036
	v_1^{min}/v_2^{min}	0.55/0.55	0.35/0.3
NMO results reproduced from Table 3.3	$\frac{c_{\mathrm{f}}}{c_{\mathrm{in}}}$	0.88	0.72
Comparison of compliance from two material models	$\frac{c_{\mathrm{nL}}}{c_{\mathrm{in}}}$	0.95	4.36
	$\frac{c_{\mathrm{nL}}}{c_{\mathrm{L}}}$	1.2	7

The optimized filament paths (Figures 3.21 and 3.22) and the numerical results (Table 3.4) from LMO are different from the NMO results in the previous section. Particularly, the LMO filament paths have larger gaps than the NMO filament paths in Figures 3.19 and 3.20. The corresponding differences in the compliance assigned by the two material models are inferred from the ratios $c_{\mathrm{nL}}/c_{\mathrm{in}}$ and $c_{\mathrm{nL}}/c_{\mathrm{L}}$. For instance, LMO yielded a compliance reduction of $c_{\mathrm{L}}/c_{\mathrm{in}}$ = 0.81 and 0.62 for the cantilever and Z-shaped part, respectively. Whereas, the proposed nonlinear material model assigned higher compliance to the LMO filament paths with $c_{\mathrm{nL}}/c_{\mathrm{in}}$ = 0.95 and 4.36.

Thus, larger inter-filament gaps and the reduced compliance result from LMO, which are due to the over-estimated transverse stiffness component at pre-fusion densities, produced, in turn, by the homogeneous linear variation of all the stiffness components with the filament spacing. In contrast, the proposed novel material model incorporates a decoupled behaviour by dropping the transverse stiffness rapidly when the filament density is below a given threshold ($\nu_i \leq \nu_a$), which is expected to be a more realistic assumption that leads to closely printed filament solutions from NMO.

The activation points ν_a play a crucial role in the effectiveness of NMO. So far, ν_a was set close to one i.e., $\nu_a \rightarrow 1$. For completeness, the influence of varying this material parameter is investigated next.

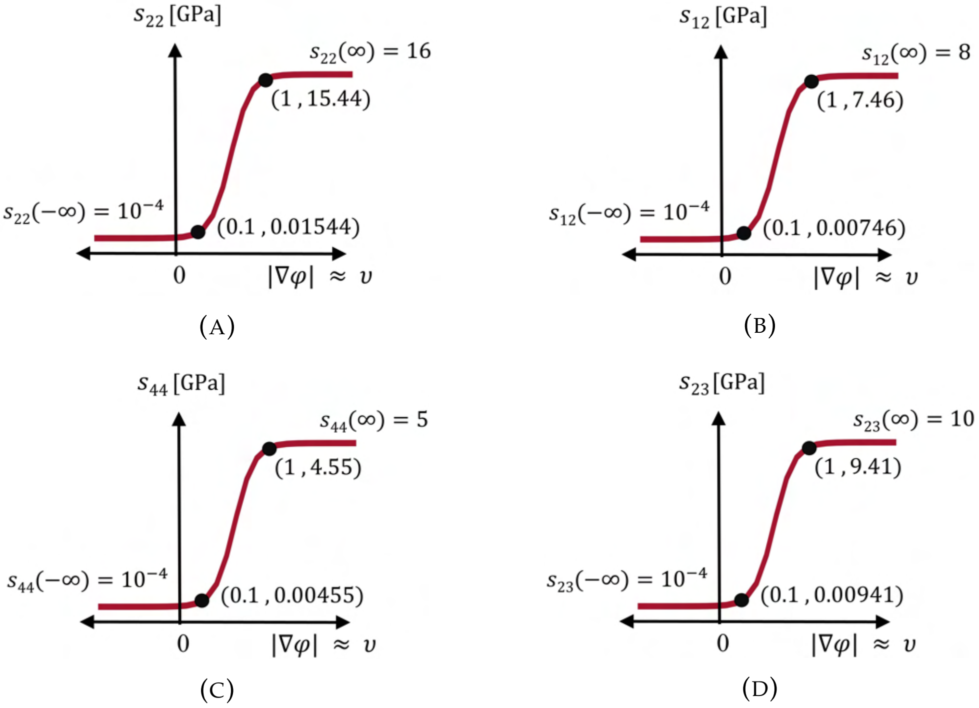


FIGURE 3.23: Evolution of the stiffness parameters s_{22}, s_{12}, s_{44} and s_{23} for the assumed material properties and ν^a=0.1. Before inter-filament fusion, all stiffness components are significant, which is physically unrealistic.

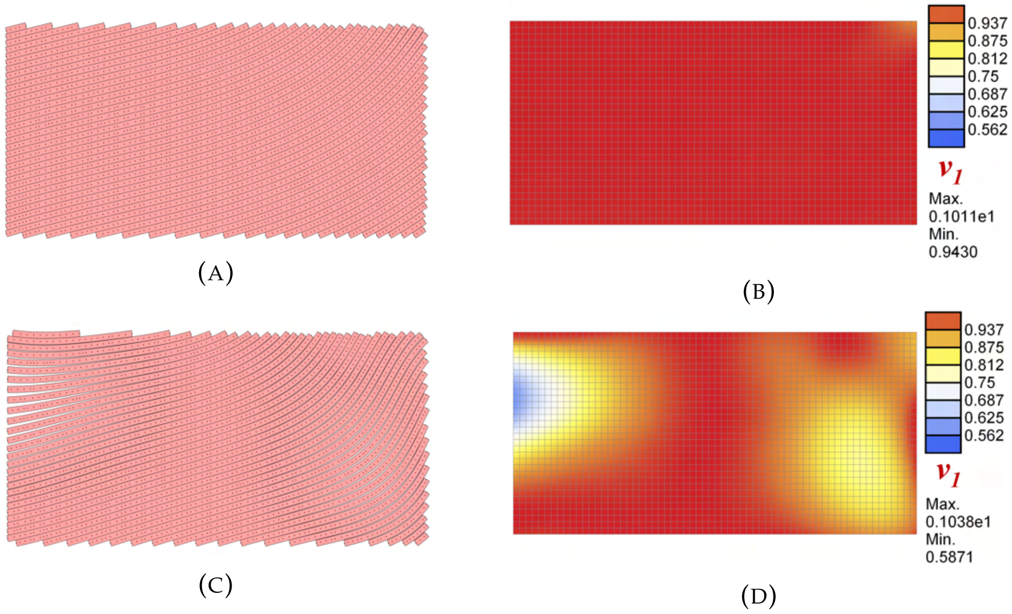
Role of the activation point ν_a

NMO is done again as in Section 3.4.2, but with a lower value of ν_a=0.1. Figure 3.23 shows the corresponding stiffness components' evolution.

The compliance reduction factors of the optimized cantilever and Z-shaped part are now $c_{\mathrm{f}}/c_{\mathrm{in}} = 0.81$ and 0.63, respectively. Figures 3.24 and 3.25 depict the optimized filament paths from $\nu_a = 0.1$, also comparing them against the solutions of $\nu_a = 0.99$. Similar to LMO, NMO with ν^a=0.1 yielded lower compliance reduction factors and larger inter-filament gaps due to the coupled variation of all the stiffness components with the filament spacing. Therefore, to avoid such solutions, the present work assumed $\nu_a \to 1$ and treated the printing and transverse stiffness differently to incorporate a physically more sound assumption.

Note that it was also observed that a higher value of ν^a was more likely to cause numerical difficulties because of not feeding the transverse stiffness of distant filaments. To illustrate this, the positive definiteness of the material stiffness matrix is studied for different volume ratios, assuming $n = 1$, $\alpha(\mathbf{x})$=45° and ν_a=0.9/0.99. The result shows that the stiffness matrix remains positive definite only for a narrow range of volume ratios (see Figure 3.26). In fact, the positive definite range is more narrow for ν_a=0.99 than for ν_a=0.9. This means that during minimization, ν_a=0.99 is more prone to reach designs with non-positive definite stiffness matrices, violating the material stability conditions, and ultimately terminating the FEA and the minimization process. The gradient based approaches can avert this issue to a certain extent by limiting the maximum step-size attempted from the initial fully-filled straight pattern (i.e., with ν=1). Differential evolution, on the other hand, does not allow for this because it searches the entire design space with numerous candidate solutions and is likely to encounter large gaps that could produce a non-positive definite stiffness matrix. For the same reason, while working with differential evolution in section 3.4.1, the value ν^a=0.9 had to be used.

Observing the significant influence of ν_a on the results, the appropriate choice of this material parameter is therefore crucial. As a part of outlook of this work, ν_a can be experimentally calibrated by conducting tensile tests along direction orthogonal to the filaments for different volume ratios.



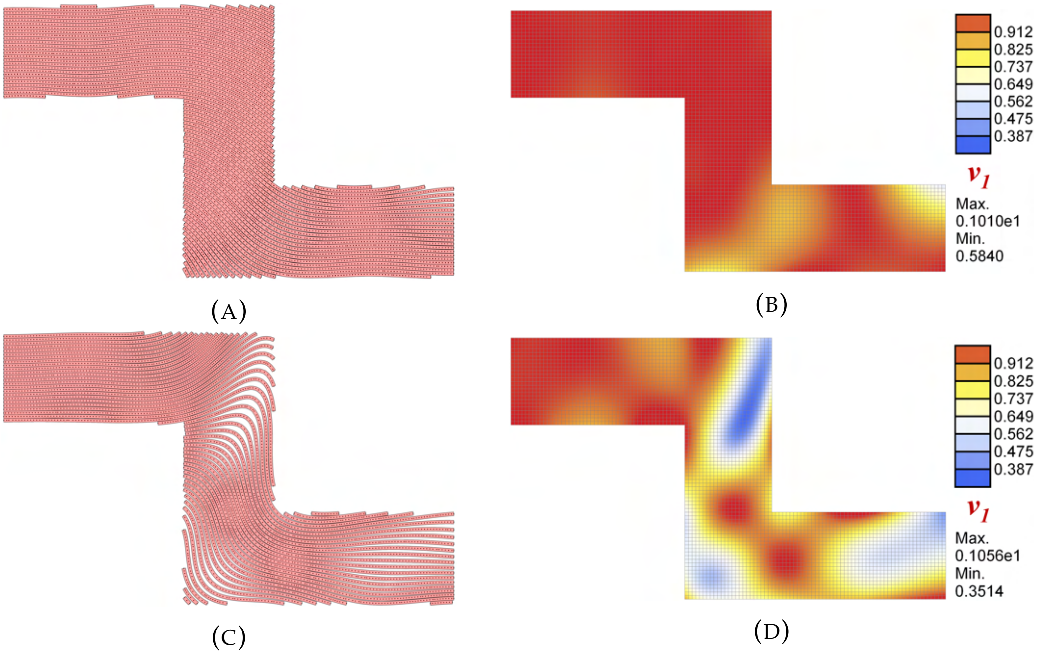
(A)

(B)

(C)

(D)

FIGURE 3.24: Influence of ν^a on Cantilever. (A) Optimized filament layout-1 with $\nu_a = 0.99$, (B) Volume-ratio plot for filament layout-1 with $\nu_a = 0.99$, (C) Optimized filament layout-1 with $\nu_a = 0.1$, (D) Volume-ratio plot for filament layout-1 with $\nu_a = 0.1$. NMO with $\nu_a = 0.1$ produces inter-filament gaps similar to LMO.



(A)

(B)

(C)

(D)

FIGURE 3.25: Influence of ν^a on the Z-Shaped Part. (A) Optimized filament layout-1 with $\nu_a = 0.99$, (B) Volume-ratio plot for filament layout-1 with $\nu_a = 0.99$, (C) Optimized filament layout-1 with $\nu_a = 0.1$, (D) Volume-ratio plot for filament layout-1 with $\nu_a = 0.1$. NMO with $\nu_a = 0.1$ produces inter-filament gaps similar to LMO.

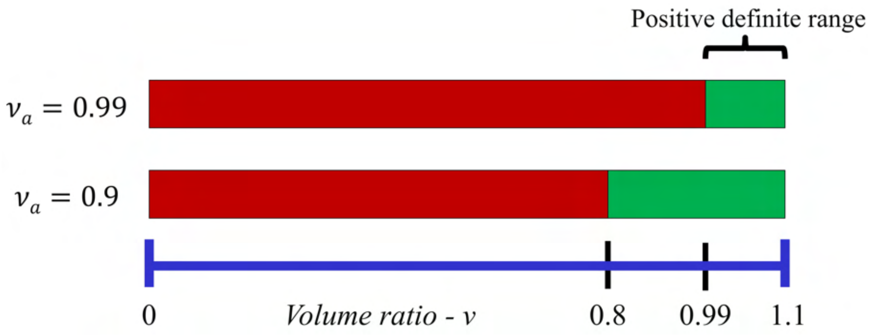


FIGURE 3.26: Positive definiteness of the stiffness matrix for different volume ratios. The red region indicates the range of volume ratios for which the stiffness matrix was not positive definite, whereas the green region indicates the positive definite range.

3.5 Parts printed with optimized filaments

As a proof of the concept detailed so far, the optimized filament paths are printed and displayed in this section. Two complex geometries are chosen: the Z-shaped part, twice the size in Figure 3.15b, and a five-hole plate. The respective dimensions and the boundary conditions are shown in Figure 3.27.

Following are the different optimization parameters used in both the examples.

1. Z-shaped part:
   P_x-by-P_y = 5×3, $N_{CP} = 48$, $\alpha_1^{in}/\alpha_2^{in}$=-3.8°/-61.9°, ϵ_p=1
2. Five-hole plate:
   P_x-by-P_y = 7×2, $N_{CP} = 50$, $\alpha_1^{in}/\alpha_2^{in}$=12.9°/-12.9°, $\epsilon_p = 10^{-4}$.

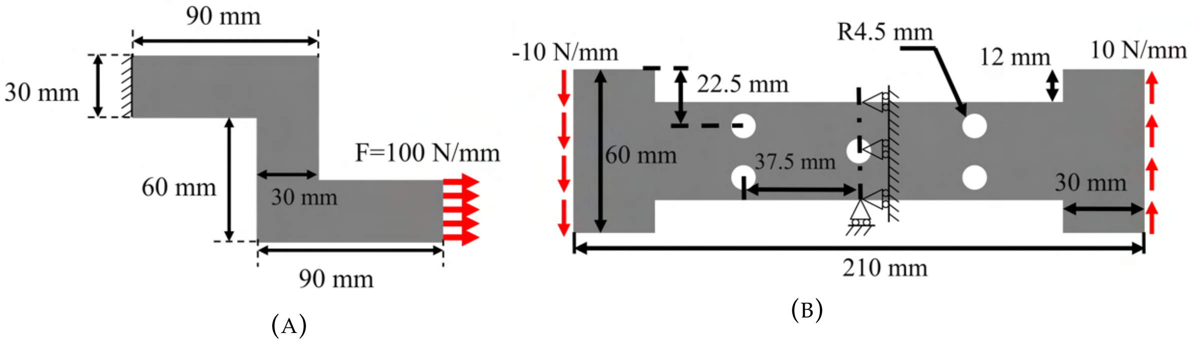
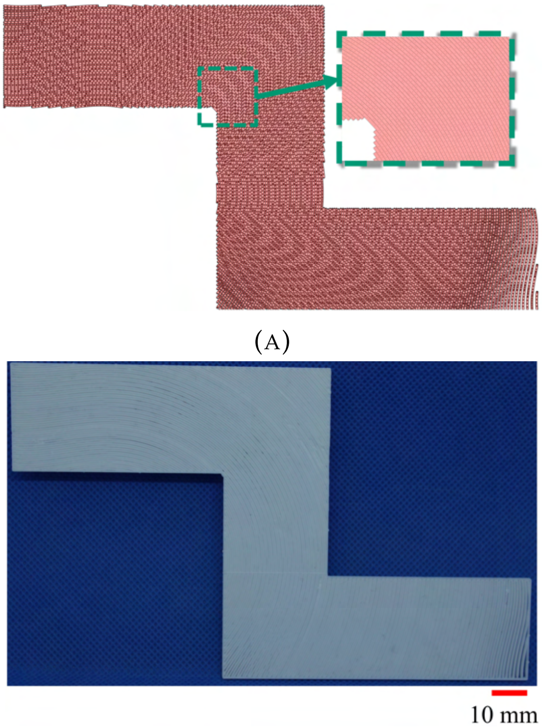


FIGURE 3.27: Example structures to be optimized and printed. (A) Z-shaped part, twice the size in Figure 3.15b, (B) Five-hole plate with shear load

For other parameters such as w, l_{max}, t, material constants and convergence criteria, the previously used values are retained here. After optimizing the filament paths, the Z-shaped part reported $c_{\mathrm{f}}/c_{\mathrm{in}} = 0.72$, $c_{\mathrm{f}}/c_{0/90} = 0.66$ and $c_{\mathrm{f}}/c_{45/\text{-}45} = 0.25$, while the five-hole plate reported $c_{\mathrm{f}}/c_{\mathrm{in}} = 0.93$, $c_{\mathrm{f}}/c_{0/90} = 0.53$ and $c_{\mathrm{f}}/c_{45/\text{-}45} = 0.2$.

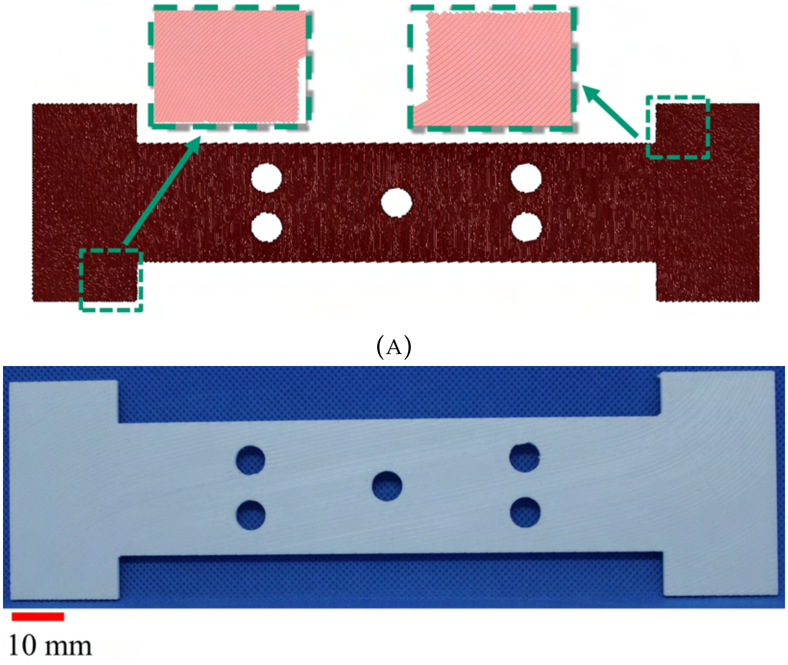
Now, the optimized filament paths are converted into suitable G-Code instructions using a Matlab algorithm and printed. Figures 3.28b and 3.29b, respectively, show the Z-shaped part and the five-hole plate printed with optimized filament paths. Each part comprises a total of eight layers, with each layer having a height h=0.2 mm. To facilitate the visual inspection of the parts, only one of the two constituent layers is printed in each example. The filament paths in the printed parts closely resemble the design solutions and preserve the optimal patterns in real time. A further perfection of the print quality is possible through an informed choice of the print parameters. Details on the printing parameters and the manufacturing aspects are provided in the next chapter, where a filament deposition algorithm is developed.



(A)

(B)

FIGURE 3.28: Printed Z-shaped part with optimized filament paths: (A) Filament path solution chosen for printing, (B) Printed part resembling the filament path solution



(A)

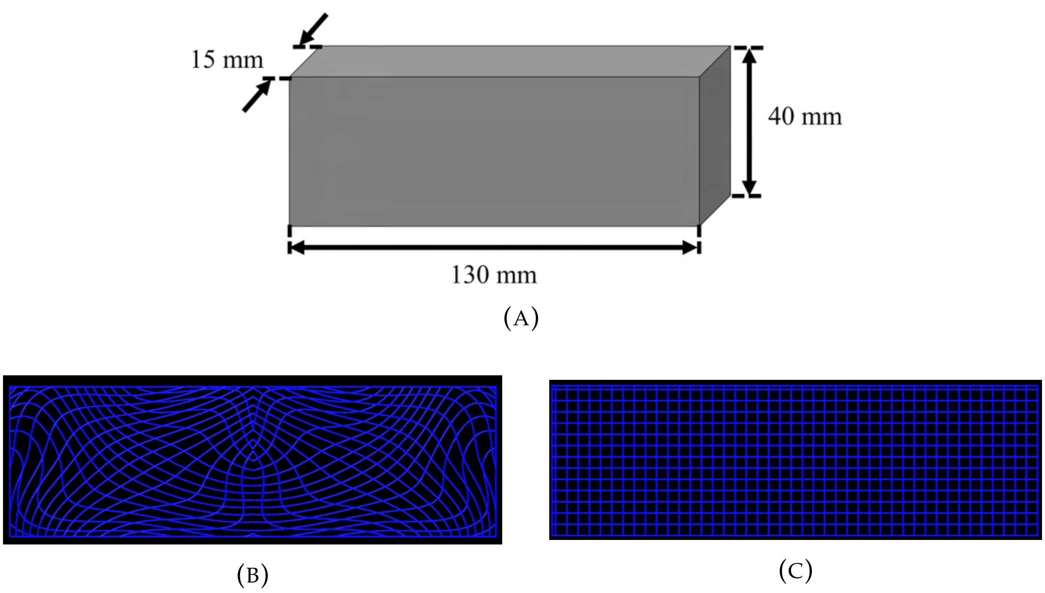
(B)

FIGURE 3.29: Printed five-hole plate with optimized filament paths: (A) Filament path solution chosen for printing, (B) Printed part resembling the filament path solution

3.6 Experimental Validation

Section 3.4.2 optimized the filament paths of strongly anisotropic FFF parts, and estimated the resulting numerical improvements to compliance. Here, the significance of optimized filament paths over standard filament patterns is verified experimentally. For this purpose, an MBB beam of dimensions shown in Figure 3.30a and printed using thermoplastic ABS is considered. First, the given MBB beam is optimized using a 2D computational model similar to Figure 3.8. The corresponding parameters including the number of patches, number of layers, penalty coefficient and filament width are reused from section 3.4.2. Since the test specimens are printed in ABS material, the stiffness components c_{mn} (at ν=1) are derived from the properties of a fully-filled 3D-printed ABS specimen: E_L=2010 MPa ; E_T=1671 MPa; G_{LT}=641 MPa ; ν_{LT}=0.32 (Alaimo et al., 2017). Note that there is little difference between the longitudinal and transverse modulus (E_L, E_T), which causes a fully-filled ABS part to exhibit an almost isotropic behaviour (or weakly anisotropic) in the linear-elastic range. Therefore, partial-filled parts are considered for this study. Accordingly, the MBB beam is optimized with an average infill of

20%, using a maximum allowable gap of l_{max} = 8 mm and a activation point of ν_a=0.9.



(A)

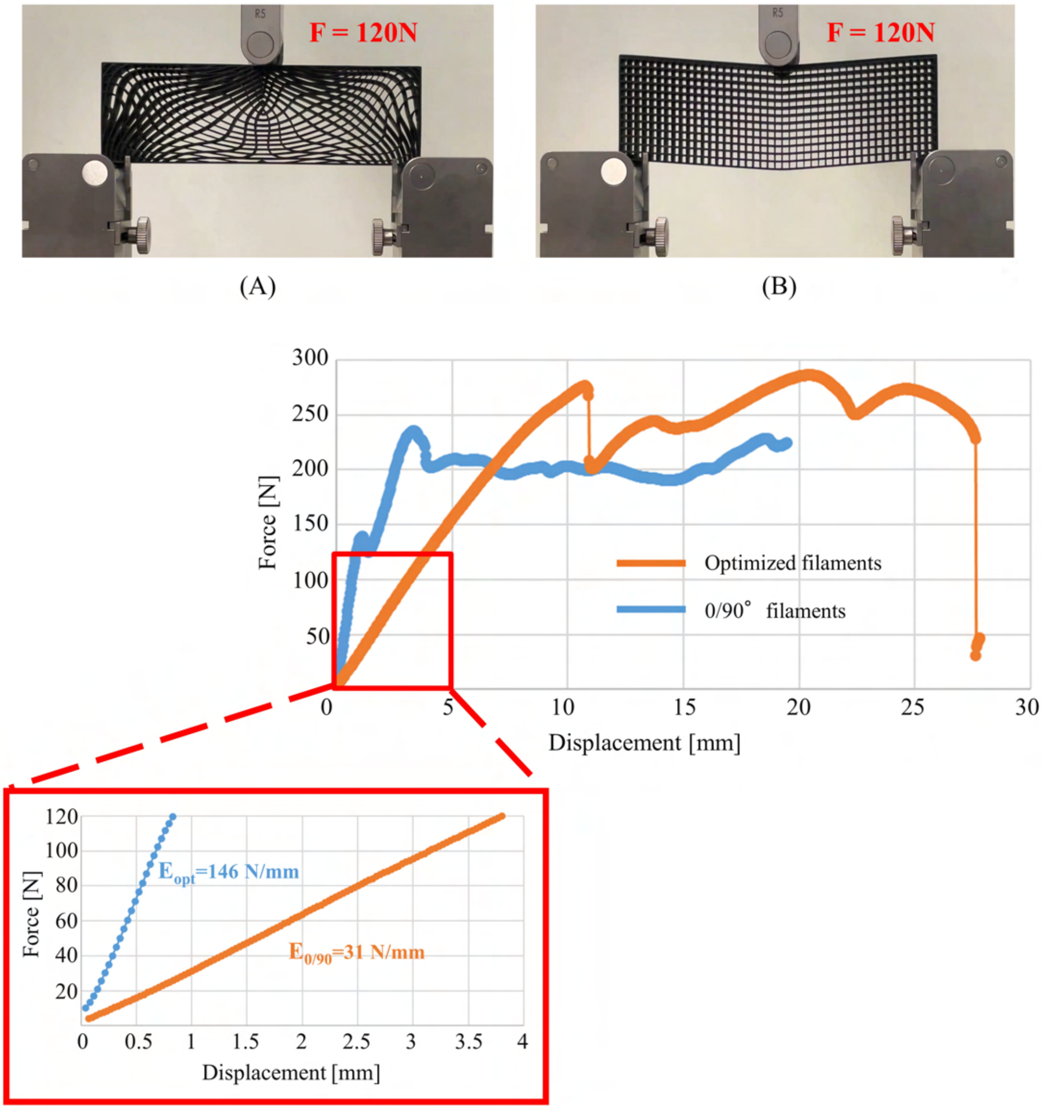
(B)

(C)

FIGURE 3.30: Test specimen for the experimental comparison of optimized and standard patterns. (A) Dimensions of the MBB beam to be tested. (B) G-code visualization for the optimized filament paths. (C) G-code visualization for the standard 0°/90° filament paths. Both the optimized and standard specimens have an average infill of 20% and weigh 18 g.

Figures 3.30b and 3.30c show the G-code paths of the MBB beam with optimized and 0°/90° patterns, respectively. The corresponding printed parts are shown in Figures 3.31a and 3.31b. Note that the inter-filament spacing of the 0/90° specimen has been calibrated to maintain the same quantity of material as the optimized specimen. Both the specimens weigh 18 g, and comprise a total of 75 layers with two constituent layers that are alternately stacked on top of each other to produce the final part.

Figure 3.31 depicts the three-point bending test on the optimized and standard specimens, along with the corresponding load-vs-displacement curves. The results show that the structural stiffness of the optimized specimen in the linear elastic region is significantly higher than that of the 0/90° specimen. While the mean stiffness of the 0/90° specimen is only $E_{0/90}$ = 31 N/mm, the optimized specimen reports a much higher value of E_{opt} = 146 N/mm (i.e., 4.7 stiffer than the 0/90° specimen), thus proving the significance of printing optimized filament patterns over standard filament patterns.



(A)

(B)

(C)

FIGURE 3.31: Three point bending test and results. (A) Loading of the optimized specimen. (B) Loading of the 0/90° specimen. (C) Load-vs-Displacement curves for the optimized and the 0/90° specimens. The optimized specimen is around 4.7 times stiffer than the 0/90° specimen in the linear elastic region.

3.7 Summary

This chapter developed an optimization framework to design production-ready in-plane filament deposition paths for minimum compliance of planar FFF structures. The framework was extensively tested for optimizing some complex two-dimensional structures, and the resulting solutions were readily converted into G-code and printed. As a proof this concept , the optimized filament patterns were also shown to outperform the classical filament patterns, both numerically and experimentally. A straightforward outlook of this work is to optimize additional parameters such as filament width, and the extension to more advanced problems like three dimensional geometries and topological optimization.

A notable highlight of this contribution is the new material model that incorporates a decoupled variation of the printing and transverse stiffness with the filament spacing. The newly proposed material model produced more realistic designs, having fewer inter-filament gaps than the model that coupled the influence of filament spacing on all the stiffness components. Furthermore, the comparative study of the minimization approaches showed that the two-step gradient-based approach achieved final compliance closer to differential evolution at a lower computational cost, proving to be more efficient and preferable for this type of problem. Another reason for employing the gradient-based approach is the appearance of numerical issues in differential evolution when the stiffness components are activated close to inter-filament fusion (i.e., $\nu_a \to 1$). Due to the crucial role played by the activation point ν_a in the new material model, the experimental calibration of ν_a is an additional outlook of this work.

Chapter 4

An Orientation-Based Filament Deposition Algorithm For Point-Wise Optimization Strategies

Chapter 3 developed an optimization framework to design readily printable filament paths that minimize the compliance of FFF parts. In this chapter, the filament paths are not optimized, but the manufacturability issues of the commonly used optimization strategies are addressed.

Point-wise optimization strategies (refer to Sections 1.4 and 2.2) parametrize the material orientations of a FE discretized structure and result in optimal orientation fields that cannot be printed using conventional slicing software. So far, only approximated techniques have been devised to deposit filaments along optimal orientation fields. For example, Hoglund and Smith, 2016 and Catapano et al., 2019 printed filaments along the given orientation fields by approximating the local orientations to the closest printing pattern in the slicing software. However, these techniques are cumbersome, time-consuming and are not guaranteed to succeed. To address this limitation, the present chapter proposes a new orientation-based filament deposition algorithm[1]. The algorithm takes as input an arbitrary discrete orientation field supplied by, for example, an optimization process or the principal stress directions, and directly converts it into manufacturable material deposition paths without introducing any approximation.

In literature, only a limited number of contributions addressed the algorithmic shortcomings in printing filaments along an orientation field. The first relevant work is by Tam and Mueller, 2017, who proposed Stress Line Additive Manufacturing (SLAM) method to generate filaments along the

[1] This chapter is inspired from the paper published in the Additive Manufacturing journal. DOI:10.1016/j.addma.2022.103064

principal stress directions. Khan et al., 2018 used an analytical method to design curvilinear filament paths for an open-hole plate subjected to a unidirectional tensile loading. Curvilinear Scaffolds were designed by Roberge and Norato, 2018 using a level-set approach for manufacturing by Direct Ink Writing (DIW), which is also a process belonging to the Material Extrusion family. Finally, a recent work by Kubalak, Wicks, and Williams, 2019a proposed Volumetric Deposition Path Generation (VDPG) that relies upon the streamline placement algorithms to generate the deposition paths from an arbitrary orientation field.

Most of the methods presented above do not allow a direct translation of the optimal orientation and the spacing fields into a G-code, thus preventing the possibility for customization. In contrast, the proposed algorithm has been implemented to be flexible and can accept changes in a multitude of parameters including filament spacing and extrusion width to support easy customization of the designed products.

This chapter is organized as follows:
Section 4.1 details the various steps involved in the proposed filament deposition algorithm. Section 4.2 assumes three example structures and employs the proposed algorithm to generate filaments along the principal stress directions of the structures. The results yielded by the algorithm and the final printed parts are presented for both partial and maximum infills. Section 4.3 provides a summary of this chapter and briefly describes the further directions of this work.

4.1 Material and Methods

The proposed algorithm aims at producing filament paths with a point-wise orientation dictated by an orientation field. In the usual slicing process, a CAD model is sliced into layers of predefined filament patterns and converted into G-code instructions. Here, instead of an input CAD model, the following quantities are used: a set of 2D positions denoting the geometrical bounds of the part to be printed, an orientation field that informs the material orientation at a discrete set of points in the infill domain (i.e., the domain to be filled with material), and a set of 2D positions used as seed points (explained in the next section. Additionally, the algorithm can consider other relevant parameters for designing the filaments' trajectories and dimensions: extrusion width w, layer height h and the minimum spacing required between filaments G_p, the number of layers n, the printing speed of the nozzle

V_p, the travel speed of the nozzle V_t, filament diameter D_ϕ and the retraction length R_L (filament length to be retracted at the end of each printing ; see Figure 4.1).

The whole algorithmic framework is developed in MATLAB R2019b (refer to Appendix C). Figure 4.2 shows the algorithm workflow consisting of three main steps: (I) streamline generation, (II) density control and, (III) path linking and G-Code generation. All these steps will be discussed in the following sections.

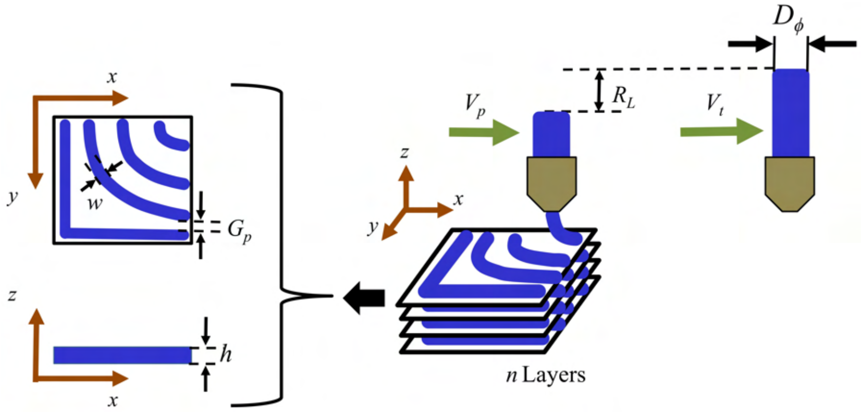


FIGURE 4.1: FFF printing parameters. w : extrusion width of each filament ; h : height of each filament/layer ; G_p : prescribed minimum distance between any two filaments ; V_t : speed of nozzle when travelling ; V_p : speed of the nozzle when printing ; R_L : length of the filament to be retracted at the end of each extrusion ; n : number of layers.

4.1.1 Streamline generation

The concept of streamlines is applied for generating filament trajectories. In fluid flow analysis, a streamline refers to the trajectory of a single particle that depicts the particle's flow over a fluid domain (McLoughlin et al., 2010). In case the fluid domain has a stationary velocity field $\mathbf{v}(\mathbf{x})$, it is expressed as the time-derivative of the particle position $\mathbf{x}$, i.e.,

$$\mathbf{v}(\mathbf{x}) = \frac{\mathrm{d}\mathbf{x}}{\mathrm{dt}}. \tag{4.1}$$

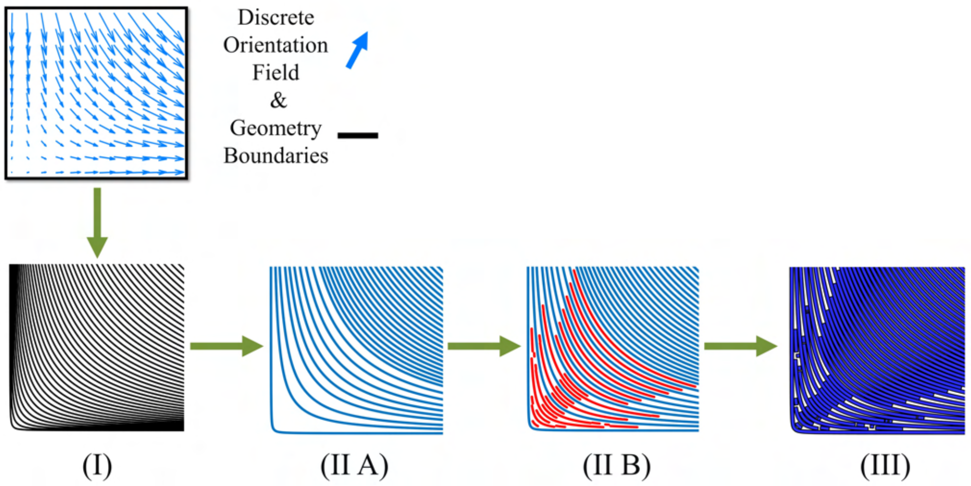
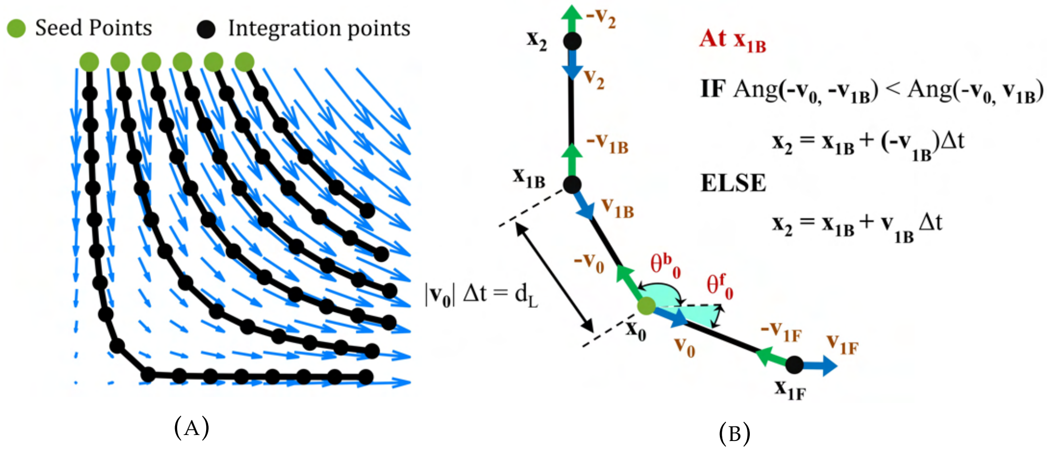


FIGURE 4.2: Working principle of filament deposition algorithm. (I) Streamline generation receives the input discrete orientation field and geometry boundaries to generate streamline paths. (II) Density control (A) removes streamlines that don't satisfy the spacing constraints and generates continuous filament trajectories. (B) - fills the residual gaps with discontinuous filament trajectories. (III) Path linking and G-Code generation links individual trajectories and creates machine instructions.



(A)

(B)

FIGURE 4.3: (A) An example showing streamlines generated over a fluid domain. (B) Application of the streamline concept to orientation field. From each seed point, the integration is done in both forward and backward directions. To find the next integration point, the algorithm chooses among the two possible velocities, the one with the least orientation to the velocity chosen at the previous integration point. d_L is the step length and Ang($\mathbf{v_m}$, $\mathbf{v_n}$) gives the angle between two vectors $\mathbf{v_m}$ and $\mathbf{v_n}$.

To generate a streamline, a seed point (i.e., the initial position of the fluid particle) is placed in the fluid domain and Eq. (4.1) is numerically integrated.

Applying, for example, the Euler's forward integration method, the following expression is yielded:

$$\mathbf{x_{j+1}} = \mathbf{x_j} + \mathbf{v}(\mathbf{x_j})\Delta t, \tag{4.2}$$

where $\mathbf{x_j}$ and $\mathbf{x_{j+1}}$ represent the two consecutive positions of the particle at times t_j and t_{j+1}, respectively. The quantity Δt= t_{j+1}-t_j denotes the time step, and $\mathbf{v}(\mathbf{x_j})$ is the velocity at the point $\mathbf{x_j}$. By repeating this process, a trajectory of the particle's positions is obtained over the fluid domain, as illustrated in Figure 4.3a.

The above process can also be applied for tracing an orientation field. Let $\{\theta_{k=1,2,..N}\}$ be the set of orientations prescribed at N points in the infill domain. They are expressed as normalized velocity vectors such as $\mathbf{v}_k = [\cos(\theta_k), \sin(\theta_k)]^T$, which are further interpolated to make available the orientation information at every point $\mathbf{x}$ belonging to the infill domain (for interpolation, the built-in Matlab function *scatteredInterpolant* is used). The resulting interpolated velocity function $\mathbf{v}(\mathbf{x})$ is applied to Eq. (4.2), and the streamlines are generated from each seed point placed in the infill domain. For a thorough coverage of the domain by the streamlines, the seed points are scattered throughout the infill domain, and a bidirectional sense is imposed for the velocities. As a result, the numerical integration is done in the forward and the backward directions, creating streamline points in both directions from a seed point. An example is provided below using Figure 4.3b.

Let $\mathbf{x_0}$ be a seed point having velocities $\mathbf{v_0}$ and -$\mathbf{v_0}$ (bidirectional sense). Applying Eq. (4.2), two new points are computed using both the velocities as given below:

$$\mathbf{x_{1F}} = \mathbf{x_0} + \mathbf{v_0}\Delta t, \tag{4.3}$$

$$\mathbf{x_{1B}} = \mathbf{x_0} + (-\mathbf{v_0})\Delta t, \tag{4.4}$$

where $\mathbf{x_{1F}}$ and $\mathbf{x_{1B}}$ are the points computed using the velocities $\mathbf{v_0}$ and -$\mathbf{v_0}$, respectively (F and B denote the forward and backward directions). Now, the integration is separately carried out from the points $\mathbf{x_{1F}}$ and $\mathbf{x_{1B}}$ using their corresponding velocities (i.e., using $\mathbf{v_{1F}}$/-$\mathbf{v_{1F}}$ and $\mathbf{v_{1B}}$/-$\mathbf{v_{1B}}$). To compute a new point at each step, the algorithm will use the velocity that subtends the least angle with the velocity chosen at the previous step. For instance, $\mathbf{x_2}$ is found from $\mathbf{x_{1B}}$ in the following fashion:

```
At $\mathbf{x_{1B}}$,
    IF $\text{Ang}(-\mathbf{v_0}, -\mathbf{v_{1B}}) \leq \text{Ang}(-\mathbf{v_0}, \mathbf{v_{1B}})$,
        $\mathbf{x_2} = \mathbf{x_{1B}} + (-\mathbf{v_{1B}})\,\Delta t.$
    ELSE,
        $\mathbf{x_2} = \mathbf{x_{1B}} + \mathbf{v_{1B}}\,\Delta t,$
```

where the function Ang($\mathbf{v_m}$, $\mathbf{v_n}$) returns the angle between any two vectors $\mathbf{v_m}$ and $\mathbf{v_n}$. Among the two equal and opposite velocities at $\mathbf{x_{1B}}$, $-\mathbf{v_{1B}}$ subtends the least angle to the velocity $-\mathbf{v}_0$ chosen at the previous step (refer to Figure 4.3b). Therefore, $\mathbf{x_2}$ is found using the expression $\mathbf{x_2} = \mathbf{x_{1B}} + (-\mathbf{v_{1B}})\Delta t$. The integration process continues in both the directions until the streamline terminates at a design boundary where a zero velocity has been set.

The distance between any two integration points $\mathbf{x_j}$ and $\mathbf{x_{j+1}}$ is the step length $\mathrm{d_L}$ that is defined as, $\mathrm{d_L} = |\mathbf{v_j}|.\Delta t$. The step length affects both the quality of the filament trajectories and the efficiency of the algorithm. The smaller $\mathrm{d_L}$, the smoother and the more accurate would be the filament trajectories, while the higher would be the computational effort of the algorithm. FFF printer in-plane resolution is also an additional factor that dictates the choice of the step length $\mathrm{d_L}$.

4.1.2 Density control

Streamlines generated from the seed points scattered throughout the infill domain are likely to cover the entire domain, neglecting any physical spacing between them. However, filaments are physical entities and they require a minimum distance to be prescribed between their corresponding centerlines. Therefore, this section of the algorithm checks the spacing between the streamlines and iteratively removes those that violate the spacing requirements. For this purpose, the method of Delaunay triangulation (Mebarki, Alliez, and Devillers, 2005) is used. The integration points of the streamlines are subjected to a Delaunay Triangulation, which yields a mesh of triangles, the vertices of which are made of streamline integration points (see Figure 4.4). Each triangle in the mesh can be associated with a unique circle called the circumcircle that passes through its three vertices and its diameter can closely approximate the local distance between adjacent streamlines. Let D_{ij} represent the diameter of the circumcircle of a triangle, the vertex of which is the j^{th} integration point from the i^{th} streamline. If D_{ij} is larger than or equal to the sum of extrusion width w and spacing G_p, then it means that the spacing requirement between the streamline i and its neighbouring streamline is locally satisfied. To check if the entire streamline i is satisfying the spacing requirements with the adjacent streamlines, the following condition needs to be ensured at its all integration points.

$$D_{ij} \geq w + G_p \quad \forall j \in \mathbb{I} \tag{4.5}$$

where $\mathbb{I}$ denotes the set of all integration points belonging to streamline i. If the above condition is not satisfied even at a single point ($\exists j \in \mathbb{I}$ such that $D_{ij} < w + G_p$), then the adjacent streamline connected to the i^{th} streamline through the triangulation will be removed from the data set. An example is provided in Figure 4.4 for gaining clarity.

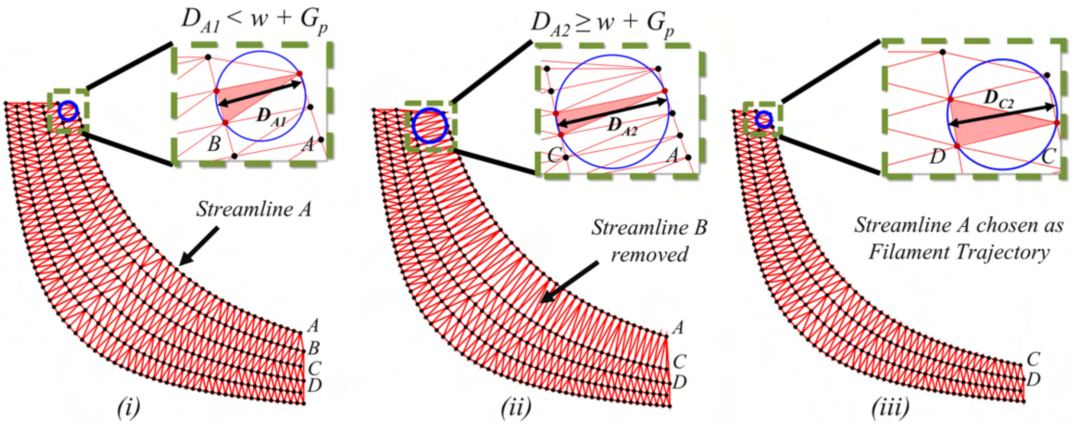


FIGURE 4.4: Illustration of density control in filaments.

In Step (i) of Figure 4.4, all the streamlines are subjected to Delaunay Triangulation. First, the spacing around the streamline A is checked using the diameters of the circumcircles. In this example, the condition in Eq. (4.5) was found not to be satisfied at the first integration point (j=1) belonging to the streamline A (i.e., $D_{A1} < w + G_p$), implying the violation of spacing requirement between the streamlines A and its neighbouring streamline B. Accordingly, the streamline B is completely removed from the data set and the spacing around streamline A is again checked (Step (ii) of Figure 4.4). Now, the diameters of the circumcircles corresponding to all the integration points of streamline A were found to satisfy the condition in Eq. (4.5), indicating the fulfilment of the spacing requirements between the streamline A and currently adjacent streamline C. Then, the streamline A is removed from the triangulation data set only to be retained as a chosen filament trajectory. As the next step, the spacing around Streamline C is checked using the same procedure described so far (Step (iii) of Figure 4.4). This process is repeated until the spacing of all the streamlines are checked and those that violate the spacing requirements are completely removed.

The above explained procedure is computationally expensive compared to the strategy of placing streamlines at a preset distance and removing only the violating portions. However, the latter can potentially lead to more discontinuities that ultimately reduces the stiffness of the final part. Aiming to avoid this, the proposed algorithm first generates streamlines from seed points scattered throughout the infill domain and then retains only the continuous filament trajectories that satisfy the required spacing conditions.

Filling gaps using discontinuous filaments

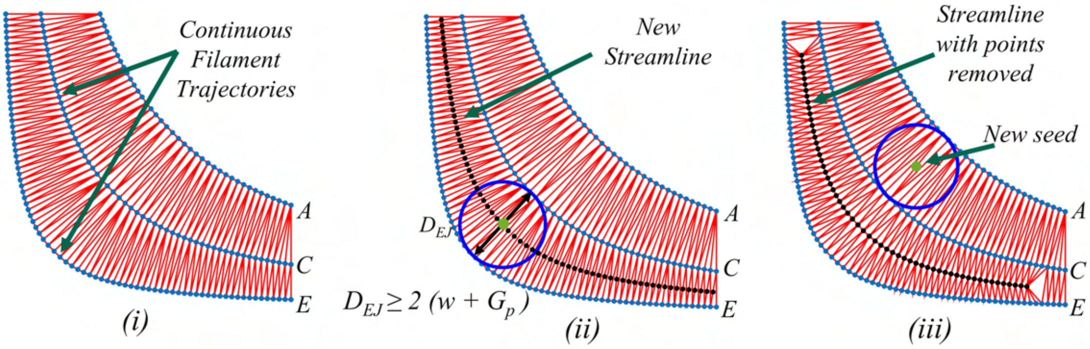
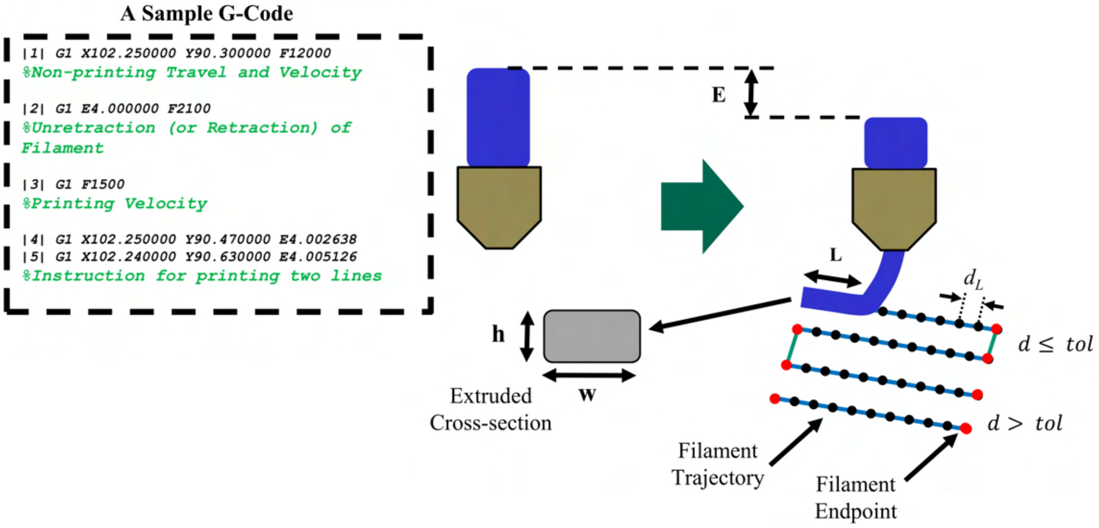

Continuous Filament Trajectories
A
C
E
(i)
New Streamline
D_{EJ}
$D_{EJ} \geq 2\ (w + G_p)$
A
C
E
(ii)
Streamline with points removed
New seed
A
C
E
(iii)

FIGURE 4.5: Illustration of the generation of discontinuous filament trajectories.

The aforementioned process results in a set of continuous filament trajectories with the required spacing. However, it does not attain the maximum possible filament area coverage due to the residual gaps existing in between the continuous curvilinear filaments. Hence, when maximum filament area coverage is desired, a further procedure is introduced to fill the gaps using discontinuous filaments. The continuous filament trajectories yielded by the previous steps are subjected to Delaunay Triangulation (Step (i) of Figure 4.5). Then, all the circumcircles are evaluated according to the following condition:

$$D_{ij} \geq 2(w + G_p) \quad \forall j \in \mathbb{I},\ \forall i \in \mathbb{S} \tag{4.6}$$

where $\mathbb{S}$ is the set of pre-existing filament paths. The condition in Eq. (4.6) enables the identification of all gaps that can hold a filament of width w while respecting the spacing constraints with the neighbouring filaments. The largest circle that satisfies this condition is identified and a seed point is placed at the center of this circle to generate a new streamline that fills the gap (Step (ii) of Figure 4.5). Then, the triangulation is updated with the newly formed streamline and the spacing constraints with the pre-existing filament trajectories are checked following the procedure described in section 4.1.2. Finally, the points that violate the constraints are removed from the data structure leaving a discontinuous filament trajectory (Step (iii) of Figure 4.5). The above series of steps are repeated for the next largest circle and are consecutively iterated until there are no longer gaps to contain a filament of width w.



A Sample G-Code

```
|1| G1 X102.250000 Y90.300000 F12000
%Non-printing Travel and Velocity

|2| G1 E4.000000 F2100
%Unretraction (or Retraction) of
Filament

|3| G1 F1500
%Printing Velocity

|4| G1 X102.250000 Y90.470000 E4.002638
|5| G1 X102.240000 Y90.630000 E4.005126
%Instruction for printing two lines
```

FIGURE 4.6: Linking of paths and G-Code generation. X and Y values in the sample G-code denote the coordinates of the filament trajectories. E is the length of the feedstock filament pushed in the heating chamber to extrude a rectangular cross-section through a length L. If two filament endpoints have a distance greater than a given distance tol, the nozzle retracts the filament, moves to the new point and then extrudes the filament.

4.1.3 Path linking and G-Code generation

The printing instructions for a single layer are defined by sequentially linking all the filament trajectories in the layer. This step is done aiming at reducing the total non-printing travel movements. During the printing process, the nozzle starts to print from one of the end points of a single filament trajectory and proceeds towards the other end, extruding molten material and then, it moves to the closest filament end point (Figure 4.6). If the two filament end points are farther than a certain distance (tol), then the printing filament has to be retracted during the travel movement in order to prevent the oozing of material. The length of the filament to be retracted can be mentioned in the input parameter R_L.

The printing of filaments as per design specifications strongly depends upon suitable G-Code instructions. A G-Code instruction to print a filament has two parts: X-Y positions and the flow rate E. While the X and Y values instruct the printing head to move to a specific position, E specifies the length of the raw filament that is pushed into the heating chamber to extrude a filament of width w and a height h. The X and Y values are obtained from the previous steps, whereas E is calculated based on the extruded filament's length and cross-section. With reference to Figure 7, the flow rate E is calculated as:

$$E = \frac{C_m V_e}{A_\phi} = \frac{0.99(whL)}{\frac{\pi}{4} D_\phi^2} \tag{4.7}$$

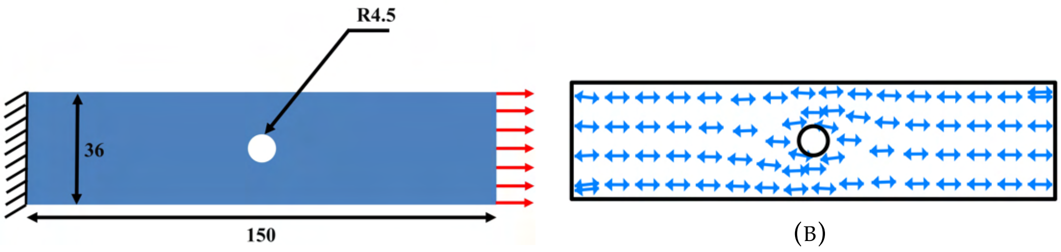
where, $V_e = w\ h\ L$ is the volume of material extruded while printing a path of length L. A rectangular shape of dimensions $w \times h$ is assumed for the extruded cross-section in order to achieve a higher adhesion in between the filaments (Koch, Van Hulle, and Rudolph, 2017). The multiplication factor $C_m = 0.99$ (adopted from commercial slicing software (KISSlicer, 2022)) is used to account for the discrepancy between the assumed perfect rectangular cross-section and the actual shape of the extruded cross-section. Finally, $A_\phi = \pi/4\ D_\phi^2$ is the area of cross-section for the feedstock filament of diameter D_ϕ. Details on the calculation of flow rate can be found in Hodgson, Ranellucci, and Moe, 2015.

4.2 Results

Three structural problems, namely case A, B, and C, are presented to validate the algorithm. The corresponding geometries and the boundary conditions are shown in Figures 4.7a, 4.8a, and 4.9a. The inputs to the algorithm are obtained through the Finite Element Analysis (FEA) of the structural problems, applying a linear elastic isotropic constitutive model under plane stress and small strain assumptions (for FEA, the commercial software Ansys is used). Figures 4.7b, 4.8b, and 4.9b show the first principal stress directions of the structures that specify the input orientation field. The boundary nodes of the meshed geometries set the bounds of the infill domains, while the centroids of elements in FE meshes provide the seed point locations. Besides, the values of other input parameters are reported in Table 4.1.

The given geometries are printed using the new filament deposition algorithm, and the results are presented in four sections: *(a) Infill Control*, *(b) Multi-Oriented Stacking*, *(c) Experimental Measure of Inter-filament Spacing*, and the *(d) Seed Point Influence*. For printing, a 3ntr A4v3 machine is used. It runs a modified MK4duo firmware and has a build volume of 295 x 295 x 200 mm with a heating chamber. The extruders can reach a maximum temperature of 390°C, and the build plate can reach up to 120°C.

Case A : Tensile load on an open-hole plate

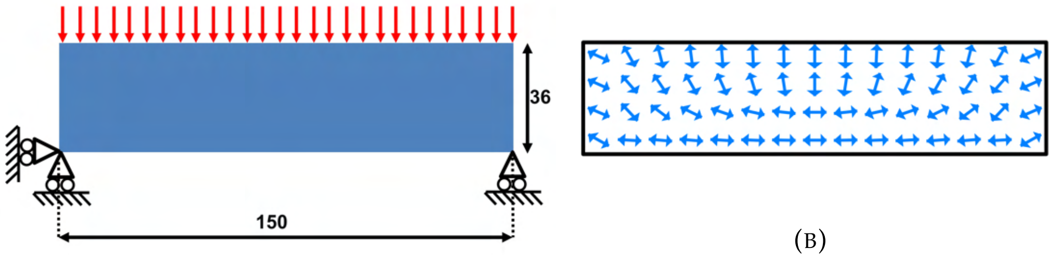


(A)

(B)

FIGURE 4.7: Plate with a hole. (A) Geometry and boundary Conditions (values in mm), (B) First principal stress directions

Case B : Bending of plate

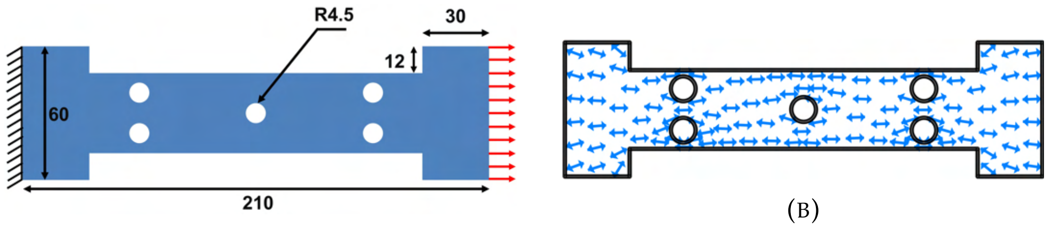


(A)

(B)

FIGURE 4.8: Bending Problem. (A) Geometry and boundary conditions (values in mm), (B) First principal stress directions

Case C : Tensile load on a five holed plate



(A)

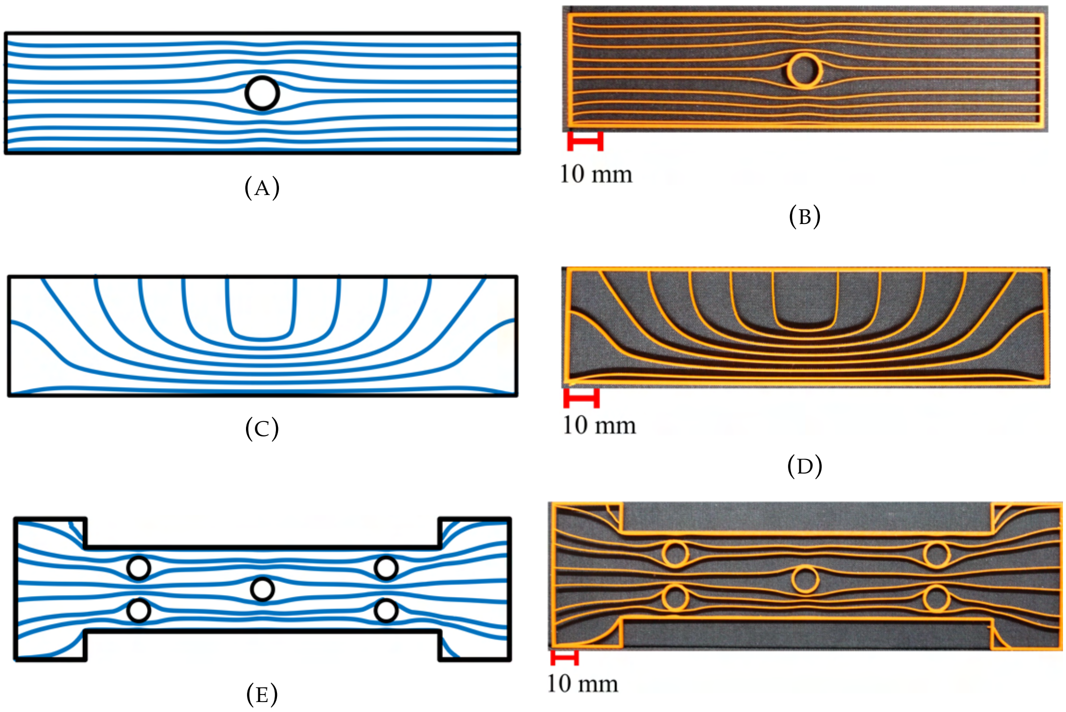
(B)

FIGURE 4.9: Plate with five holes. (A) - Geometry and boundary conditions (values in mm), (B) - First principal stress directions

TABLE 4.1: Input parameters for printing. The parameters given in the table are held constant for all the case studies.

Input Parameters	Values
Extrusion Width (w)	0.5 mm
Layer Height (h)	0.2 mm
Step Length (d_L)	0.3 mm
Number of Layers (n)	8
Non-Printing Travel Speed (V_t)	200 mm/s
Printing Speed (V_p)	25 mm/s
Retraction Length (R_L)	4 mm
Filament Diameter (D_ϕ)	2.85 mm
Retraction tolerance (tol)	0.7 to 1 mm

4.2.1 Infill control



(A)

(B)

(C)

(D)

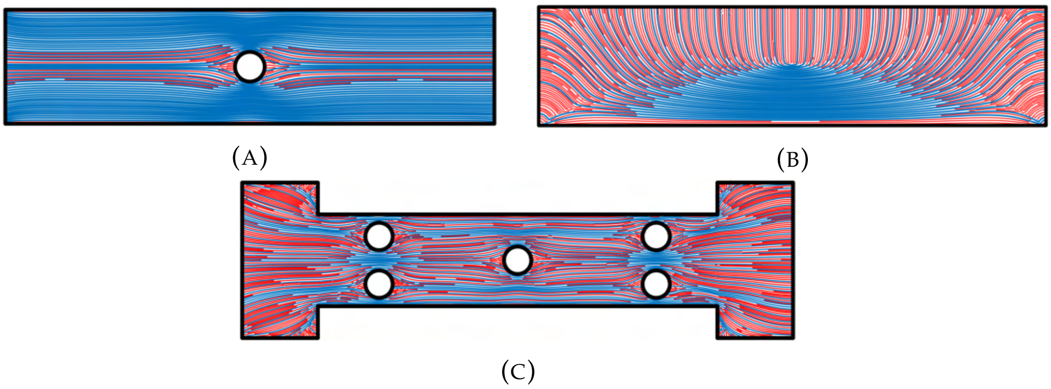
(E)

(F)

FIGURE 4.10: Filament trajectories and Printed Part for G_p = 2 mm. (A) Filament Trajectories for Case A, (B) Printed Part for Case A, (C) Filament Trajectories for Case B, (D) Printed Part for Case B, (E) Filament Trajectories for Case C, (F) Printed Part for Case C.

The algorithm's capability to produce different infills is presented here. First, inter-filament spacing of G_p = 2 mm is used, corresponding to a positive gap between the filaments that yields a low-density part. Note that G_p refers to the minimum spacing between adjacent filaments. A maximum spacing value cannot be prescribed here (as in the previous chapter) since the filament trajectories are constrained to follow a specific orientation field. Figures 4.10a, 4.10c, and 4.10e show the resulting filament trajectories, while Figures 4.10b, 4.10d, and 4.10f show the printed parts. Note that the residual gaps are not filled with the discontinuous filaments, as described in section 4.1.2. This result is presented only to show the capability of the algorithm to produce partial-filled parts. Such parts might not be suitable for structural purposes since they offer stiffness and strength only along the filament directions.

Next, a spacing value of G_p = -0.05 mm is used, meaning that the filaments are constrained to overlap with each other. Closely packed filaments with least voids lead to better mechanical properties for the printed part (Rodríguez, Thomas, and Renaud, 2001; Koch, Van Hulle, and Rudolph, 2017; Rodríguez-Panes, Claver, and Camacho, 2018). Nevertheless, besides aiming for the maximum infill, the algorithm is also strategized to prefer continuous filaments. The reason is that the continuous filaments running from one boundary to another carry loads more effectively and are structurally beneficial. On the other hand, the discontinuous filaments give rise to more intra-layer voids and reduce both the part's stiffness and its strength. Figures 4.11a, 4.11b, and 4.11c show the resulting filament trajectories for the three examples, in which the continuous trajectories are in blue and the discontinuous trajectories are in red. The corresponding G-code paths are shown in Figures 4.12, and the photos of the printed parts are shown in Figure 4.13.

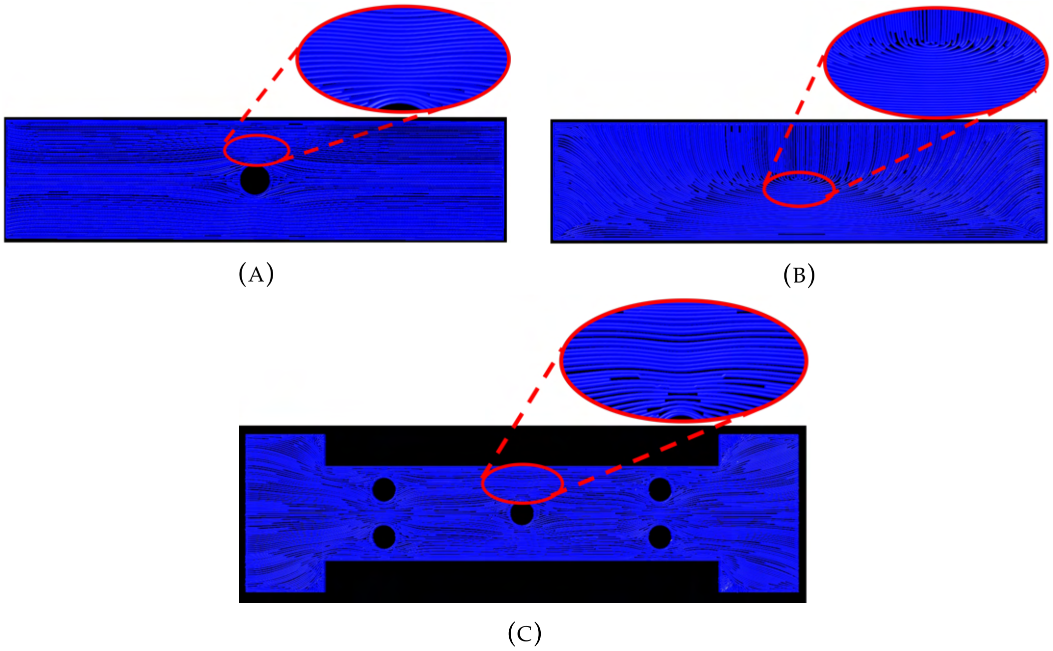


(A)

(B)

(C)

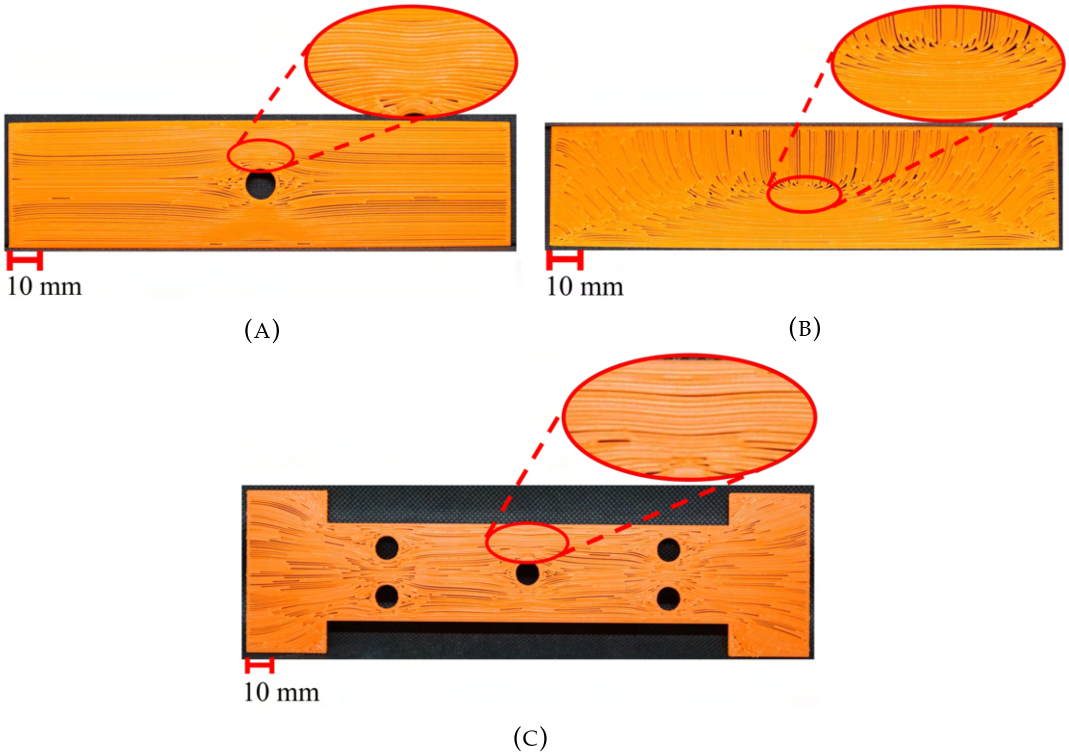
FIGURE 4.11: Filament Trajectories for G_p = -0.05 mm. (a) Case A, (b) Case B, (c) Case C. The continuous filament trajectories are shown in blue color and the discontinuous filament trajectories are shown in red color.


(A)

(B)

(C)

FIGURE 4.12: G-code paths visualized for G_p = -0.05 mm. (A) Case A, (B) Case B, (C) Case C.



(A)

(B)

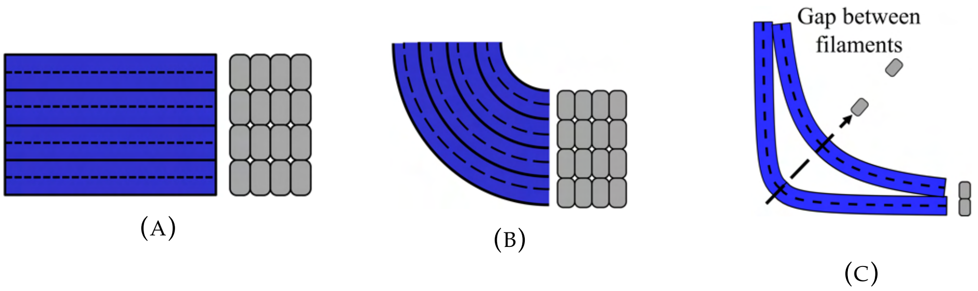
(C)

FIGURE 4.13: Photos of the printed parts for G_p = -0.05 mm. (A) Case A, (B) Case B, and (C) Case C.

Cases	A	B	C
Filament Area Coverage A_f [%]	87	89	90
Time T_f [min]	30	18	51
Printing Time T_p [min]	58	66	107

TABLE 4.2: Area coverage and time taken for G_p = -0.05 mm. A_f is the percentage of area covered by the filaments in the geometry. T_f is the time taken by the algorithm to fetch the trajectory solution and T_p is the time taken the print the parts.

Table 4.2 reports the filament area coverage A_f, the computational time T_f for the filament trajectories (measured on an Intel(R) Xeon(R) W-2125 CPU (4.00 GHz) with 192 GB RAM), and the printing time T_p for the parts. For all three cases, the maximum filament area coverage A_f is around 90%. In commercial slicing software, the user specifies the infill value beforehand and, as a result, obtains parts of predetermined density. Such a condition is elementary to achieve with the predefined deposition patterns which place filaments in a parallel fashion (Figures 4.14a and 4.14b). However, unlike parallel filaments, the filaments following an orientation field change their direction at every point and are prone to develop intra-layer voids (Kulkarni and Dutta, 1999). These voids cannot be entirely filled using filaments of constant width (Figure 4.14c), which ultimately reduce the area coverage for the printed part. It is worth noting that this situation is contrary to the case of the filament path optimization framework described in Chapter 3, where the occurrence of gaps can be better controlled using the parameter l_{max}.



(A)

(B)

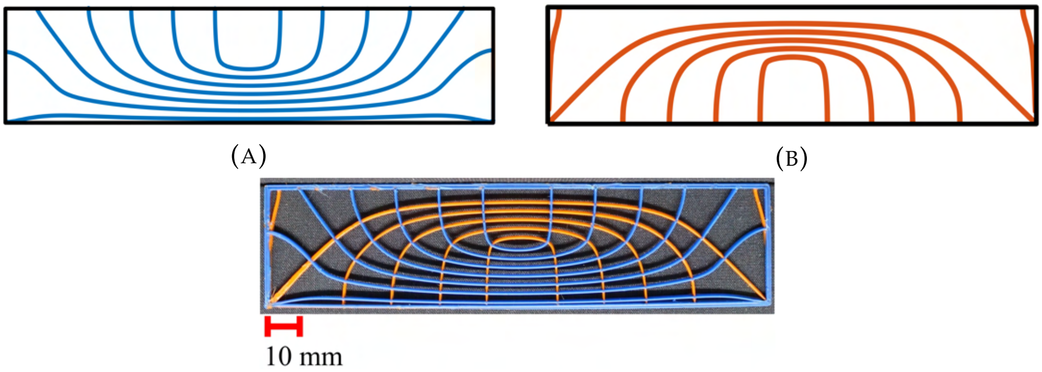
(C)

FIGURE 4.14: Filament Placement Characteristics. (a) Linear Filaments, (b) Constant Curvature Filaments, (c) Variable Curvature Filaments. The filaments following an orientation field have variable curvature and tend to leave gaps.

4.2.2 Multi-oriented Stacking Sequence

Multi-oriented stacking is commonly adopted in FFF parts to overcome the weak mechanical properties along non-printing directions. Commercial slicing software can print different stacking sequences, wherein a sequence comprises two or more layers of different orientations alternately repeated to produce a part. The resulting part has a lattice-type mesostructure with a periodically repeated unit-cell characterized by quasi-isotropic mechanical properties. The rationale behind such a strategy is to have a part characterized by high average strength and stiffness in all directions. The proposed algorithm fully exploits this concept, enabling complete freedom in tailoring FFF parts' properties according to the requested mechanical performance.

To describe this approach, Case B is taken as an example. Two orientation fields from the first and second principal stress directions are used as input. Figures 4.15a and 4.15b show the resulting filament trajectories for $G_p = 2$ mm, while Figure 4.15c shows the printed part. The part comprises eight layers, in which two layers having different filament trajectories, and termed P and Q, are alternately stacked in a sequence PQ to produce the final part. Note that the two constituent layers are also printed in different materials.


(A)

(B)

(C)

FIGURE 4.15: Bending problem with multi-oriented stacking sequence. (a) Solution P from the first principal stress directions of Case B, (b) Solution Q from the second principal stress directions of Case B, (c) Part printed with solutions P and Q, stacked alternately and in different materials.

4.2.3 Experimental Measure of Inter-Filament Spacing:

The inter-filament spacing conditions set for the maximum- and partial-filled parts are verified experimentally. The printed parts are viewed through a microscope, and the centerline distances (i.e., $w + G_p$) between the filaments are measured with the help of *MII ImageView* software.

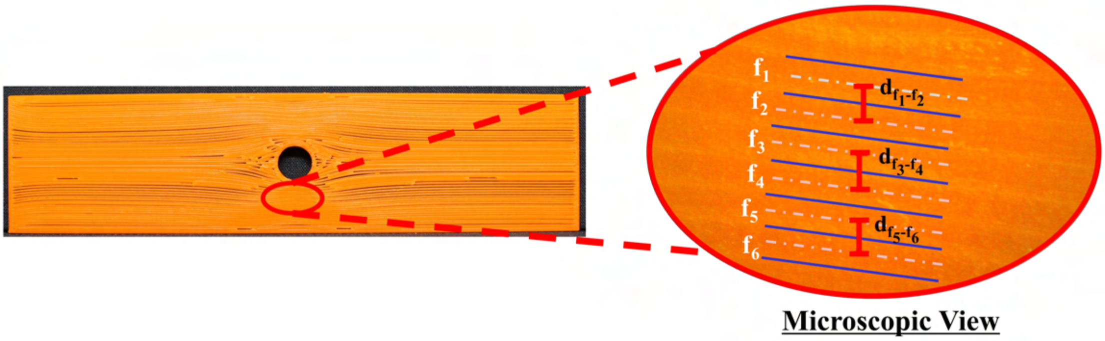


FIGURE 4.16: Experimental validation of the inter-filament spacing for maximum-filled plate with a hole. Centerline distance between filaments f_1 and f_2, $d_{f_1-f_2}$ = 0.46 mm ; centerline distance between filaments f_3 and f_4, $d_{f_3-f_4}$ = 0.48 mm ; centerline distance between filaments f_5 and f_6, $d_{f_5-f_6}$ = 0.45 mm. The boundaries of the filaments in the microscopic view are marked in blue color and their centerlines are depicted in dashed white color.

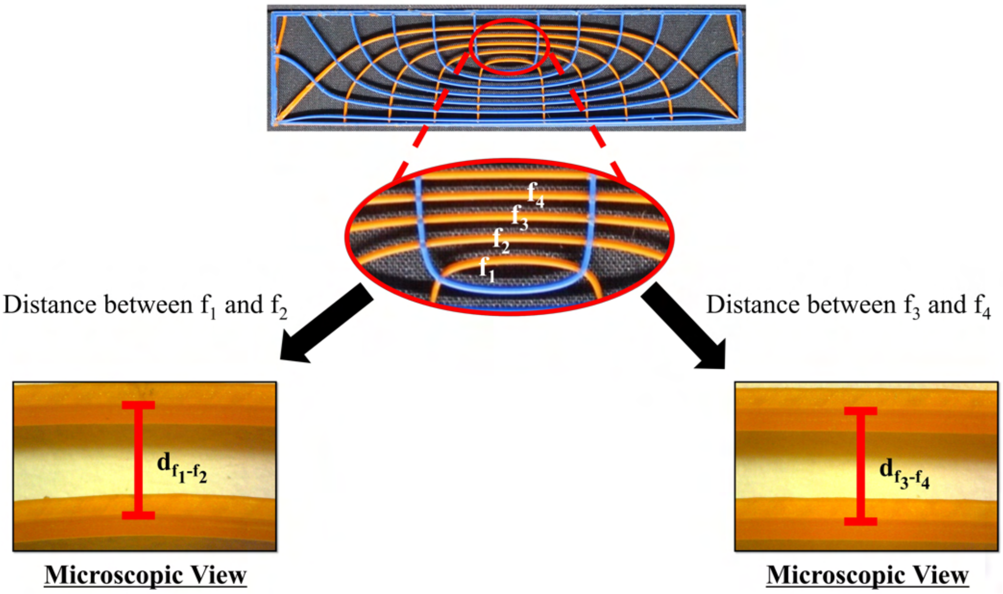


FIGURE 4.17: Experimental validation of the inter-filament spacing for partial-filled part with multi-oriented stacking sequence. Centerline distance between filaments f_1 and f_2, $d_{f_1-f_2}$ = 2.54 mm ; centerline distance between filaments f_3 and f_4, $d_{f_3-f_4}$ = 2.51 mm.

Figure 4.16 shows the centerline distances measured between three pairs of filaments in the maximum-filled plate with a hole. The corresponding values of extrusion width and the minimum spacing are w = 0.5 mm and G_p

= -0.05 mm, respectively. The filaments f_1-f_2, f_3-f_4, and f_5-f_6 (present in the densest region) have centerline distances of $d_{f_1-f_2}$ = 0.46 mm, $d_{f_3-f_4}$ = 0.48 mm, and $d_{f_5-f_6}$ = 0.45 mm, respectively. All the measured values are greater than or equal to w + G_p = 0.45 mm, thus, respecting the spacing condition imposed in the density control step (refer to Eq. (4.5)).

Similarly, Figure 4.17 shows the filament centerline distances for the part with the multi-oriented stacking sequence having G_p = 2 mm. In a region with the least spacing, the centerline distance between filaments f_1 and f_2 is $d_{f_1-f_2}$ = 2.54 mm, and between filaments f_3 and f_4 is $d_{f_3-f_4}$ = 2.51 mm. Thus, the measured values are larger than w + G_p = 2.5 mm, which was the condition imposed at the density control step.

4.2.4 Effect of seed points on coverage and efficiency

Seed points determine the starting point of streamlines and thus, both their number and position play a major role on the computation of filament trajectories. However, the current work did not place the seed points informatively. They were distributed throughout the infill domain so that the dependency of the infill quality on the seed positions is eliminated. This section provides some insights on choosing the seed point number by studying their influence on both area coverage and the computational time of the algorithm.

Figure 4.18 shows the percentage of area covered by the continuous filaments (A_{cf}) and total time taken to yield all filament trajectories (T_f) against different number of seed points (N_S) used, with G_p = -0.05 mm and w = 0.5 mm. Note that the area covered by the discontinuous filaments is not included here since the initially placed seed points do not directly affect the gap-filling step. Anyway, the percentage of discontinuous filaments can be inferred by comparing the total filament area A_f and the continuous filament area A_{cf} in Table 4.2 and Figure 4.18, respectively.

It can be observed that the computational time increases with respect to the number of seed points. This is owing to the fact that increasing the number of seed points increases the number of generated streamlines which, in turn, increases the computational effort of the density control section. A different behaviour can be observed for the filament area coverage. Although the area coverage initially increases with seed points, it converges for all three cases. For a given value of G_p and w, there is only a fixed number of filaments that can be placed in the infill domain. Accordingly, once the area coverage has converged, increasing the number of seed points is not only useless with respect to the quality of the solution, but it unnecessarily increases the computational time of the algorithm. For instance, by increasing the seed point number from 500 at B_1 to nearly 3000 at B_2, the percentage of area covered by continuous filaments increased meagerly, close to 3%. However, the computational time increased four-fold from approximately 5 minutes to 20 minutes. Thus, an optimal number of seed points is the one that yields the best area coverage with the least computational time.

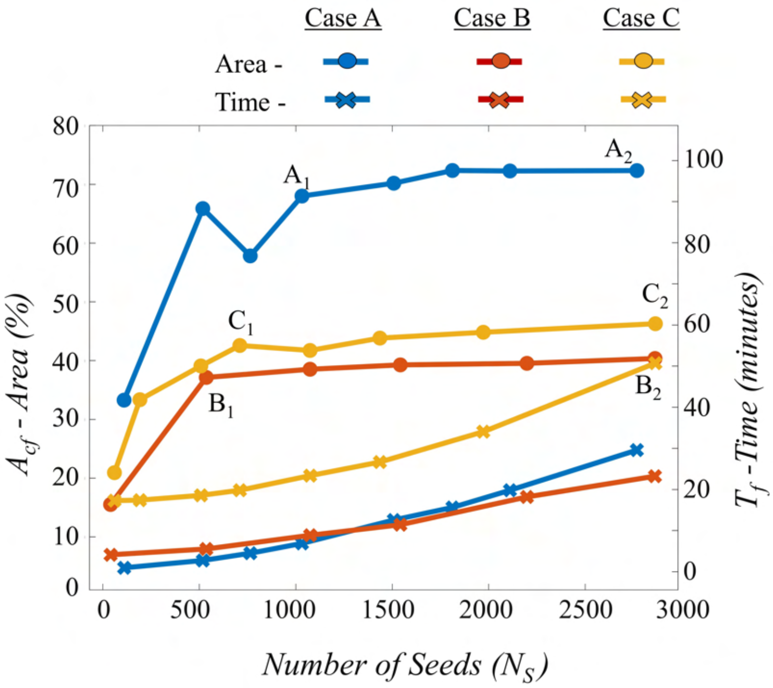


FIGURE 4.18: Influence of seed point number on area coverage and evaluation time for G_p = -0.05 mm and w = 0.5 mm. A_{cf} is the percentage of area covered by continuous filaments and T_f is the time taken by the algorithm to fetch the filament path solution. Between the pairs of points A_1 - A_2, B_1 - B_2 and C_1 - C_2, A_{cf} remains almost constant while T_f continuously increases.

TABLE 4.3: Thumb rules for seed point number selection. N_S - the number of seed points, A_{in} - Infill Area in mm^2

Cases	**Thumb-Rules**
A	$N_S \geq 0.2A_{in}$
B	$N_S \geq 0.1A_{in}$
C	$N_S \geq 0.09A_{in}$

Based on the convergence study in Figure 4.18, certain thumb-rules are derived for choosing the seed point number. Let us define seed point density as ρ_s=N_S/A_{in}, where N_S is the number of seeds in a geometry having an infill area of A_{in}. At the points A_1, B_1 and C_1 in the Figure 4.18, the respective values of ρ_s are roughly estimated to be 0.2, 0.1 and 0.09 number of seed points/mm^2. Thus, the minimum number of seed points required to attain a converged filament path solution for the three examples can be stated using the thumb-rules in Table 4.3.

4.3 Summary

A new free-form filament deposition algorithm was proposed to deposit filaments along the orientation fields that result from the point-wise optimization strategies. Its capabilities were tested by printing complex geometries at different infills and with multi-oriented stacking sequence. The proposed algorithm proved to be able to deposit filaments following a given orientation field, thus enabling higher exploitation of the deposition freedom of the FFF technology.

Experimental studies on FFF parts having varying in-plane filament orientations are lacking, since most works used fixed orientation per layer (Rodríguez, Thomas, and Renaud, 2001; Ahn et al., 2002; Li et al., 2002; Durgun and Ertan, 2014; Riddick et al., 2016; Dawoud, Taha, and Ebeid, 2016). Next steps of the work will investigate mechanical performances of components printed with the proposed algorithm against standard infill patterns. Such experiments can also help determine the role of discontinuities in the final structural performance. Further improvements of the algorithm will include the implementation of variable width filaments, the extension to multi-axis systems (Kubalak, Wicks, and Williams, 2019b) for printing more complex structures and account for an increased number of printing parameters such as the number of perimeters, flow velocity etc.

Chapter 5

Conclusion

Currently, a lack of dedicated design and manufacturing tools for Fused Filament Fabrication (FFF) impedes its transition from prototyping to load-bearing applications. In particular, the slicing software are agnostic to the role played by filament deposition paths on the structural performance of FFF parts. They practise filament deposition in predefined patterns and fail to capitalize on the anisotropy of the FFF parts, leading to an inferior structural performance. To overcome this, the present thesis proposed new tools to design and print optimized filaments that improve the structural performance of FFF parts.

Two new contributions were proposed. The first and the major contribution addressed the lack of manufacturing considerations in the existing filament path optimization techniques. A new filament path optimization framework was developed, which directly imposes the manufacturing constraints into the optimization process to yield readily-printable filament paths that minimize the compliance of FFF parts. The second contribution addressed the algorithmic shortcomings that affect the printing of the optimized solutions. Existing filament path optimization strategies often produce solutions that either cannot be printed or need conversion to the closest printing pattern of the slicing software that, in turn, reduces the maximum structural performance of FFF parts (Hoglund and Smith, 2016; Catapano et al., 2019). Therefore, a new filament deposition algorithm was developed to print the solutions resulting from the existing optimization strategies without relying on the slicing software.

Initially, the essentials of both the contributions were presented in Chapter 2. The chief mechanical characteristics of FFF parts were presented along with the state-of-the-art methods to predict their behaviour. Then, the direct and indirect strategies to parametrize in-plane filament paths were discussed, comparing their attributes that allow the printing of optimized solutions. Finally, the filament path optimization problem was stated in a general setting and two different approaches were detailed to solve the problem, including a two-step gradient-based method and the differential evolution.

Chapter 3 developed the filament path optimization framework that offers production-ready solutions for FFF parts. An indirect parametrization technique of Fernandez et al., 2019 was employed for this purpose. It involves expressing the in-plane filament paths, the material model, and the manufacturing constraints in terms of the level-sets of a B-Surface and using its control parameters as the design variables in the optimization process. The technique was applied to minimize the compliance of some illustrative two-dimensional structures under linear elastic, small strain regimes. Then, the resulting optimized filament patterns were readily converted into G-code instructions and printed without much post-processing steps, preserving the optimality conceived at the design stage.

To exhibit the advantage of filament path optimization, the structural compliance of the optimized and the standard filament patterns were compared, both numerically and experimentally. The numerical study assumed maximum-filled, strongly anisotropic material for different FFF parts, and showed that the final optimized compliance were $35\% - 75\%$ smaller than the compliance of the 0°/90° and 45°/-45° patterns. On the other hand, the experimental study compared the optimized pattern of a 20%-filled ABS specimen against a 0°/90°counterpart using a three-point bending test. The results showed that the optimized patterns were 4.7 times stiffer than the standard 0°/90°pattern (i.e., $\approx 80\%$ compliance reduction), thus proving the significance of printing optimized filament patterns.

Besides displaying the benefits of optimization, this contribution also includes the following novelties: (i) the development and the implementation of a new phenomenological material model for FFF parts that incorporates the transverse stiffness loss in the absence of inter-filament fusion, and (ii) the comparative study between the two-step gradient-based method and the differential evolution for the filament path optimization. The results showed that the new material model yields more realistic filament patterns compared to the state-of-the-art material model that neglected the transverse stiffness loss without inter-filament fusion. In addition, the comparison of optimization approaches suggested the preference for the two-step gradient-based approach due to its better efficiency, flexibility, and compatibility with the proposed material model.

Chapter 4 developed the filament deposition algorithm for printing the orientation fields resulting from the commonly used point-wise optimization strategies, which the slicing software cannot print directly. The proposed algorithm, in addition to printing along an orientation field, also allows for the control of various printing parameters such as extrusion width, spacing, nozzle velocity etc., to support easy customization of the filament designs. Its capabilities were tested by printing complex two-dimensional geometries with filaments tracing the principal stress directions at different infills. The proposed algorithm proved to be able to deposit filaments along the input orientation fields and generated suitable G-code instructions to print them, yielding a more accurate, efficient and flexible tool than the ones commonly used.

Thus, in conclusion, the contributions of this thesis exploit the flexibility of the FFF technology and enable a smooth process from designing to printing of structurally-informed filament paths for FFF components, avoiding the use of the slicing software that compromises their functionality. In doing so, the proposed tools prove capable of realizing function-centric designs and enhance the potential of FFF parts in serving as critical end-use components.

Outlook

There are multiple facets to this research that require further attention. They are listed below in the order of preference.

- □ **Experimental validation:** A preliminary experimental investigation in Chapter 3 proved that the optimized filament patterns were less compliant than the standard filament patterns of the slicing software. As a further step, this study can be extended to more design scenarios such as different infill percentages, filament width, layer height etc. Besides, a comparison between the filament patterns of Chapters 4 and the standard filament patterns will be useful, as it can reveal how far the discontinuities affect the structural performance.

- □ **Comparison of the filament designs from different optimization approaches:** When comparing different optimization approaches, such as the point-wise method (Hoglund and Smith, 2016) and the B-surface method (Fernandez et al., 2019), only the design space and manufacturability aspects are mentioned in the literature. To the author's knowledge there is no discussion on the relative mechanical performances of the final optimized solutions from different approaches. To close this gap, the filament designs from Chapter 3 and Chapter 4 can be compared, both numerically and experimentally. For a numerical comparison, the principal stress directions of a loaded structure can be extracted, and the compliance of the material oriented along the principal stress directions at each FE can be evaluated against the designs optimized using the B-surface. To account for the effect of the residual gaps between the filaments along the principal stress directions, an average volume ratio can be estimated from the filament area coverage and applied to the material model.

□ **Improvements to the material model:** The phenomenological material model proposed in Chapter 3 can be improved. Considering the crucial influence of the activation point ν_a on the predicted structural response, the parameter can be calibrated through tensile tests along the transverse filament directions for different volume ratios. Further, the material model can also be extended to include non-linear effects and failure mechanisms.

□ **Improvements to the filament deposition algorithm:** The filament deposition algorithm in Chapter 4 can be further developed to include more features such as the variable-width filaments, number of perimeters, flow velocity etc., for printing more complex FFF parts. In particular, equipping the algorithm to print variable-width filaments can eliminate the residual voids and ensure a print quality comparable to the results of Chapter 3.

□ **Extension to 3D and flexible geometries:** The proposed tools can be extended for optimizing three-dimensional structures, in which the structural responses are computed layer-wise using 3D finite elements. Besides, the extension of this concept to flexible geometries (i.e., with changes in topology) and multi-axis systems can serve a new set of potential applications.

□ **Process design:** Like most works in the literature, the present work exclusively dealt with the part design while ignoring the process design aspects that are indispensable for the reliability of the FFF system. Future works can concurrently optimize the building and process parameters to achieve the desired functional requirements. To effectively tackle the enlarged design space due to multiple parameter influence, data-driven approaches and Machine Learning algorithms can be employed.

Appendix A

Example G-Code

A sample G-code to generate a rectangular geometry is given below. Figure A.1 visualizes the G-code.

```
M400 T1 ; Select new extruder
M104 S215 ; Set extrusion temperature
G1 X256.38 Y90.50 F9000 ; Move nozzle to starting point
G92 E0
M106 S255 ; Set fan speed fan
; Generate filament paths
G1 X256.38 Y90.5 Z0.2 E0 F360 ; Lower the nozzle to print first layer
G1 E3.5 F2100 ; Purge material
G1 X256.38 Y111.56 E3.9902 F1080 ; Start printing
G1 X256.05 Y112.22 E4.0075
G1 X255.5 Y112.38 E4.0207
G1 X44.94 Y112.38 E8.9221
G1 X44.28 Y112.05 E8.9394
G1 X44.12 Y111.5 E8.9526
G1 X44.12 Y50.44 E10.374
G1 X44.45 Y49.78 E10.3912
G1 X45 Y49.63 E10.4044
```

```
12. G1 X255.56 Y49.62 E15.3058

13. G1 X256.22 Y49.95 E15.3231

14. G1 X256.38 Y50.5 E15.3363

15. G1 X256.38 Y90.5 E16.2674

16. G1 E12.7674 F2100 ; Retract material

17. G1 X0 Y0 F9000 ; Move nozzle to home position
```

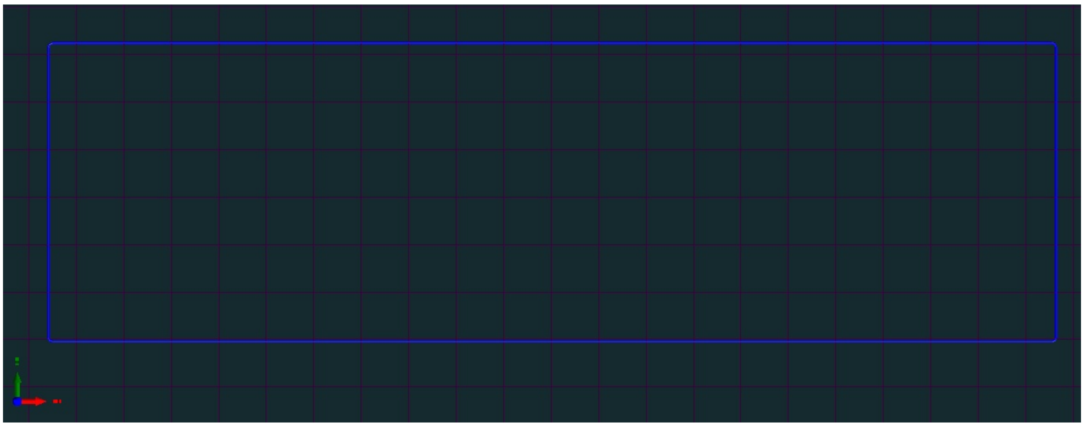


FIGURE A.1: Visualization of the G-code to generate a rectangle. The blue lines depict the filament paths.

Appendix B

Algorithm for Filament Path Optimization

The AceGen and AceFEM codes for optimizing the filament paths of the two-layered MBB structure in Figure 3.8 are presented below.

B.1 AceGen Code

```
<< AceGen`

(*--------------------------INITIALIZATION----------------------*)
SMSInitialize["M1FE50nu0.99Del0.1", "Environment" -> "AceFEM"];

SMSTemplate["SMSTopology" -> "Q1", "SMSSymmetricTangent" -> True,
  "SMSDomainDataNames" -> {"Em - Young's Modulus",
    "Nu -Poisson's ratio", "t -thickness", "nCPX", "nCPY", "hx",
    "hy", "minX", "minY", "XY1", "XY2", "XY3", "XY4", "XY5", "XY6",
    "XY7", "XY8", "XY9", "XY10", "XY11", "XY12", "XY13", "XY14",
    "XY15", "XY16", "XY17", "XY18", "XY19", "XY20", "XY21", "XY22",
    "XY23", "XY24", "XY25", "XY26", "XY27", "XY28", "XY29", "XY30",
    "XY31", "XY32", "XY33", "XY34", "XY35", "XY36", "XY37", "XY38",
    "XY39", "XY40", "XY41", "XY42", "XY43", "XY44", "XY45", "XY46",
    "XY47", "XY48", "XY49", "XY50", "D1", "D2", "D3", "D4", "D5",
    "D6", "D7", "D8", "D9", "D10", "D11", "D12", "D13", "D14", "D15",
    "D16", "D17", "D18", "D19", "D20", "D21", "D22", "D23", "D24",
    "D25", "E1", "E2", "E3", "E4", "E5", "E6", "E7", "E8", "E9",
    "E10", "E11", "E12", "E13", "E14", "E15", "E16", "E17", "E18",
    "E19", "E20", "E21", "E22", "E23", "E24", "E25", "αg1"},
  "SMSDefaultData" -> {200, 0.3, 1, 4, 5, 10, 10, 0, 0, 1, 2, 3, 4, 5,
     6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
    23, 24, 25, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
     17, 18, 19, 20, 21, 22, 23, 24, 25, 1, 2, 3, 4, 5, 6, 7, 8, 9,
    10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 1,
   2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
  20, 21, 22, 23, 24, 25, 45}, "SMSCharSwitch" -> {"Compliance"}];

(*-------------------------FINITE ELEMENT \
QUANTITIES----------------------------*)
ElementDefinitions[] := (
  nnodes = SMSNoNodes; ndim = SMSNoDimensions; ndof = SMSNoDOFGlobal;
  {Em, Nu, t} |- SMSReal[Table[es$$["Data", i], {i, 3}]];
  (*--Youngs Modulus and Poissions Ratio if Isotropic Material is used--*)
  {nCPX, nCPY, HX, HY, MinX, MinY} |-
   SMSReal[Table[es$$["Data", i], {i, 4, 9}]];
  NCP |- nCPX*nCPY;



  XCP |- SMSReal[Table[es$$["Data", i], {i, 10, 34}]];
  YCP |- SMSReal[Table[es$$["Data", i], {i, 35, 59}]];
  dCP |- SMSReal[Table[es$$["Data", i], {i, 60, 84}]];
  eCP |-
   SMSReal[Table[
     es$$["Data", i], {i, 85, Length[SMSDomainDataNames] - 1}]];
  α1 |-
   SMSReal[es$$["Data", Length[SMSDomainDataNames]]];

  XCP1 |- SMSArray[Table[XCP[[i]], {i, 1, Length[XCP]}]];
  YCP1 |- SMSArray[Table[YCP[[i]], {i, 1, Length[YCP]}]];
  dCP1 |- SMSArray[Table[dCP[[i]], {i, 1, Length[dCP]}]];
  eCP1 |- SMSArray[Table[eCP[[i]], {i, 1, Length[eCP]}]];
  lmax = 1.6;
  b = 0.6;



  XIO |-
   Table[SMSReal[nd$$[i, "X", j]], {i, nnodes}, {j, ndim}];
  uIO |-
   Table[SMSReal[nd$$[i, "at", j]], {i, nnodes}, {j, ndim}];
  pe = Flatten[uIO];
  Ξ = {ξ, η, ζ} |-
    Table[SMSReal[es$$["IntPoints", i, Ig]], {i, 3}];
  Nh |=
   1/4 {(1 - ξ) (1 - η), (1 + ξ) (1 - η), (1 + \
ξ) (1 + η), (1 - ξ) (1 + η)}; (*
  FE Shape functions *)
  X |- SMSFreeze[Append[Nh.XIO, ζ]];
  u |= Append[Nh.uIO, 0];
  Je |= SMSD[X, Ξ];
  Jed |= Det[Je];
```

```
H ⊨
 SMSD[u, X, "Dependency" -> {Ξ, X, SMSInverse[Je]}];


SMSFreeze[ϵ, 1/2 (H + Transpose[H]), "Symmetric" -> True];


PtchCoord ⊨ {X[[1]], X[[2]]};
PtchNoX ⊨ SMSInteger[(PtchCoord[[1]] - MinX)/HX] + 1;
PtchNoY ⊨ SMSInteger[(PtchCoord[[2]] - MinY)/HY] + 1;
startI ⊨ SMSInteger[PtchNoX + (nCPX∗(PtchNoY - 1))];

xLoc ⊨ {};
yLoc ⊨ {};
zLoc ⊨ {};
z2Loc ⊨ {};

AppendTo[xLoc, SMSPart[XCP1, startI]];
AppendTo[xLoc, SMSPart[XCP1, startI + 1]];
AppendTo[xLoc, SMSPart[XCP1, startI + 2]];
AppendTo[xLoc, SMSPart[XCP1, startI + 3]];
AppendTo[yLoc, SMSPart[YCP1, startI]];
AppendTo[yLoc, SMSPart[YCP1, startI + 1]];
AppendTo[yLoc, SMSPart[YCP1, startI + 2]];
AppendTo[yLoc, SMSPart[YCP1, startI + 3]];
AppendTo[zLoc, SMSPart[dCP1, startI]];
AppendTo[zLoc, SMSPart[dCP1, startI + 1]];
AppendTo[zLoc, SMSPart[dCP1, startI + 2]];
AppendTo[zLoc, SMSPart[dCP1, startI + 3]];
AppendTo[z2Loc, SMSPart[eCP1, startI]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 1]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 2]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 3]];

startI ⊨ SMSInteger[startI + nCPX];
AppendTo[xLoc, SMSPart[XCP1, startI]];
AppendTo[xLoc, SMSPart[XCP1, startI + 1]];
AppendTo[xLoc, SMSPart[XCP1, startI + 2]];
AppendTo[xLoc, SMSPart[XCP1, startI + 3]];
AppendTo[yLoc, SMSPart[YCP1, startI]];
AppendTo[yLoc, SMSPart[YCP1, startI + 1]];
AppendTo[yLoc, SMSPart[YCP1, startI + 2]];
AppendTo[yLoc, SMSPart[YCP1, startI + 3]];
AppendTo[zLoc, SMSPart[dCP1, startI]];
AppendTo[zLoc, SMSPart[dCP1, startI + 1]];
AppendTo[zLoc, SMSPart[dCP1, startI + 2]];
```

```
AppendTo[zLoc, SMSPart[dCP1, startI + 3]];
AppendTo[z2Loc, SMSPart[eCP1, startI]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 1]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 2]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 3]];
startI |= SMSInteger[startI + nCPX];
AppendTo[xLoc, SMSPart[XCP1, startI]];
AppendTo[xLoc, SMSPart[XCP1, startI + 1]];
AppendTo[xLoc, SMSPart[XCP1, startI + 2]];
AppendTo[xLoc, SMSPart[XCP1, startI + 3]];
AppendTo[yLoc, SMSPart[YCP1, startI]];
AppendTo[yLoc, SMSPart[YCP1, startI + 1]];
AppendTo[yLoc, SMSPart[YCP1, startI + 2]];
AppendTo[yLoc, SMSPart[YCP1, startI + 3]];
AppendTo[zLoc, SMSPart[dCP1, startI]];
AppendTo[zLoc, SMSPart[dCP1, startI + 1]];
AppendTo[zLoc, SMSPart[dCP1, startI + 2]];
AppendTo[zLoc, SMSPart[dCP1, startI + 3]];
AppendTo[z2Loc, SMSPart[eCP1, startI]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 1]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 2]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 3]];

startI |= SMSInteger[startI + nCPX];
AppendTo[xLoc, SMSPart[XCP1, startI]];
AppendTo[xLoc, SMSPart[XCP1, startI + 1]];
AppendTo[xLoc, SMSPart[XCP1, startI + 2]];
AppendTo[xLoc, SMSPart[XCP1, startI + 3]];
AppendTo[yLoc, SMSPart[YCP1, startI]];
AppendTo[yLoc, SMSPart[YCP1, startI + 1]];
AppendTo[yLoc, SMSPart[YCP1, startI + 2]];
AppendTo[yLoc, SMSPart[YCP1, startI + 3]];
AppendTo[zLoc, SMSPart[dCP1, startI]];
AppendTo[zLoc, SMSPart[dCP1, startI + 1]];
AppendTo[zLoc, SMSPart[dCP1, startI + 2]];
AppendTo[zLoc, SMSPart[dCP1, startI + 3]];
AppendTo[z2Loc, SMSPart[eCP1, startI]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 1]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 2]];
AppendTo[z2Loc, SMSPart[eCP1, startI + 3]];

PCenX |= (xLoc[[7]] + xLoc[[6]])/2;
PCenY |= (yLoc[[10]] + yLoc[[6]])/2;
ξ1 |- SMSReal[2 (X[[1]] - PCenX)/HX];
η1 |- SMSReal[2 (X[[2]] - PCenY)/HY];
NB1 |= {1/48 (1 - 3 ξ1 + 3 ξ1^2 - ξ1^3),
  1/48 (23 - 15 ξ1 - 3 ξ1^2 + 3 ξ1^3),
  1/48 (23 + 15 ξ1 - 3 ξ1^2 - 3 ξ1^3),
  1/48 (1 + 3 ξ1 + 3 ξ1^2 + ξ1^3)}; (* B-
Surface Bicubic Shape functions *)
NB2 ⊢= {1/
    48 (1 - 3 η1 + 3 η1^2 - η1^3),
  1/48 (23 - 15 η1 - 3 η1^2 + 3 η1^3),
  1/48 (23 + 15 η1 - 3 η1^2 - 3 η1^3),
  1/48 (1 + 3 η1 + 3 η1^2 + η1^3)};
NB1NB2 ⊢= Flatten[TensorProduct[NB2, NB1]];
DNξ ⊢= {((1 - 3 η1 +
     3 η1^2 - η1^3) (-3 + 6 ξ1 - 3 ξ1^2))/
  2304, ((1 - 3 η1 + 3 η1^2 - η1^3) (-15 -
     6 ξ1 + 9 ξ1^2))/
  2304, ((1 - 3 η1 + 3 η1^2 - η1^3) (15 - 6 ξ1 -
     9 ξ1^2))/
  2304, ((1 - 3 η1 + 3 η1^2 - η1^3) (3 + 6 ξ1 +
     3 ξ1^2))/
  2304, ((23 - 15 η1 - 3 η1^2 + 3 η1^3) (-3 +
     6 ξ1 - 3 ξ1^2))/
  2304, ((23 - 15 η1 - 3 η1^2 + 3 η1^3) (-15 -
     6 ξ1 + 9 ξ1^2))/
  2304, ((23 - 15 η1 - 3 η1^2 + 3 η1^3) (15 -
     6 ξ1 - 9 ξ1^2))/
  2304, ((23 - 15 η1 - 3 η1^2 + 3 η1^3) (3 +
     6 ξ1 + 3 ξ1^2))/
  2304, ((23 + 15 η1 - 3 η1^2 - 3 η1^3) (-3 +
     6 ξ1 - 3 ξ1^2))/
  2304, ((23 + 15 η1 - 3 η1^2 - 3 η1^3) (-15 -
     6 ξ1 + 9 ξ1^2))/
  2304, ((23 + 15 η1 - 3 η1^2 - 3 η1^3) (15 -
     6 ξ1 - 9 ξ1^2))/
  2304, ((23 + 15 η1 - 3 η1^2 - 3 η1^3) (3 +
     6 ξ1 + 3 ξ1^2))/
  2304, ((1 + 3 η1 + 3 η1^2 + η1^3) (-3 + 6 ξ1 -
     3 ξ1^2))/
  2304, ((1 + 3 η1 + 3 η1^2 + η1^3) (-15 -
     6 ξ1 + 9 ξ1^2))/
  2304, ((1 + 3 η1 + 3 η1^2 + η1^3) (15 - 6 ξ1 -
     9 ξ1^2))/
  2304, ((1 + 3 η1 + 3 η1^2 + η1^3) (3 + 6 ξ1 +
     3 ξ1^2))/2304};
DNη ⊢= {((-3 + 6 η1 - 3 η1^2) (1 -
     3 ξ1 + 3 ξ1^2 - ξ1^3))/
  2304, ((-3 + 6 η1 - 3 η1^2) (23 - 15 ξ1 -
     3 ξ1^2 + 3 ξ1^3))/
```

```
 2304, ((-3 + 6 η1 - 3 η1^2) (23 + 15 ξ1 -
   3 ξ1^2 - 3 ξ1^3))/
 2304, ((-3 + 6 η1 - 3 η1^2) (1 + 3 ξ1 +
   3 ξ1^2 + ξ1^3))/
 2304, ((-15 - 6 η1 + 9 η1^2) (1 - 3 ξ1 +
   3 ξ1^2 - ξ1^3))/
 2304, ((-15 - 6 η1 + 9 η1^2) (23 - 15 ξ1 -
   3 ξ1^2 + 3 ξ1^3))/
 2304, ((-15 - 6 η1 + 9 η1^2) (23 + 15 ξ1 -
   3 ξ1^2 - 3 ξ1^3))/
 2304, ((-15 - 6 η1 + 9 η1^2) (1 + 3 ξ1 +
   3 ξ1^2 + ξ1^3))/
 2304, ((15 - 6 η1 - 9 η1^2) (1 - 3 ξ1 +
   3 ξ1^2 - ξ1^3))/
 2304, ((15 - 6 η1 - 9 η1^2) (23 - 15 ξ1 -
   3 ξ1^2 + 3 ξ1^3))/
 2304, ((15 - 6 η1 - 9 η1^2) (23 + 15 ξ1 -
   3 ξ1^2 - 3 ξ1^3))/
 2304, ((15 - 6 η1 - 9 η1^2) (1 + 3 ξ1 +
   3 ξ1^2 + ξ1^3))/
 2304, ((3 + 6 η1 + 3 η1^2) (1 - 3 ξ1 +
   3 ξ1^2 - ξ1^3))/
 2304, ((3 + 6 η1 + 3 η1^2) (23 - 15 ξ1 -
   3 ξ1^2 + 3 ξ1^3))/
 2304, ((3 + 6 η1 + 3 η1^2) (23 + 15 ξ1 -
   3 ξ1^2 - 3 ξ1^3))/
 2304, ((3 + 6 η1 + 3 η1^2) (1 + 3 ξ1 +
   3 ξ1^2 + ξ1^3))/2304};

DNξη |= Transpose[{DNξ, DNη}];



jbs11 |= DNξ.xLoc;
jbs12 |= DNη.xLoc;
jbs21 |= DNξ.yLoc;
jbs22 |= DNη.yLoc;
JBs |= {{jbs11, jbs12}, {jbs21, jbs22}};

JBsIn |= SMSInverse[Transpose[JBs]];
DNxy |= DNξη.JBsIn;
DCzxy |= Transpose[DNxy].zLoc;
DCzxy2 |= Transpose[DNxy].z2Loc;

MagDCz |= SMSSqrt[DCzxy[[1]]^2 + DCzxy[[2]]^2];
MagDCz2 |= SMSSqrt[DCzxy2[[1]]^2 + DCzxy2[[2]]^2];
JBsDet |= SMSDet[JBs];
```

```
α |-
 SMSFreeze[(180/Pi)*
   ArcTan[SMSFreeze[DCzxy[[2]], "Code" -> True],
    SMSFreeze[-DCzxy[[1]], "Code" -> True]]];  (*
Filament Angle Layer-1 *)


α2 |-
 SMSFreeze[(180/Pi)*
   ArcTan[SMSFreeze[DCzxy2[[2]], "Code" -> True],
    SMSFreeze[-DCzxy2[[1]], "Code" -> True]]]; (*
Filament Angle Layer-2 *)


v |= SMSFreeze[MagDCz]; (*
Filament volume ratio Layer-1 *)

v2 |= SMSFreeze[MagDCz2]; (*
Filament volume ratio Layer-2 *)


(* ------------------------- MATERIAL MODEL NU=0.99 SL =
0.1 MPa------------------------ *)

C1 |= v 152470;


C2 |=
 7999.95 ((Exp[513.28441512576 v - 511.626028344188] -
       Exp[-513.28441512576 v + 511.626028344188])/(Exp[
        513.28441512576 v - 511.626028344188] +
       Exp[-513.28441512576 v + 511.626028344188])) + 8000.05;
C3 |=
 3999.95 ((Exp[480.796605079825 v - 479.483741005463] -
       Exp[-480.796605079825 v + 479.483741005463])/(Exp[
        480.796605079825 v - 479.483741005463] +
       Exp[-480.796605079825 v + 479.483741005463])) + 4000.05;
C4 |=
 2499.95 ((Exp[466.849581592693 v - 465.692775117234] -
       Exp[-466.849581592693 v + 465.692775117234])/(Exp[
        466.849581592693 v - 465.692775117234] +
       Exp[-466.849581592693 v + 465.692775117234])) + 2500.05;
C5 |=
 4999.95 ((Exp[487.38524531497 v - 486.000547780655] -
       Exp[-487.38524531497 v + 486.000547780655])/(Exp[
        487.38524531497 v - 486.000547780655] +
       Exp[-487.38524531497 v + 486.000547780655])) + 5000.05;



C6 |= (C2 - C5)/2;

Rz1 |= {{Cos[α Degree], -Sin[α Degree],
    0}, {Sin[α Degree], Cos[α Degree], 0}, {0, 0, 1}};
C01 |= {{{{C1, 0, 0}, {0, C3, 0}, {0, 0, C3}}, {{0,
     C4, 0}, {C4, 0, 0}, {0, 0, 0}}, {{0, 0, C4}, {0, 0, 0}, {C4, 0,
      0}}}, {{{0, C4, 0}, {C4, 0, 0}, {0, 0, 0}}, {{C3, 0, 0}, {0,
     C2, 0}, {0, 0, C5}}, {{0, 0, 0}, {0, 0, C6}, {0, C6,
     0}}}, {{{0, 0, C4}, {0, 0, 0}, {C4, 0, 0}}, {{0, 0, 0}, {0, 0,
     C6}, {0, C6, 0}}, {{C3, 0, 0}, {0, C5, 0}, {0, 0, C2}}}};

Ca1 |=
 TensorContract[
  TensorProduct[
   C01 , (Transpose[
     Transpose[TensorProduct[Rz1, Rz1], {1, 3, 2, 4}], {3, 4, 1,
      2}])], {{3, 5}, {4, 6}}];
Cm1 |=
 TensorContract[
  TensorProduct[(Transpose[TensorProduct[Rz1, Rz1], {1, 3, 2, 4}]),
   Ca1], {{3, 5}, {4, 6}}];




Ψ1 |= (1/2) TensorContract[
   TensorProduct[Cm1, ϵ], {{3, 5}, {4, 6}}];
Ψ1 |=
 TensorContract[
  TensorProduct[Ψ1, ϵ], {{1, 3}, {2, 4}}];



C1 |= v2 152470;

C2 |=
 7999.95 ((Exp[513.28441512576 v2 - 511.626028344188] -
       Exp[-513.28441512576 v2 + 511.626028344188])/(Exp[
        513.28441512576 v2 - 511.626028344188] +
       Exp[-513.28441512576 v2 + 511.626028344188])) + 8000.05;
C3 |=
 3999.95 ((Exp[480.796605079825 v2 - 479.483741005463] -
        Exp[-480.796605079825 v2 + 479.483741005463])/(Exp[
         480.796605079825 v2 - 479.483741005463] +
        Exp[-480.796605079825 v2 + 479.483741005463])) + 4000.05;
  C4 |=
   2499.95 ((Exp[466.849581592693 v2 - 465.692775117234] -
        Exp[-466.849581592693 v2 + 465.692775117234])/(Exp[
         466.849581592693 v2 - 465.692775117234] +
        Exp[-466.849581592693 v2 + 465.692775117234])) + 2500.05;
  C5 |=
   4999.95 ((Exp[487.38524531497 v2 - 486.000547780655] -
        Exp[-487.38524531497 v2 + 486.000547780655])/(Exp[
         487.38524531497 v2 - 486.000547780655] +
        Exp[-487.38524531497 v2 + 486.000547780655])) + 5000.05;



  C6 |= (C2 - C5)/2;



  Rz2 |= {{Cos[α2 Degree], -Sin[α2 \
Degree], 0}, {Sin[α2 Degree], Cos[α2 Degree], 0}, {0, 0,
      1}};
  C02 |= {{{{C1, 0, 0}, {0, C3, 0}, {0, 0, C3}}, {{0,
       C4, 0}, {C4, 0, 0}, {0, 0, 0}}, {{0, 0, C4}, {0, 0, 0}, {C4, 0,
        0}}}, {{{0, C4, 0}, {C4, 0, 0}, {0, 0, 0}}, {{C3, 0, 0}, {0,
       C2, 0}, {0, 0, C5}}, {{0, 0, 0}, {0, 0, C6}, {0, C6,
       0}}}, {{{0, 0, C4}, {0, 0, 0}, {C4, 0, 0}}, {{0, 0, 0}, {0, 0,
       C6}, {0, C6, 0}}, {{C3, 0, 0}, {0, C5, 0}, {0, 0, C2}}}};

  Ca2 |=
   TensorContract[
    TensorProduct[
     C02, (Transpose[
       Transpose[TensorProduct[Rz2, Rz2], {1, 3, 2, 4}], {3, 4, 1,
        2}])], {{3, 5}, {4, 6}}];
  Cm2 |=
   TensorContract[
    TensorProduct[(Transpose[TensorProduct[Rz2, Rz2], {1, 3, 2, 4}]),
     Ca2], {{3, 5}, {4, 6}}];



  Ψ2 |= (1/2) TensorContract[
     TensorProduct[Cm2, ϵ], {{3, 5}, {4, 6}}];
  Ψ  |= Ψ1 +
    TensorContract[
     TensorProduct[Ψ2, ϵ], {{1, 3}, {2, 4}}];
  )
```

```
(* ------------------------- SOLVE FE EQUATIONS------------------------ \
*)
SMSStandardModule["Tangent and residual"];

SMSDo[Ig, 1, SMSInteger[es$$["id", "NoIntPoints"]]];
ElementDefinitions[];

wgp |- SMSReal[es$$["IntPoints", 4, Ig]];
        SMSDo[m, 1, ndof];
                Rgm |= Jed  SMSD[ Ψ, pe, m];
                SMSExport[wgp  Rgm, p$$[m], "AddIn" ->  True];
                        SMSDo[n, m, ndof];
                                Kgmn |= SMSD[Rgm, pe, n];
                                SMSExport[wgp Kgmn, s$$[m, n], "AddIn" ->  True
                                    ];
                        SMSEndDo[];
        SMSEndDo[];
SMSEndDo[];

(* ------------------------- COMPUTE POST-PROCESSING QUANTITIES
    ------------------------ \
*)

SMSStandardModule["Postprocessing"];

SMSDo[Ig, 1, SMSInteger[es$$["id", "NoIntPoints"]]];
ElementDefinitions[];

σ |=
  SMSD[Ψ, ϵ, "Symmetric" -> True];
Cz |= NB1NB2.zLoc;
Cz2 |= NB1NB2.z2Loc;

SMSGPostNames = {"DispX", "DispY", "DispZ", "Volr", "Volr2",
   "StressX", "StressY", "StressXY", "StrainX", "StrainY", "Cz",
   "Cz2", "angle", "dcz1", "dcz2"};
SMSExport[{u[[1]], u[[2]], u[[3]], MagDCz,
   MagDCz2, σ[[1, 1]], σ[[2, 2]], σ[[1,
     2]], ϵ[[1, 1]], ϵ[[2, 2]], Cz, Cz2, α,
   DCzxy[[1]], DCzxy[[2]]}, gpost$$[Ig, #1] &];
SMSEndDo[];

(* ------------------------- COMPUTE COMPLIANCE AND SENSITIVITIES
    ------------------------ \
*)
```

```
SMSStandardModule["Tasks"];

task |= SMSInteger[Task$$];
SMSIf[task == -1,
  SMSExport[{1, 0, 0, 0, 5 + 50 + 25 + 25 + 25 + 25}, TasksData$$];];
SMSDo[Ig, 1, SMSInteger[es$$["id", "NoIntPoints"]]];
ElementDefinitions[];

wgp |- SMSReal[es$$["IntPoints", 4, Ig]];
σ |=
  SMSD[Ψ, ϵ, "Symmetric" -> True];
comp1 |=
  TensorContract[
   TensorProduct[σ, ϵ], {{1, 3}, {2, 4}}];
comp |= t Jed wgp comp1 ;

cons1Loc |= (MagDCz^2 - 1);
(*cons1A|=SMSIf[cons1Loc<=0,0,(Jed wgp cons1Loc)];*)

SMSIf[cons1Loc <= 0];
cons1 =| 0;
dcons1d1 =| 0; dcons1d2 =| 0; dcons1d3 \
=| 0; dcons1d4 =| 0; dcons1d5 \
=| 0; dcons1d6 =| 0; dcons1d7 \
=| 0; dcons1d8 =| 0; dcons1d9 \
=| 0; dcons1d10 =| 0; dcons1d11 \
=| 0; dcons1d12 =| 0; dcons1d13 \
=| 0; dcons1d14 =| 0;
dcons1d15 =| 0; dcons1d16 =| 0; dcons1d17 \
=| 0; dcons1d18 =| 0; dcons1d19 \
=| 0; dcons1d20 =| 0; dcons1d21 \
=| 0; dcons1d22 =| 0; dcons1d23 \
=| 0; dcons1d24 =| 0; dcons1d25 \
=| 0;
SMSElse[];
cons1 -| (Jed wgp cons1Loc);
dcons1d1 -|
 SMSD[(Jed wgp cons1Loc), dCP[[1]]]; dcons1d2 -|
 SMSD[(Jed wgp cons1Loc), dCP[[2]]]; dcons1d3 -|
 SMSD[(Jed wgp cons1Loc), dCP[[3]]]; dcons1d4 -|
 SMSD[(Jed wgp cons1Loc), dCP[[4]]]; dcons1d5 -|
 SMSD[(Jed wgp cons1Loc), dCP[[5]]]; dcons1d6 -|
 SMSD[(Jed wgp cons1Loc), dCP[[6]]]; dcons1d7 -|
 SMSD[(Jed wgp cons1Loc), dCP[[7]]]; dcons1d8 -|
 SMSD[(Jed wgp cons1Loc), dCP[[8]]]; dcons1d9 -|
 SMSD[(Jed wgp cons1Loc), dCP[[9]]]; dcons1d10 -|
 SMSD[(Jed wgp cons1Loc), dCP[[10]]]; dcons1d11 -|
 SMSD[(Jed wgp cons1Loc), dCP[[11]]]; dcons1d12 -|
 SMSD[(Jed wgp cons1Loc), dCP[[12]]]; dcons1d13 -|
 SMSD[(Jed wgp cons1Loc), dCP[[13]]]; dcons1d14 -|
 SMSD[(Jed wgp cons1Loc), dCP[[14]]];
dcons1d15 -|
 SMSD[(Jed wgp cons1Loc), dCP[[15]]]; dcons1d16 -|
 SMSD[(Jed wgp cons1Loc), dCP[[16]]]; dcons1d17 -|
 SMSD[(Jed wgp cons1Loc), dCP[[17]]]; dcons1d18 -|
 SMSD[(Jed wgp cons1Loc), dCP[[18]]]; dcons1d19 -|
 SMSD[(Jed wgp cons1Loc), dCP[[19]]]; dcons1d20 -|
 SMSD[(Jed wgp cons1Loc), dCP[[20]]]; dcons1d21 -|
 SMSD[(Jed wgp cons1Loc), dCP[[21]]]; dcons1d22 -|
 SMSD[(Jed wgp cons1Loc), dCP[[22]]]; dcons1d23 -|
 SMSD[(Jed wgp cons1Loc), dCP[[23]]]; dcons1d24 -|
 SMSD[(Jed wgp cons1Loc), dCP[[24]]]; dcons1d25 -|
 SMSD[(Jed wgp cons1Loc), dCP[[25]]];

SMSEndIf[cons1, dcons1d1, dcons1d2, dcons1d3, dcons1d4, dcons1d5,
  dcons1d6, dcons1d7, dcons1d8, dcons1d9, dcons1d10, dcons1d11,
  dcons1d12, dcons1d13, dcons1d14,
  dcons1d15, dcons1d16, dcons1d17, dcons1d18, dcons1d19, dcons1d20,
  dcons1d21, dcons1d22, dcons1d23, dcons1d24, dcons1d25];



cons1BLoc |= (MagDCz2^2 - 1);


SMSIf[cons1BLoc <= 0];
cons1B =| 0;
dcons1Bd1 =| 0; dcons1Bd2 =| 0; dcons1Bd3 \
=| 0; dcons1Bd4 =| 0; dcons1Bd5 \
=| 0; dcons1Bd6 =| 0; dcons1Bd7 \
=| 0; dcons1Bd8 =| 0; dcons1Bd9 \
=| 0; dcons1Bd10 =| 0; dcons1Bd11 \
=| 0; dcons1Bd12 =| 0; dcons1Bd13 \
=| 0; dcons1Bd14 =| 0;
dcons1Bd15 =| 0; dcons1Bd16 =| 0; \
dcons1Bd17 =| 0; dcons1Bd18 =| 0; \
dcons1Bd19 =| 0; dcons1Bd20 =| 0; \
dcons1Bd21 =| 0; dcons1Bd22 =| 0; \
dcons1Bd23 =| 0; dcons1Bd24 =| 0; \
dcons1Bd25 =| 0;
SMSElse[];
```

```
cons1B -| (Jed wgp cons1BLoc);
dcons1Bd1 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[1]]]; dcons1Bd2 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[2]]]; dcons1Bd3 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[3]]]; dcons1Bd4 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[4]]]; dcons1Bd5 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[5]]]; dcons1Bd6 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[6]]]; dcons1Bd7 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[7]]]; dcons1Bd8 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[8]]]; dcons1Bd9 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[9]]]; dcons1Bd10 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[10]]]; dcons1Bd11 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[11]]]; dcons1Bd12 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[12]]]; dcons1Bd13 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[13]]]; dcons1Bd14 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[14]]];
dcons1Bd15 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[15]]]; dcons1Bd16 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[16]]]; dcons1Bd17 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[17]]]; dcons1Bd18 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[18]]]; dcons1Bd19 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[19]]]; dcons1Bd20 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[20]]]; dcons1Bd21 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[21]]]; dcons1Bd22 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[22]]]; dcons1Bd23 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[23]]]; dcons1Bd24 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[24]]]; dcons1Bd25 -|
 SMSD[(Jed wgp cons1BLoc), eCP[[25]]];

SMSEndIf[cons1B, dcons1Bd1, dcons1Bd2, dcons1Bd3, dcons1Bd4,
  dcons1Bd5, dcons1Bd6, dcons1Bd7, dcons1Bd8, dcons1Bd9, dcons1Bd10,
  dcons1Bd11, dcons1Bd12, dcons1Bd13, dcons1Bd14,
  dcons1Bd15, dcons1Bd16, dcons1Bd17, dcons1Bd18, dcons1Bd19,
  dcons1Bd20, dcons1Bd21, dcons1Bd22, dcons1Bd23, dcons1Bd24,
  dcons1Bd25];




cons2Loc |= (1 - ((lmax/b) MagDCz )^2);



SMSIf[cons2Loc <= 0];
cons2 =| 0;
dcons2d1 =| 0; dcons2d2 =| 0; dcons2d3 \
=| 0; dcons2d4 =| 0; dcons2d5 \
=| 0; dcons2d6 =| 0; dcons2d7 \
=| 0; dcons2d8 =| 0; dcons2d9 \
=| 0; dcons2d10 =| 0; dcons2d11 \
=| 0; dcons2d12 =| 0; dcons2d13 \
=| 0; dcons2d14 =| 0;
dcons2d15 =| 0; dcons2d16 =| 0; dcons2d17 \
=| 0; dcons2d18 =| 0; dcons2d19 \
=| 0; dcons2d20 =| 0; dcons2d21 \
=| 0; dcons2d22 =| 0; dcons2d23 \
=| 0; dcons2d24 =| 0; dcons2d25 \
=| 0;
SMSElse[];
cons2 -| (Jed wgp cons2Loc);
dcons2d1 -|
 SMSD[(Jed wgp cons2Loc), dCP[[1]]]; dcons2d2 -|
 SMSD[(Jed wgp cons2Loc), dCP[[2]]]; dcons2d3 -|
 SMSD[(Jed wgp cons2Loc), dCP[[3]]]; dcons2d4 -|
 SMSD[(Jed wgp cons2Loc), dCP[[4]]]; dcons2d5 -|
 SMSD[(Jed wgp cons2Loc), dCP[[5]]]; dcons2d6 -|
 SMSD[(Jed wgp cons2Loc), dCP[[6]]]; dcons2d7 -|
 SMSD[(Jed wgp cons2Loc), dCP[[7]]]; dcons2d8 -|
 SMSD[(Jed wgp cons2Loc), dCP[[8]]]; dcons2d9 -|
 SMSD[(Jed wgp cons2Loc), dCP[[9]]]; dcons2d10 -|
 SMSD[(Jed wgp cons2Loc), dCP[[10]]]; dcons2d11 -|
 SMSD[(Jed wgp cons2Loc), dCP[[11]]]; dcons2d12 -|
 SMSD[(Jed wgp cons2Loc), dCP[[12]]]; dcons2d13 -|
 SMSD[(Jed wgp cons2Loc), dCP[[13]]]; dcons2d14 -|
 SMSD[(Jed wgp cons2Loc), dCP[[14]]];
dcons2d15 -|
 SMSD[(Jed wgp cons2Loc), dCP[[15]]]; dcons2d16 -|
 SMSD[(Jed wgp cons2Loc), dCP[[16]]]; dcons2d17 -|
 SMSD[(Jed wgp cons2Loc), dCP[[17]]]; dcons2d18 -|
 SMSD[(Jed wgp cons2Loc), dCP[[18]]]; dcons2d19 -|
 SMSD[(Jed wgp cons2Loc), dCP[[19]]]; dcons2d20 -|
 SMSD[(Jed wgp cons2Loc), dCP[[20]]]; dcons2d21 -|
 SMSD[(Jed wgp cons2Loc), dCP[[21]]]; dcons2d22 -|
 SMSD[(Jed wgp cons2Loc), dCP[[22]]]; dcons2d23 -|
 SMSD[(Jed wgp cons2Loc), dCP[[23]]]; dcons2d24 -|
 SMSD[(Jed wgp cons2Loc), dCP[[24]]]; dcons2d25 -|
 SMSD[(Jed wgp cons2Loc), dCP[[25]]];

SMSEndIf[cons2, dcons2d1, dcons2d2, dcons2d3, dcons2d4, dcons2d5,
  dcons2d6, dcons2d7, dcons2d8, dcons2d9, dcons2d10, dcons2d11,
  dcons2d12, dcons2d13, dcons2d14,
  dcons2d15, dcons2d16, dcons2d17, dcons2d18, dcons2d19, dcons2d20,
  dcons2d21, dcons2d22, dcons2d23, dcons2d24, dcons2d25];
```

```
cons2BLoc |= (1 - ((lmax/b) MagDCz2 )^2);


SMSIf[cons2BLoc <= 0];
cons2B =| 0;
dcons2Bd1 =| 0; dcons2Bd2 =| 0; dcons2Bd3 \
=| 0; dcons2Bd4 =| 0; dcons2Bd5 \
=| 0; dcons2Bd6 =| 0; dcons2Bd7 \
=| 0; dcons2Bd8 =| 0; dcons2Bd9 \
=| 0; dcons2Bd10 =| 0; dcons2Bd11 \
=| 0; dcons2Bd12 =| 0; dcons2Bd13 \
=| 0; dcons2Bd14 =| 0;
dcons2Bd15 =| 0; dcons2Bd16 =| 0; \
dcons2Bd17 =| 0; dcons2Bd18 =| 0; \
dcons2Bd19 =| 0; dcons2Bd20 =| 0; \
dcons2Bd21 =| 0; dcons2Bd22 =| 0; \
dcons2Bd23 =| 0; dcons2Bd24 =| 0; \
dcons2Bd25 =| 0;
SMSElse[];
cons2B -| (Jed wgp cons2BLoc);
dcons2Bd1 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[1]]]; dcons2Bd2 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[2]]]; dcons2Bd3 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[3]]]; dcons2Bd4 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[4]]]; dcons2Bd5 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[5]]]; dcons2Bd6 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[6]]]; dcons2Bd7 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[7]]]; dcons2Bd8 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[8]]]; dcons2Bd9 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[9]]]; dcons2Bd10 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[10]]]; dcons2Bd11 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[11]]]; dcons2Bd12 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[12]]]; dcons2Bd13 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[13]]]; dcons2Bd14 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[14]]];
dcons2Bd15 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[15]]]; dcons2Bd16 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[16]]]; dcons2Bd17 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[17]]]; dcons2Bd18 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[18]]]; dcons2Bd19 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[19]]]; dcons2Bd20 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[20]]]; dcons2Bd21 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[21]]]; dcons2Bd22 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[22]]]; dcons2Bd23 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[23]]]; dcons2Bd24 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[24]]]; dcons2Bd25 -|
 SMSD[(Jed wgp cons2BLoc), eCP[[25]]];

SMSEndIf[cons2B, dcons2Bd1, dcons2Bd2, dcons2Bd3, dcons2Bd4,
  dcons2Bd5, dcons2Bd6, dcons2Bd7, dcons2Bd8, dcons2Bd9, dcons2Bd10,
  dcons2Bd11, dcons2Bd12, dcons2Bd13, dcons2Bd14,
  dcons2Bd15, dcons2Bd16, dcons2Bd17, dcons2Bd18, dcons2Bd19,
  dcons2Bd20, dcons2Bd21, dcons2Bd22, dcons2Bd23, dcons2Bd24,
  dcons2Bd25];

DCMDA |=
  SMSD[Cm1, dCP,
   "Dependency" -> {{α, dCP, SMSD[α, dCP]}, {v, dCP,
      SMSD[v, dCP]}}];
DCD1A |=
  TensorContract[TensorProduct[DCMDA, ϵ], {{3, 6}, {4, 7}}];
DCD2A |=
  TensorContract[TensorProduct[DCD1A, ϵ], {{1, 4}, {2, 5}}];
DCD3A |= t Jed wgp DCD2A ;

DCMDB |=
  SMSD[Cm2, eCP,
   "Dependency" -> {{α2, eCP, SMSD[α2, eCP]}, {v2, eCP,
      SMSD[v2, eCP]}}];
DCD1B |=
  TensorContract[TensorProduct[DCMDB, ϵ], {{3, 6}, {4, 7}}];
DCD2B |=
  TensorContract[TensorProduct[DCD1B, ϵ], {{1, 4}, {2, 5}}];
DCD3B |= t Jed wgp DCD2B ;

(* ------------------------- RETURN COMPLIANCE AND SENSITIVITIES
    ------------------------ \
*)
SMSIf[task == 1,
  SMSExport[ {comp, cons1, cons1B, cons2, cons2B, DCD3A[[1]],
     DCD3A[[2]], DCD3A[[3]], DCD3A[[4]], DCD3A[[5]], DCD3A[[6]],
     DCD3A[[7]], DCD3A[[8]], DCD3A[[9]], DCD3A[[10]], DCD3A[[11]],
     DCD3A[[12]], DCD3A[[13]], DCD3A[[14]], DCD3A[[15]], DCD3A[[16]],
     DCD3A[[17]], DCD3A[[18]], DCD3A[[19]], DCD3A[[20]], DCD3A[[21]],
     DCD3A[[22]], DCD3A[[23]], DCD3A[[24]], DCD3A[[25]], DCD3B[[1]],
     DCD3B[[2]], DCD3B[[3]], DCD3B[[4]], DCD3B[[5]], DCD3B[[6]],
     DCD3B[[7]], DCD3B[[8]], DCD3B[[9]], DCD3B[[10]], DCD3B[[11]],
     DCD3B[[12]], DCD3B[[13]], DCD3B[[14]], DCD3B[[15]], DCD3B[[16]],
     DCD3B[[17]], DCD3B[[18]], DCD3B[[19]], DCD3B[[20]], DCD3B[[21]],
     DCD3B[[22]], DCD3B[[23]], DCD3B[[24]], DCD3B[[25]], dcons1d1,
     dcons1d2, dcons1d3, dcons1d4, dcons1d5, dcons1d6, dcons1d7,
     dcons1d8, dcons1d9, dcons1d10, dcons1d11, dcons1d12, dcons1d13,
     dcons1d14,
     dcons1d15, dcons1d16, dcons1d17, dcons1d18, dcons1d19, dcons1d20,
      dcons1d21, dcons1d22, dcons1d23, dcons1d24, dcons1d25,
     dcons1Bd1, dcons1Bd2, dcons1Bd3, dcons1Bd4, dcons1Bd5, dcons1Bd6,
      dcons1Bd7, dcons1Bd8, dcons1Bd9, dcons1Bd10, dcons1Bd11,
     dcons1Bd12, dcons1Bd13, dcons1Bd14,
     dcons1Bd15, dcons1Bd16, dcons1Bd17, dcons1Bd18, dcons1Bd19,
     dcons1Bd20, dcons1Bd21, dcons1Bd22, dcons1Bd23, dcons1Bd24,
     dcons1Bd25, dcons2d1, dcons2d2, dcons2d3, dcons2d4, dcons2d5,
     dcons2d6, dcons2d7, dcons2d8, dcons2d9, dcons2d10, dcons2d11,
     dcons2d12, dcons2d13, dcons2d14,
     dcons2d15, dcons2d16, dcons2d17, dcons2d18, dcons2d19, dcons2d20,
      dcons2d21, dcons2d22, dcons2d23, dcons2d24, dcons2d25,
     dcons2Bd1, dcons2Bd2, dcons2Bd3, dcons2Bd4, dcons2Bd5, dcons2Bd6,
      dcons2Bd7, dcons2Bd8, dcons2Bd9, dcons2Bd10, dcons2Bd11,
     dcons2Bd12, dcons2Bd13, dcons2Bd14,
     dcons2Bd15, dcons2Bd16, dcons2Bd17, dcons2Bd18, dcons2Bd19,
     dcons2Bd20, dcons2Bd21, dcons2Bd22, dcons2Bd23, dcons2Bd24,
     dcons2Bd25}, RealOutput$$, "AddIn" -> True];
  ];
SMSEndDo[];

SMSWrite[];
```

B.2 AceFEM Code

```
<< AceFEM‘

(*-----Initialize Control Points----*)
nCPX = 5;
nCPY = 5;
xCP = {0, 10, 20, 30, 40, 0, 10, 20, 30, 40, 0, 10, 20, 30, 40, 0, 10,
    20, 30, 40, 0, 10, 20, 30, 40};

yCP = {0, 0, 0, 0, 0, 10, 10, 10, 10, 10, 20, 20, 20, 20, 20, 30, 30,
   30, 30, 30, 40, 40, 40, 40, 40};

(*Optimal Filament Angles from Step-1*)
α1 = 61.79
α2 = 14.30;
```

```
 (*Initial Design Variables*)
d1CP = ((-Sin[α1 Degree])*
    Flatten[xCP] + (Cos[α1 Degree])*Flatten[yCP])

d2CP = (-Sin[α2 Degree])*
   Flatten[xCP] + (Cos[α2 Degree])*Flatten[yCP]


xyCP = AppendTo[xCP, yCP];
xyCP = Flatten[xyCP];


ϵp = 10;

tt = AbsoluteTime[];


hx = 10;
hy = 10;
minX = 10;
minY = 10;

L1 = 10;
L2 = 30;
W1 = 10;
W2 = 30;
αg1 = 0;
Compl[d1_?NumericQ, d2_?NumericQ, d3_?NumericQ, d4_?NumericQ,
   d5_?NumericQ, d6_?NumericQ, d7_?NumericQ, d8_?NumericQ,
   d9_?NumericQ, d10_?NumericQ, d11_?NumericQ, d12_?NumericQ,
   d13_?NumericQ, d14_?NumericQ, d15_?NumericQ, d16_?NumericQ,
   d17_?NumericQ, d18_?NumericQ, d19_?NumericQ, d20_?NumericQ,
   d21_?NumericQ, d22_?NumericQ, d23_?NumericQ, d24_?NumericQ,
   d25_?NumericQ, e1_?NumericQ, e2_?NumericQ, e3_?NumericQ,
   e4_?NumericQ, e5_?NumericQ, e6_?NumericQ, e7_?NumericQ,
   e8_?NumericQ, e9_?NumericQ, e10_?NumericQ, e11_?NumericQ,
   e12_?NumericQ, e13_?NumericQ, e14_?NumericQ, e15_?NumericQ,
   e16_?NumericQ, e17_?NumericQ, e18_?NumericQ, e19_?NumericQ,
   e20_?NumericQ, e21_?NumericQ, e22_?NumericQ, e23_?NumericQ,
   e24_?NumericQ, e25_?NumericQ] := (
   d1CP = {d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13,
     d14, d15, d16, d17, d18, d19, d20, d21, d22, d23, d24, d25};
   d2CP = {e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12, e13,
     e14, e15, e16, e17, e18, e19, e20, e21, e22, e23, e24, e25};

   SMTInputData[(*"Threads"-> 1*)];
```

```
(* ----Boundary Conditions ----*)
SMTAddEssentialBoundary[Point[{L1, W1}], 2 -> 0];
SMTAddEssentialBoundary[Line[{{L2, W1}, {L2, W2}}], 1 -> 0];

SMTAddNaturalBoundary[Point[{L2, W2}], 2 -> -1000];

(* ----Send Information to AceGen Code ----*)
SMTAddDomain["quad",
 "M1FE50nu0.9Del0.1", {"Em *" -> 200, "\[Nu] *" -> 0.3, "t *" -> 1,
   "nCPX" -> 5, "nCPY" -> 5, "hx" -> 10, "hy" -> 10, "minX" -> 10,
  "minY" -> 10, "XY1" -> xyCP[[1]], "XY2" -> xyCP[[2]],
  "XY3" -> xyCP[[3]], "XY4" -> xyCP[[4]], "XY5" -> xyCP[[5]],
  "XY6" -> xyCP[[6]], "XY7" -> xyCP[[7]], "XY8" -> xyCP[[8]],
  "XY9" -> xyCP[[9]], "XY10" -> xyCP[[10]], "XY11" -> xyCP[[11]],
  "XY12" -> xyCP[[12]], "XY13" -> xyCP[[13]], "XY14" -> xyCP[[14]],
   "XY15" -> xyCP[[15]], "XY16" -> xyCP[[16]],
  "XY17" -> xyCP[[17]], "XY18" -> xyCP[[18]], "XY19" -> xyCP[[19]],
   "XY20" -> xyCP[[20]], "XY21" -> xyCP[[21]],
  "XY22" -> xyCP[[22]], "XY23" -> xyCP[[23]], "XY24" -> xyCP[[24]],
   "XY25" -> xyCP[[25]], "XY26" -> xyCP[[26]],
  "XY27" -> xyCP[[27]], "XY28" -> xyCP[[28]], "XY29" -> xyCP[[29]],
   "XY30" -> xyCP[[30]], "XY31" -> xyCP[[31]],
  "XY32" -> xyCP[[32]], "XY33" -> xyCP[[33]], "XY34" -> xyCP[[34]],
   "XY35" -> xyCP[[35]], "XY36" -> xyCP[[36]],
  "XY37" -> xyCP[[37]], "XY38" -> xyCP[[38]], "XY39" -> xyCP[[39]],
   "XY40" -> xyCP[[40]], "XY41" -> xyCP[[41]],
  "XY42" -> xyCP[[42]], "XY43" -> xyCP[[43]], "XY44" -> xyCP[[44]],
   "XY45" -> xyCP[[45]], "XY46" -> xyCP[[46]],
  "XY47" -> xyCP[[47]], "XY48" -> xyCP[[48]], "XY49" -> xyCP[[49]],
   "XY50" -> xyCP[[50]], "D1" -> d1CP[[1]], "D2" -> d1CP[[2]],
  "D3" -> d1CP[[3]], "D4" -> d1CP[[4]], "D5" -> d1CP[[5]],
  "D6" -> d1CP[[6]], "D7" -> d1CP[[7]], "D8" -> d1CP[[8]],
  "D9" -> d1CP[[9]], "D10" -> d1CP[[10]], "D11" -> d1CP[[11]],
  "D12" -> d1CP[[12]], "D13" -> d1CP[[13]], "D14" -> d1CP[[14]],
  "D15" -> d1CP[[15]], "D16" -> d1CP[[16]], "D17" -> d1CP[[17]],
  "D18" -> d1CP[[18]], "D19" -> d1CP[[19]], "D20" -> d1CP[[20]],
  "D21" -> d1CP[[21]], "D22" -> d1CP[[22]], "D23" -> d1CP[[23]],
  "D24" -> d1CP[[24]], "D25" -> d1CP[[25]], "E1" -> d2CP[[1]],
  "E2" -> d2CP[[2]], "E3" -> d2CP[[3]], "E4" -> d2CP[[4]],
  "E5" -> d2CP[[5]], "E6" -> d2CP[[6]], "E7" -> d2CP[[7]],
  "E8" -> d2CP[[8]], "E9" -> d2CP[[9]], "E10" -> d2CP[[10]],
  "E11" -> d2CP[[11]], "E12" -> d2CP[[12]], "E13" -> d2CP[[13]],
  "E14" -> d2CP[[14]], "E15" -> d2CP[[15]], "E16" -> d2CP[[16]],
  "E17" -> d2CP[[17]], "E18" -> d2CP[[18]], "E19" -> d2CP[[19]],
  "E20" -> d2CP[[20]], "E21" -> d2CP[[21]], "E22" -> d2CP[[22]],
     "E23" -> d2CP[[23]], "E24" -> d2CP[[24]], "E25" -> d2CP[[25]],
     "αg1" -> αg1}];

   (* ----Mesh ----*)
   SMTMesh["quad",
    "Q1", {32, 32}, {{{L1, W1}, {L2, W1}}, {{L1, W2}, {L2, W2}}}];

   (* ----Perform FEA ----*)
   SMTAnalysis[];

   Do[
        SMTNextStep["\[Lambda]" -> 1];
        While[
     SMTConvergence[10^-8, 10]
     , SMTNewtonIteration[];
     ];
        , {i, 1, 1}];

   c = SMTTask["Compliance"];
   comp = c[[1]];
   cons1A = εp (Max[c[[2]], 0])^2;
   cons1B = εp (Max[c[[3]], 0])^2;
   cons2A = εp (Max[c[[4]], 0])^2;
   cons2B = εp (Max[c[[5]], 0])^2;
   zeros = ConstantArray[0, 25];
   CompgradA = c[[6 ;; 30]];
   CompgradB = c[[31 ;; 55]];
   Compgrad = Flatten[AppendTo[CompgradA, CompgradB]];
   cons1Agrad = c[[56 ;; 80]];
   cons1Agrad = Flatten[AppendTo[cons1Agrad, zeros]];
   cons1Bgrad = c[[81 ;; 105]];
   cons1Bgrad = Flatten[AppendTo[zeros, cons1Bgrad]];
   zeros = ConstantArray[0, 25];
   cons2Agrad = c[[106 ;; 130]];
   cons2Agrad = Flatten[AppendTo[cons2Agrad, zeros]];
   cons2Bgrad = c[[131 ;; 155]];
   cons2Bgrad = Flatten[AppendTo[zeros, cons2Bgrad]];

   (* ----Return Penalized Cost Function ----*)
   Return[comp + cons1A + cons1B +cons2A + cons2B];
   );

dCompl[d1_?NumericQ, d2_?NumericQ, d3_?NumericQ, d4_?NumericQ,
   d5_?NumericQ, d6_?NumericQ, d7_?NumericQ, d8_?NumericQ,
   d9_?NumericQ, d10_?NumericQ, d11_?NumericQ, d12_?NumericQ,
   d13_?NumericQ, d14_?NumericQ, d15_?NumericQ, d16_?NumericQ,
   d17_?NumericQ, d18_?NumericQ, d19_?NumericQ, d20_?NumericQ,
   d21_?NumericQ, d22_?NumericQ, d23_?NumericQ, d24_?NumericQ,
   d25_?NumericQ, e1_?NumericQ, e2_?NumericQ, e3_?NumericQ,
   e4_?NumericQ, e5_?NumericQ, e6_?NumericQ, e7_?NumericQ,
   e8_?NumericQ, e9_?NumericQ, e10_?NumericQ, e11_?NumericQ,
   e12_?NumericQ, e13_?NumericQ, e14_?NumericQ, e15_?NumericQ,
   e16_?NumericQ, e17_?NumericQ, e18_?NumericQ, e19_?NumericQ,
   e20_?NumericQ, e21_?NumericQ, e22_?NumericQ, e23_?NumericQ,
   e24_?NumericQ, e25_?NumericQ] := (

   (* ----Return Sensitivity ----*)
   grad = -Compgrad + (2 ϵp c[[
        2]] cons1Agrad) + (2 ϵp c[[
        3]] cons1Bgrad) + (2 ϵp c[[
        4]] cons2Agrad) + (2 ϵp c[[5]] cons2Bgrad);

   Return[grad];

   );

(* ----Perform Optimization ----*)
sol = Reap[
  FindMinimum[{Compl[d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11,
     d12, d13, d14, d15, d16, d17, d18, d19, d20, d21, d22, d23, d24,
     d25, e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12, e13, e14,
      e15, e16, e17, e18, e19, e20, e21, e22, e23, e24, e25] }, {{d1,
     d1CP[[1]]}, {d2, d1CP[[2]]}, {d3, d1CP[[3]]}, {d4,
     d1CP[[4]]}, {d5, d1CP[[5]]}, {d6, d1CP[[6]]}, {d7,
     d1CP[[7]]}, {d8, d1CP[[8]]}, {d9, d1CP[[9]]}, {d10,
     d1CP[[10]]}, {d11, d1CP[[11]]}, {d12, d1CP[[12]]}, {d13,
     d1CP[[13]]}, {d14, d1CP[[14]]}, {d15, d1CP[[15]]}, {d16,
     d1CP[[16]]}, {d17, d1CP[[17]]}, {d18, d1CP[[18]]}, {d19,
     d1CP[[19]]}, {d20, d1CP[[20]]}, {d21, d1CP[[21]]}, {d22,
     d1CP[[22]]}, {d23, d1CP[[23]]}, {d24, d1CP[[24]]}, {d25,
     d1CP[[25]]}, {e1, d2CP[[1]]}, {e2, d2CP[[2]]}, {e3,
     d2CP[[3]]}, {e4, d2CP[[4]]}, {e5, d2CP[[5]]}, {e6,
     d2CP[[6]]}, {e7, d2CP[[7]]}, {e8, d2CP[[8]]}, {e9,
     d2CP[[9]]}, {e10, d2CP[[10]]}, {e11, d2CP[[11]]}, {e12,
     d2CP[[12]]}, {e13, d2CP[[13]]}, {e14, d2CP[[14]]}, {e15,
     d2CP[[15]]}, {e16, d2CP[[16]]}, {e17, d2CP[[17]]}, {e18,
     d2CP[[18]]}, {e19, d2CP[[19]]}, {e20, d2CP[[20]]}, {e21,
     d2CP[[21]]}, {e22, d2CP[[22]]}, {e23, d2CP[[23]]}, {e24,
     d2CP[[24]]}, {e25, d2CP[[25]]}},
   Gradient :>
    dCompl[d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13,
     d14, d15, d16, d17, d18, d19, d20, d21, d22, d23, d24, d25, e1,
     e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12, e13, e14, e15,
     e16, e17, e18, e19, e20, e21, e22, e23, e24, e25],
   MaxIterations -> 2500, AccuracyGoal -> {2, 2},
   Method -> {"QuasiNewton",
     "StepControl" -> {"LineSearch", "MaxRelativeStepSize"  -> 0.1}},
   StepMonitor :>
    Sow[{Compl[d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13,
        d14, d15, d16, d17, d18, d19, d20, d21, d22, d23, d24, d25,
       e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12, e13, e14,
       e15, e16, e17, e18, e19, e20, e21, e22, e23, e24, e25]}]]]

(AbsoluteTime[] - tt)
```

Appendix C

Pseudo-Code for Filament Deposition Algorithm

The pseudo-code of the orientation-based filament deposition algorithm is presented below.

```
%%%%%%%%%% Initial Operations %%%%%%%%%%

1. Define Input parameters $w$, $h$, $ol$, $V$, $tstep$
   % w = extrusion width
   h=layer height
   ol = overlap in percentage
   Vm = velocity magnitude, tstep = time step
   Vm and tstep are used during numerical integration to define the length
   $d_L$ between two points
   Overlap values correspond to the minimum gap $G_p$

2. Define Seeds % Imported seed points from FE mesh

3. Define xyNodes % Imported internal nodes of FE mesh

4. Define u = Vm*S1(:,1), v = Vm*S1(:,2)
   %Import Bidirectional Principal Stress directions at each node
   % Principal directions S1 are unit vectors. They are multiplied with velocity magnitude Vm to yield velocity components u and v
```

```
5. Define BNodes
   % Imported boundary Nodes from FE mesh
   % Assign them to have zero velocities. This step is useful for terminating the streamlines at the design boundaries

   Copy the positions and velocities of boundary nodes into xyNodes and u-v respectively

6. Define U, V = Interpolate(xyNodes,u), Interpolate(xyNodes,v)
   % Velocity components available throughout the design domain after interpolating nodal velocities

   %%%%%%%%%% Streamline generation code %%%%%%%%%%

7. Initialize XY=[] % a Matrix data structure to hold streamline information
```

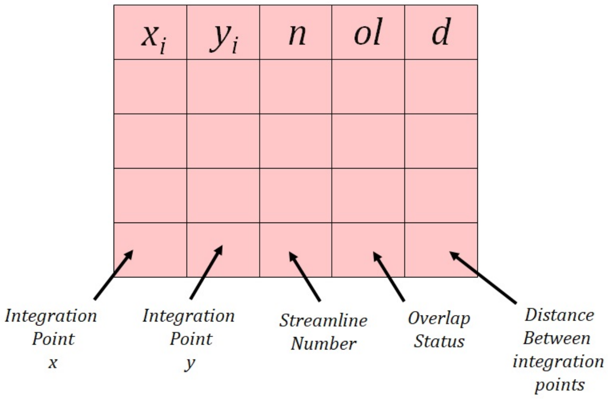


FIGURE C.1: XY Data Structure

```
   Initialize indices=[] % A matrix to hold the starting and ending indices for streamlines

   Define flagForward=1 flagBackward=1
   % Takes two values 0 and 1. Stops or continues streamline generation in two directions
```

```
Define ForwardFlip=0  BackwardFlip=0
% Takes two values 0 and 1. Controls flipping of vectors in two directions
For each seed point j = 1 to length(Seeds)
    While 1
        % Loop to generate streamlines in both forward and backward directions. Forward and backward directions correspond to downflow and upflow directions of a particle

        % Euler's rule :
        $\mathbf{x}_{i+1} = \mathbf{x}_i + V(\mathbf{x}_i)\Delta t$ - Forward direction
        $\mathbf{x}_{i+1} = \mathbf{x}_i - V(\mathbf{x}_i)\Delta t$ - backward direction

        % Forward Direction
        if flagForward==1
            if ForwardFlip==1
                Flip the current velocity % -ve velocity at current point
            endif
            Generate streamline in Forward Direction (using Euler's rule)
            Find the velocity at the new point : U(@new point), V(@new point)
            if the new velocity is not in the forward direction
                set ForwardFlip=1 % to flip velocity at next point to move in forward direction
            else
                set ForwardFlip=0
            endif
            if new point reaches design boundaries ($U \approx 0$ && $V \approx 0$)
                set flagForward=0
                % to stop streamline generation in forward direction
            elseif new point comes near to backward direction points
                set flagForward=0
                % to stop streamline generation in forward direction and prevent loops
            endif
        endif

        % Backward Direction
        if flagBackward==1
            if BackwardFlip==1
                Flip the current velocity % -ve velocity at current point
            endif
            Generate streamline in backward Direction (using Euler's
            rule)
            Find the velocity at the new point : U(@new point), V(@new
            point)
            if the new velocity is not in the backward direction
                set BackwardFlip=1 % to flip velocity at next point to move
                in backward direction
            else
                set BackwardFlip=0
            endif
            if new point reaches design boundaries ($U \approx 0$ && $V \approx 0$)
                set flagBackward=0
                % to stop streamline generation in backward direction
            elseif new point comes near to forward direction points
            % to stop streamline generation in backward direction and
            prevent loops
                set flagBackward=0
            endif
        endif
        if flagForward==1 && flagBackward==1
            Break;
        else
            Compute and store distance between the subsequent integration points
        endif
    % End of loop- Repeat the streamline generation process for new
    point
```

```
        endWhile
        Copy the forward and backward integration points into XY
        Copy the starting and ending index for the streamline in XY to indices
        Set streamline number XY(indices(Start,j):indices(End,j),3)=j
        Set overlap status XY(indices(Start,j):indices(End,j),4)=0
        Set XY(indices(Start,j):indices(End,j),5) = the distance between integration points
    endFor

    %%%%%%%%% Control Density of streamlines %%%%%%%%%
    % Three overlap status for XY data structure
    % Untreated Streamlines : XY(indices(Start,n):indices(End,n),3)=0
    % Removed Streamlines : XY(indices(Start,n):indices(End,n),3)=1
    % Streamline chosen to be filament trajectory :
    XY(indices(Start,n):indices(End,n),3)=-1

    Define IndStr = Indices with XY(:,4)==0
    Define  i=1 % Choose streamline number
    While IndStr is not empty
        Delaunay Triangulation for XY(IndStr,1:2)
        Find triangles TR connected to the points of current streamline i
        Find all the vertices (integration points) of the triangles TR
        Find the streamline numbers $n_1$ of these vertices from data structures XY and indices
        Compare all integration points from streamlines i and $n_1$
        if connected streamlines $n_1$ and i are collinear
            set the overlap status for all points of these streamlines to 1
            $XY(indices(Start,n_1) : indices(End, n_1), 4) = 1$
        else
            Find the radius $r$ of the circumcircles for all triangles $TR$
            Choose the triangles having $r \leq 0.5w(1-0.01ol)$
            Find the corresponding vertices
            Find the streamline numbers $n_1$ of these vertices from data structures $XY$ and $indices$
            if $n_1$ is not empty
            % streamlines overlap
                set the overlap status for all points of these streamlines to 1
            $XY(indices(Start,n_1) : indices(End,n_1),4) = 1$
            else
            % No streamlines overlap. Streamline $i$ can be chosen as a filament trajectory. Another streamline must be chosen as $i$ for next iteration
                set the overlap status to -1 for all points of streamlines $i$
                $XY(indices(Start,i):indices(End,i),4)$=-1

                Find minimum radius of the circumcircle $r_{min}$ in the $TR$
                Choose triangle with $r_{min}$
                Find the corresponding vertices
                Find the streamline numbers $n_1$ of these vertices from data structures $XY$ and $indices$
                For next iteration assign, $i$ = any one streamline from $n_1$
            endif
        endif
        Set $IndStr$ = Indices with XY(:,4)==0
    % End of While loop. Repeat the process for new $i$ and new triangulation data set. The process continues until all streamlines are either removed or chosen as the filament trajectory
    endWhile
    %%%%%% Following section to be included only if $ol \geq 0$ %%%%%%

    %%%%%%%%%% Insert Discontinuous Streamlines %%%%%%%%%%
    % Initialize a similar data structure XYBr for discontinuous streamlines
```

```
Define IndFil = Indices with XY(:,4)==-1

Define XYT = XY(IndFil,1:2)

While 1

    Delaunay Triangulation for XYT
    Obtain all triangles TR
    Find maximum radius $r_{max}$ of the circumcircles for TR
    if$r_{max} \leq 0.5w$
        % Break out of the loop when there are no more large enough
        gaps to place new seeds
        Break;
    endif
    Choose the centre of the corresponding circle as the new seed point.
    % Generate streamline from new seed point
    Set flagForward=1  flagBackward=1
    Set ForwardFlip=0  BackwardFlip=0
    While 1
        % Loop to generate streamlines in both forward and backward
        directions. Forward and backward directions correspond to
        downflow and upflow directions of a particle

        % Euler's rule :
        $\mathbf{x}_{i+1} = \mathbf{x}_i + V(\mathbf{x}_i)\Delta t$ - Forward direction
        $\mathbf{x}_{i+1} = \mathbf{x}_i - V(\mathbf{x}_i)\Delta t$ - backward direction

        % Forward Direction
        if flagForward==1
            if ForwardFlip==1
              Flip the current velocity % -ve velocity at current point
            endif
            Generate streamline in Forward Direction (using Euler forward method)
            Find the velocity at the new point : U(@new point), V(@new
            point)
        if the new velocity is not in the forward direction
            set ForwardFlip=1 % to flip velocity at next point to move
            in forward direction
        else
            set ForwardFlip=0
        endif
        if new point reaches design boundaries ($U \approx 0$ && $V \approx 0$)
            set flagForward=0
            % to stop streamline generation in forward direction
        elseif new point comes near to backward direction points
            set flagForward=0
            % to stop streamline generation in forward direction and
            prevent loops
        endif
    endif

    % Backward Direction
    if flagBackward==1
        if BackwardFlip==1
            Flip the current velocity % -ve velocity at current point
        endif
        Generate streamline in backward Direction (using Euler's
        rule)
        Find the velocity at the new point : U(@new point), V(@new
        point)
        if the new velocity is different from backward direction
            set BackwardFlip=1 % to flip velocity at next point to move
            in backward direction
        else
            set BackwardFlip=0
        endif
        if new point reaches design boundaries ($U \approx 0$ && $V \approx 0$)
            set flagBackward=0
            % to stop streamline generation in backward direction
        elseif new point comes near to forward direction points
```

```
            % to stop streamline generation in backward direction and prevent loops
              set flagBackward=0
            endif
        endif
        if flagForward==1 && flagBackward==1
            Break;
        else
            Compute and store the distance between the subsequent integration points
        endif
    % End of loop- Repeat the streamline generation process for new point
    endWhile
    Add the new streamline points to XYT and create Delaunay Triangulation
    Find the triangles attached to the new streamline
    Find the radius $r$ of the circumcircles for these triangles
    Choose the triangles having $r \leq 0.5w(1-0.01ol)$
    Find the corresponding vertices (or integration points)
    Among the vertices choose only the points in the current streamline
    Remove these points from triangulation data set $XYT$
    Copy the new streamline into the $XYBr$ and set the overlap status to 1 for overlapping points
    % End of loop- Repeat the process for a new triangulation data set with additional discontinuous streamlines
endWhile
%%%%%%%%% Path Linking and G-Code Generation %%%%%%%%%

Define G-code inputs - $n_L$, $NPrSpeedXY$, $NPrSpeedZ$, $PrSpeed$, $ExSpeed$, $r_L$, $filD$, $tol$, $X0$, $Y0$ and $Z0$
```

```
% $n_L$ - number of layers, NPrSpeedXY - non-printing speed for XY movement, NPrSpeedZ - non-printing speed for Z movement,PrSpeed - printing speed, ExSpeed - extrusion speed, $r_L$ - retraction length, filD - raw filament diameter, tol - tolerance for path linking, X0, Y0 and Z0 - origin point

Define extruded cross-section $A = wh$

Define raw filament cross-section filA = $\Pi/4 filD^2$

Redefine XY(:,1) = XY(:,1)+X0 ; XY(:,2) = XY(:,2)+Y0
Redefine XYBr(:,1) = XYBr(:,1)+X0 ; XYBr(:,2) = XYBr(:,2)+Y0
% Redefine the streamline paths according to the origin point

Combine non-overlapping streamlines from XY and XYBr (if $ol \geq 0$)

Define Cindices
% Indices of non-overlapping streamlines chosen as filament trajectories

Define EndPointsX and EndPointsY
% The endpoints of the non-overlapping streamlines are stored separately

Open text file for writing the G-Code

Write instructions for printing the raft

Write instructions for heating the filament extruder
j=1 % Streamline number
```

```
Initialize e=1 % Has value 1 or 2 depending upon first or last endpoint of a filament

Initialize $d_{min}$ = 1000 % $d_{min}$ is the closest distance of an unprinted filament endpoint to the last printed point. Intially, a large value is assigned

For i=1 to $n_L$
% Loop instructions for $n_L$ layers
    Write instructions for moving the table height defined as H0 + i h
    Write instructions for printing perimeters
    For k = 1:length(EndPointsX(1,:))
        if e==1
            if $d_{min} \leq tol$
                Write instructions for retracting filament
                Write instructions for moving the nozzle to the first point in the next filament XY(Cindices(1,j),1:2)
                Write instructions for pushing the filament to nozzle tip
                Write instructions to print non-overlapping filament from endpoint 1 to endpoint 2 XY(Cindices(1,j):Cindices(2,j),1:2)
            else
                Write instructions to print non-overlapping filament from endpoint 1 to endpoint 2 XY(Cindices(1,j):Cindices(2,j),1:2)
        else
            if $d_{min} \leq tol$
                Write instructions for retracting filament
                Write instructions for moving the nozzle to the second point in the next filament XY(Cindices(2,j),1:2)
                Write instructions for push the filament to nozzle tip
                Write instructions to print non-overlapping filament (from endpoint 2 to endpoint 1) XY(Cindices(2,j):-1:Cindices(1,j),1:2)
            else
                Write instructions to print non-overlapping filament from
                endpoint 2 to endpoint 1
                XY(Cindices(2,j):-1:Cindices(2,j),1:2)
        endif
        % The printing instructions for filament include the X-Y positions and the distance of raw material to be extruded at each step to produce a filament of width w and height h. It can be calculated using extruded volume $v = A\ d$ and cross-sectional area of the raw filament where, $d$ is the distance between streamline integration points
        Find the distance of all non-printed filament endpoints from
        the last point printed
        % Dist(XY(Cindices(e,j),1:2), [EndPointsX EndpointsY])
        Find the distance $d_{min}$ of the endpoint closest to the point last
        printed point

        if closest point is the first endpoint of a filament
            Set e=1
        else % the closest point is the second endpoint of a filament
            Set e=2
        endif
        Assign j = the streamline number corresponding to the closest
        point found
    % End of For loop - Repeat instructions for printing next filament
    endFor

% End of For loop - Repeat instructions for printing next layer

endFor

Write instructions for retracting filament


Write instructions for homing the nozzle position


Write instructions for cooling down extruder
```

```
    Write instructions for lowering table

8. Save and close the text file

    %%%%%%%%% End of Pseudo Code %%%%%%%%%
```

Bibliography

Acar, Pinar et al. (2016). “Optimization of spatially varying fiber paths for a symmetric laminate with a circular cutout under remote uniaxial tension”. In: *SAE International Journal of Materials and Manufacturing* 9.1, pp. 75–80.

Ahn, Sung-Hoon et al. (2002). “Anisotropic material properties of fused deposition modeling ABS”. In: *Rapid prototyping journal* 8.4, pp. 248–257.

Alaimo, Gianluca et al. (2017). “Influence of meso-structure and chemical composition on FDM 3D-printed parts”. In: *Composites Part B: Engineering* 113, pp. 371–380.

Ali, MM and WX Zhu (2013). “A penalty function-based differential evolution algorithm for constrained global optimization”. In: *Computational Optimization and Applications* 54.3, pp. 707–739.

Aliheidari, Nahal et al. (2017). “Fracture resistance measurement of fused deposition modeling 3D printed polymers”. In: *Polymer Testing* 60, pp. 94–101.

Amstutz, Samuel (2010). “A penalty method for topology optimization subject to a pointwise state constraint”. In: *ESAIM: Control, Optimisation and Calculus of Variations* 16.3, pp. 523–544.

Amstutz, Samuel, AA Novotny, and EA de Souza Neto (2012). “Topological derivative-based topology optimization of structures subject to Drucker–Prager stress constraints”. In: *Computer Methods in Applied Mechanics and Engineering* 233, pp. 123–136.

Amstutz, Samuel and Antonio A Novotny (2010). “Topological optimization of structures subject to von Mises stress constraints”. In: *Structural and Multidisciplinary Optimization* 41.3, pp. 407–420.

ASTM (2012). “Standard terminology for additive manufacturing technologies”. In: *ASTM International F2792-12a*.

Attaran, Mohsen (2017). “The rise of 3-D printing: The advantages of additive manufacturing over traditional manufacturing”. In: *Business Horizons* 60.5, pp. 677–688.

Bartels, Richard H, John C Beatty, and Brian A Barsky (1995). *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann.

Blok, Lourens G et al. (2018). "An investigation into 3D printing of fibre reinforced thermoplastic composites". In: *Additive Manufacturing* 22, pp. 176–186.

Blom, Adriana W et al. (2008). "Design of variable-stiffness conical shells for maximum fundamental eigenfrequency". In: *Computers & structures* 86.9, pp. 870–878.

Bourell, David L, Ming C Leu, and David W Rosen (2009). "Roadmap for additive manufacturing: identifying the future of freeform processing". In: *The University of Texas at Austin, Austin, TX*, pp. 11–15.

Brampton, Christopher J, K Chauncey Wu, and H Alicia Kim (2015). "New optimization method for steered fiber composites using the level set method". In: *Structural and Multidisciplinary Optimization* 52.3, pp. 493–505.

Brenken, Bastian et al. (2018). "Fused filament fabrication of fiber-reinforced polymers: A review". In: *Additive Manufacturing* 21, pp. 1–16.

Brett Champion, Adam Strzebonski (2008). *Constrained Optimization*. https://library.wolfram.com/infocenter/ID/8506/, Last Visited: 26-11-2021.

Bruijn, Ariadna Chueca de, Giovanni Gómez-Gras, and Marco A Pérez (2020). "Mechanical study on the impact of an effective solvent support-removal methodology for FDM Ultem 9085 parts". In: *Polymer Testing* 85, p. 106433.

Bruyneel, Michael (2011). "SFP—a new parameterization based on shape functions for optimal material selection: application to conventional composite plies". In: *Structural and Multidisciplinary Optimization* 43.1, pp. 17–27.

Calignano, Flaviana et al. (2017). "Overview on additive manufacturing technologies". In: *Proceedings of the IEEE* 105.4, pp. 593–612.

Casavola, Caterina et al. (2016). "Orthotropic mechanical properties of fused deposition modelling parts described by classical laminate theory". In: *Materials & design* 90, pp. 453–458.

Catapano, Anita, Boris Desmorat, and Paolo Vannucci (2015). "Stiffness and strength optimization of the anisotropy distribution for laminated structures". In: *Journal of Optimization Theory and Applications* 167.1, pp. 118–146.

Catapano, Anita et al. (2019). "Rapid prototyping of variable angle-tow composites". In: *Aerotecnica Missili & Spazio* 98.4, pp. 257–271.

Chacón, JM et al. (2017). "Additive manufacturing of PLA structures using fused deposition modelling: Effect of process parameters on mechanical properties and their optimal selection". In: *Materials & Design* 124, pp. 143–157.

Choi, Kyung K and Nam-Ho Kim (2004). *Structural sensitivity analysis and optimization 1: linear systems*. Springer Science & Business Media.

Christensen, Peter W and Anders Klarbring (2008). *An introduction to structural optimization*. Vol. 153. Springer Science & Business Media.

Cotteleer, Mark and Jim Joyce (2014). "3D opportunity: Additive manufacturing paths to performance, innovation, and growth". In: *Deloitte Review* 14, pp. 5–19.

Creusen, Frederic (2019). "Micro-scale computational analysis of Fused Filament Fabricated Materials: Experimentally-validated Finite Element Analyses of Representative Volume Elements". PhD thesis. Delft University of Technology (TU Delft).

Cuan-Urquizo, Enrique and Atul Bhaskar (2018). "Flexural elasticity of woodpile lattice beams". In: *European Journal of Mechanics-A/Solids* 67, pp. 187–199.

Cuan-Urquizo, Enrique, S Yang, and A Bhaskar (2015). "Mechanical characterisation of additively manufactured material having lattice microstructure". In: *IOP Conference Series: Materials Science and Engineering*. Vol. 74. 1. IOP Publishing, p. 012004.

Cuan-Urquizo, Enrique et al. (2019). "Characterization of the mechanical properties of FFF structures and materials: A review on the experimental, computational and theoretical approaches". In: *Materials* 12.6, p. 895.

Cuellar, Juan Sebastian et al. (2018). "Additive manufacturing of non-assembly mechanisms". In: *Additive Manufacturing* 21, pp. 150–158.

Curry, Haskell B (1944). "The method of steepest descent for non-linear minimization problems". In: *Quarterly of Applied Mathematics* 2.3, pp. 258–261.

Davidon, W C (1959). *Variable metric method for minimization*. Tech. rep. Argonne National Laboratory.

Dawoud, Michael, Iman Taha, and Samy J Ebeid (2016). "Mechanical behaviour of ABS: An experimental study using FDM and injection moulding techniques". In: *Journal of Manufacturing Processes* 21, pp. 39–45.

Deng, Xiaohu et al. (2018). "Mechanical properties optimization of poly-ether-ether-ketone via fused deposition modeling". In: *Materials* 11.2, p. 216.

Druckwege (2022). *Druckwege 3D printing*. https://druckwege.de/en/home-en/technology/fused-deposition-modelling-fdm, Last Visited: 01-04-2022.

Durgun, Ismail and Rukiye Ertan (2014). “Experimental investigation of FDM process for improvement of mechanical properties and production cost”. In: *Rapid Prototyping Journal* 20.3, pp. 228–235.

Duvaut, G et al. (2000). “Optimization of fiber reinforced composites”. In: *Composite Structures* 48.1-3, pp. 83–89.

Es-Said, OS et al. (2000). “Effect of layer orientation on mechanical properties of rapid prototyped samples”. In: *Materials and Manufacturing Processes* 15.1, pp. 107–122.

Esposito, L et al. (2019). “Topology optimization-guided stiffening of composites realized through automated fiber placement”. In: *Composites Part B: Engineering* 164, pp. 309–323.

Fayazbakhsh, Kazem, Mobina Movahedi, and Jordan Kalman (2019). “The impact of defects on tensile properties of 3D printed parts manufactured by fused filament fabrication”. In: *Materials Today Communications* 18, pp. 140–148.

Fernandez, Felipe et al. (2019). “Optimal design of fiber reinforced composite structures and their direct ink write fabrication”. In: *Computer Methods in Applied Mechanics and Engineering* 353, pp. 277–307.

Fletcher, Roger (2013). *Practical methods of optimization*. John Wiley & Sons.

Gämperle, Roger, Sibylle D Müller, and Petros Koumoutsakos (2002). “A parameter study for differential evolution”. In: *Advances in intelligent systems, fuzzy systems, evolutionary computation* 10.10, pp. 293–298.

Gaynor, Andrew Thomas (2015). “Topology optimization algorithms for additive manufacturing”. PhD thesis. Johns Hopkins University.

Gebisa, Aboma Wagari and Hirpa G Lemu (2018). “Investigating effects of Fused-Deposition Modeling (FDM) processing parameters on flexural properties of ULTEM 9085 using designed experiment”. In: *Materials* 11.4, p. 500.

Ghiasi, Hossein, Damiano Pasini, and Larry Lessard (2009). “Optimum stacking sequence design of composite materials Part I: Constant stiffness design”. In: *Composite Structures* 90.1, pp. 1–11. ISSN: 0263-8223.

Ghiasi, Hossein et al. (2010). “Optimum stacking sequence design of composite materials Part II: Variable stiffness design”. In: *Composite structures* 93.1, pp. 1–13.

Giorgio, Ivan, Alessandro Ciallella, and Daria Scerrato (2020). “A study about the impact of the topological arrangement of fibers on fiber-reinforced composites: some guidelines aiming at the development of new ultra-stiff and ultra-soft metamaterials". In: *International Journal of Solids and Structures* 203, pp. 73–83.

Haldar, Ayan et al. (2018). "Thermally induced multistable configurations of variable stiffness composite plates: Semi-analytical and finite element investigation". In: *Composite Structures* 183, pp. 161–175.

Hart, Kevin R and Eric D Wetzel (2017). "Fracture behavior of additively manufactured acrylonitrile butadiene styrene (ABS) materials". In: *Engineering Fracture Mechanics* 177, pp. 1–13.

Hodgson, G, A Ranellucci, and J Moe (2015). *Slic3r manual*. Tech. rep. https://manual.slic3r.org/.

Hoglund, R and DE Smith (2016). "Continuous fiber angle topology optimization for polymer fused filament fabrication". In: *Proceedings of the 27th annual international solid freeform fabrication symposium, austin, TX, USA*, pp. 8–10.

Honda, Shinya and Yoshihiro Narita (2011). "Vibration design of laminated fibrous composite plates with local anisotropy induced by short fibers and curvilinear fibers". In: *Composite Structures* 93.2, pp. 902–910.

— (2012). "Natural frequencies and vibration modes of laminated composite plates reinforced with arbitrary curvilinear fiber shape paths". In: *Journal of Sound and Vibration* 331.1, pp. 180–191.

Honda, Shinya, Yoshihiro Narita, and Katsuhiko Sasaki (2009). "Maximizing the fundamental frequency of laminated composite plates with optimally shaped curvilinear fibers". In: *Journal of system design and dynamics* 3.6, pp. 867–876.

Hou, Zhanghao et al. (2021). "Optimization design and 3D printing of curvilinear fiber reinforced variable stiffness composites". In: *Composites Science and Technology* 201, p. 108502.

Huang, Guanxin, Hu Wang, and Guangyao Li (2016). "An efficient reanalysis assisted optimization for variable-stiffness composite design by using path functions". In: *Composite Structures* 153, pp. 409–420.

Huang, Jinhua and RT Haftka (2005). "Optimization of fiber orientations near a hole for increased load-carrying capacity of composite laminates". In: *Structural and Multidisciplinary Optimization* 30.5, pp. 335–341.

Hyer, M W and RF Charette (1991). "Use of curvilinear fiber format in composite structure design". In: *AIAA journal* 29.6, pp. 1011–1015.

Hyer, Michael W and HH Lee (1991). "The use of curvilinear fiber format to improve buckling resistance of composite plates with central circular holes". In: *Composite structures* 18.3, pp. 239–261.

Izzi, Michele Iacopo, Anita Catapano, and Marco Montemurro (2021). "Strength and mass optimisation of variable-stiffness composites in the polar parameters space". In: *Structural and Multidisciplinary Optimization* 64.4, pp. 2045–2073.

Jared, Bradley H et al. (2017). "Additive manufacturing: Toward holistic design". In: *Scripta Materialia* 135, pp. 141–147.

Khan, Sadben et al. (2018). "Curvilinear variable stiffness 3D printing technology for improved open-hole tensile strength". In: *Additive Manufacturing* 24, pp. 378–385.

Kim, Byung Chul, Kevin Potter, and Paul M Weaver (2012). "Continuous tow shearing for manufacturing variable angle tow composites". In: *Composites Part A: Applied Science and Manufacturing* 43.8, pp. 1347–1356.

KISSlicer (2022). *KISSlicer Commercial Slicing Software Documentation*. https://www.kisslicer.com/documentation.html, Last Visited: 27-04-2022.

Kiyono, CY, ECN Silva, and JN Reddy (2017). "A novel fiber optimization method based on normal distribution function with continuously varying fiber path". In: *Composite structures* 160, pp. 503–515.

Klahn, Christoph, Bastian Leutenecker, and Mirko Meboldt (2015). "Design strategies for the process of additive manufacturing". In: *Procedia Cirp* 36, pp. 230–235.

Koch, Carsten, Luke Van Hulle, and Natalie Rudolph (2017). "Investigation of mechanical anisotropy of the fused filament fabrication process via customized tool path generation". In: *Additive Manufacturing* 16, pp. 138–145.

Kollar, Laszlo P and George S Springer (2003). *Mechanics of composite structures*. Cambridge university press.

Korelc, Jože (2007). *AceGen and AceFEM Manual*. http://symech.fgg.uni-lj.si/, Last Visited: 03-1-2022.

Kubalak, Joseph R, Alfred L Wicks, and Christopher B Williams (2019a). "Deposition path planning for material extrusion using specified orientation fields". In: *Procedia Manufacturing* 34, pp. 754–763.

— (2019b). "Exploring multi-axis material extrusion additive manufacturing for improving mechanical properties of printed parts". In: *Rapid Prototyping Journal*.

Kulkarni, P and D Dutta (1999). "Deposition strategies and resulting part stiffnesses in fused deposition modeling". In: *Journal of Manufacturing Science and Engineering*.

Le, Chau et al. (2010). "Stress-based topology optimization for continua". In: *Structural and Multidisciplinary Optimization* 41.4, pp. 605–620.

Lee, Boo Youn (1996). "Consideration of body forces in axisymmetric design sensitivity analysis using the BEM". In: *Computers & structures* 61.4, pp. 587–596.

Legrand, X et al. (2006). "Optimisation of fibre steering in composite laminates using a genetic algorithm". In: *Composite structures* 75.1-4, pp. 524–531.

Lemaire, Etienne, Samih Zein, and Michael Bruyneel (2015). "Optimization of composite structures with curved fiber trajectories". In: *Composite Structures* 131, pp. 895–904.

Li, Longmei et al. (2002). "Composite modeling and analysis for fabrication of FDM prototypes with locally controlled properties". In: *Journal of manufacturing processes* 4.2, pp. 129–141.

Liu, Jikai and Albert C To (2017). "Deposition path planning-integrated structural topology optimization for 3D additive manufacturing subject to self-support constraint". In: *Computer-Aided Design* 91, pp. 27–45.

Liu, Jikai and Huangchao Yu (2017). "Concurrent deposition path planning and structural topology optimization for additive manufacturing". In: *Rapid Prototyping Journal*.

Liu, Jikai et al. (2018). "Current and future trends in topology optimization for additive manufacturing". In: *Structural and multidisciplinary optimization* 57.6, pp. 2457–2483.

Lopes, Cláudio S et al. (2007). "Progressive failure analysis of tow-placed, variable-stiffness composite panels". In: *International Journal of Solids and Structures* 44.25-26, pp. 8493–8516.

Lopes, Claudio Saul (2009). "Damage and failure of non-conventional composite laminates". PhD thesis. TU Delft, Faculteit Luchtvaart-en Ruimtevaarttechniek.

Lopes, CS, Z Gürdal, and PP Camanho (2008). "Variable-stiffness composite panels: Buckling and first-ply failure improvements over straight-fibre laminates". In: *Computers & Structures* 86.9, pp. 897–907.

Lozano, G Gonzalez, A Tiwari, and C Turner (2018). "A design algorithm to model fibre paths for manufacturing of structurally optimised composite laminates". In: *Composite Structures* 204, pp. 882–895.

Lozano, Gustavo Gonzalez et al. (2016). "A review on design for manufacture of variable stiffness composite laminates". In: *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 230.6, pp. 981–992.

Magalhães, LC, N Volpato, and MA Luersen (2014). "Evaluation of stiffness and strength in fused deposition sandwich specimens". In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 36.3, pp. 449–459.

Malakhov, AV and AN Polilov (2016). "Design of composite structures reinforced curvilinear fibres using FEM". In: *Composites Part A: Applied Science and Manufacturing* 87, pp. 23–28.

Marschner, Steve and Peter Shirley (2015). *Fundamentals of computer graphics*. CRC Press.

MATLAB (2018). *9.7.0.1190202 (R2019b)*. Natick, Massachusetts: The MathWorks Inc.

McLoughlin, Tony et al. (2010). "Over two decades of integration-based, geometric flow visualization". In: *Computer Graphics Forum*. Vol. 29. 6. Wiley Online Library, pp. 1807–1829.

Mebarki, Abdelkrim, Pierre Alliez, and Olivier Devillers (2005). "Farthest point seeding for efficient placement of streamlines". In: *VIS 05. IEEE Visualization, 2005.* IEEE, pp. 479–486.

MII ImageView.

Mohamed, Omar A, Syed H Masood, and Jahar L Bhowmik (2015). "Optimization of fused deposition modeling process parameters: a review of current research and future prospects". In: *Advances in Manufacturing* 3.1, pp. 42–53.

Montemurro, Marco and Anita Catapano (2017). "On the effective integration of manufacturability constraints within the multi-scale methodology for designing variable angle-tow laminates". In: *composite structures* 161, pp. 145–159.

Mueller, Stefanie (2017). "3D printing for human-computer interaction". In: *interactions* 24.5, pp. 76–79.

Nagendra, S et al. (1995). "Optimization of tow fiber paths for composite design". In: *36th structures, structural dynamics and materials conference*, p. 1275.

Ngim, DB, J-S Liu, and RC Soar (2007). "Design optimization for manufacturability of axisymmetric continuum structures using metamorphic development". In: *International journal of solids and structures* 44.2, pp. 685–704.

Ngo, Tri-Dung (2020). "Introduction to composite materials". In: *Composite and Nanocomposite Materials—From Knowledge to Industrial Applications*.

Ngo, Tuan D et al. (2018). "Additive manufacturing (3D printing): A review of materials, methods, applications and challenges". In: *Composites Part B: Engineering* 143, pp. 172–196.

Nocedal, Jorge and Stephen Wright (2006). *Numerical optimization*. Springer Science & Business Media.

Onwubolu, Godfrey C and Farzad Rayegani (2014). "Characterization and optimization of mechanical properties of ABS parts manufactured by the fused deposition modelling process". In: *International Journal of Manufacturing Engineering* 2014.

Oropallo, William and Les A Piegl (2016). "Ten challenges in 3D printing". In: *Engineering with Computers* 32.1, pp. 135–148.

Pandey, PC and P Bakshi (1999). "Analytical response sensitivity computation using hybrid finite elements". In: *Computers & structures* 71.5, pp. 525–534.

Pandey, Pulak Mohan, N Venkata Reddy, and Sanjay G Dhande (2003). "Slicing procedures in layered manufacturing: a review". In: *Rapid prototyping journal*.

Parkinson, Alan R, R Balling, and John D Hedengren (2013). "Optimization methods for engineering design". In: *Brigham Young University* 5.11.

Peng, Tao et al. (2018). "Sustainability of additive manufacturing: An overview on its energy demand and environmental impact". In: *Additive Manufacturing* 21, pp. 694–704.

Price, Kenneth, Rainer M Storn, and Jouni A Lampinen (2006). *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media.

Puigoriol-Forcada, Josep M et al. (2018). "Flexural fatigue properties of polycarbonate fused-deposition modelling specimens". In: *Materials & Design* 155, pp. 414–421.

Rao, Singiresu S (2019). *Engineering optimization: theory and practice*. John Wiley & Sons.

Riddick, Jaret C et al. (2016). "Fractographic analysis of tensile failure of acrylonitrile-butadiene-styrene fabricated by fused deposition modeling". In: *Additive Manufacturing* 11, pp. 49–59.

Rinaldi, Marianna et al. (2018). "Additive layer manufacturing of poly (ether ether ketone) via FDM". In: *Composites Part B: Engineering* 145, pp. 162–172.

Roberge, Jeffrey and Julián Norato (2018). "Computational design of curvilinear bone scaffolds fabricated via direct ink writing". In: *Computer-Aided Design* 95, pp. 1–13.

Roberson, David A et al. (2015). "Comparison of stress concentrator fabrication for 3D printed polymeric izod impact test specimens". In: *Additive Manufacturing* 7, pp. 1–11.

Roberts, Tess (2020). *3D Printing Trends 2020*. https://www.hubs.com/blog/3d-printing-trends-2020/, Last Visited: 03-11-2021.

— (2021). *3D Printing Trends 2021*. https://www.hubs.com/get/trends/, Last Visited: 03-11-2021.

Rodríguez, José F, James P Thomas, and John E Renaud (2001). "Mechanical behavior of acrylonitrile butadiene styrene (ABS) fused deposition materials. Experimental investigation". In: *Rapid Prototyping Journal* 7.3, pp. 148–158.

Rodríguez-Panes, Adrián, Juan Claver, and Ana María Camacho (2018). "The influence of manufacturing parameters on the mechanical behaviour of PLA and ABS pieces manufactured by FDM: A comparative analysis". In: *Materials* 11.8, p. 1333.

Rousseau, Guillaume et al. (2019). "Automated Fiber Placement Path Planning: A state-of-the-art review". In: *Computer-Aided Design & Applications* 16.2, pp. 172–203.

Sá, Ângela AR et al. (2008). "Exploration vs. exploitation in differential evolution". In: *AISB 2008 Convention Communication, Interaction and Social Intelligence*. Vol. 1. Citeseer, p. 57.

Setoodeh, Shahriar et al. (2009). "Design of variable-stiffness composite panels for maximum buckling load". In: *Composite structures* 87.1, pp. 109–117.

Singh, Sunpreet et al. (2020). "Current status and future directions of fused filament fabrication". In: *Journal of Manufacturing Processes* 55, pp. 288–306.

Somireddy, Madhukar and Aleksander Czekanski (2017). "Mechanical characterization of additively manufactured parts by FE modeling of mesostructure". In: *Journal of Manufacturing and Materials Processing* 1.2, p. 18.

Somireddy, Madhukar, Aleksander Czekanski, and Chandra Veer Singh (2018). "Development of constitutive material model of 3D printed structure via FDM". In: *Materials Today Communications* 15, pp. 143–152.

Stegmann, Jan and Erik Lund (2005). “Discrete material optimization of general composite shell structures”. In: *International Journal for Numerical Methods in Engineering* 62.14, pp. 2009–2027.

Stillman, Douglas (2000). “Design sensitivity analysis for structures using explicit time integration”. In: *8th Symposium on Multidisciplinary Analysis and Optimization*, p. 4906.

Storn, Rainer and Kenneth Price (1997). “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces”. In: *Journal of global optimization* 11.4, pp. 341–359.

Storn, Rainer M and Kenneth Price (1995). *Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces*. Tech. rep. International Computer Science Institute.

Tam, Kam-Ming Mark and Caitlin T Mueller (2017). “Additive manufacturing along principal stress lines”. In: *3D Printing and Additive Manufacturing* 4.2, pp. 63–81.

Tatting, Brain F, Zafer Gürdal, and Dawn Jegley (2002). “Design and manufacture of elastically tailored tow placed plates”. In: *NASA STI report series*.

Tatting, Brian F (1998). “Analysis and design of variable stiffness composite cylinders”. PhD thesis. Virginia Tech.

Tekinalp, Halil L et al. (2014). “Highly oriented carbon fiber–polymer composites via additive manufacturing”. In: *Composites Science and Technology* 105, pp. 144–150.

Thompson, Mary Kathryn et al. (2016). “Design for Additive Manufacturing: Trends, opportunities, considerations, and constraints”. In: *CIRP annals* 65.2, pp. 737–760.

Tortorelli, Daniel A, Robert B Haber, and Stephen C-Y Lu (1991). “Adjoint sensitivity analysis for nonlinear dynamic thermoelastic systems”. In: *AIAA journal* 29.2, pp. 253–263.

Trunfio, Giuseppe A (2016). “Metaheuristics for continuous optimization of high-dimensional problems: State of the art and perspectives”. In: *Big Data Optimization: Recent Developments and Challenges*. Springer, pp. 437–460.

Van Keulen, F, RT Haftka, and NH Kim (2005). “Review of options for structural design sensitivity analysis. Part 1: Linear systems”. In: *Computer methods in applied mechanics and engineering* 194.30-33, pp. 3213–3243.

Varotsis, Alkaios Bournias (2019). *3D Printing Trends Q1 2019*. https://www.hubs.com/blog/3d-printing-trends-q1-2019/, Last Visited: 03-11-2021.

Wolfram Mathematica 13. https://www.wolfram.com/mathematica/, Last Visited: 03-11-2021.

Wu, Zhangming et al. (2012). "Buckling analysis and optimisation of variable angle tow composite plates". In: *Thin-walled structures* 60, pp. 163–172.

Yamanaka, Yusuke et al. (2016). "Fiber line optimization in single ply for 3D printed composites". In: *Open Journal of Composite Materials* 6.04, p. 121.

Zegard, Tomás and Glaucio H Paulino (2016). "Bridging topology optimization and additive manufacturing". In: *Structural and Multidisciplinary Optimization* 53.1, pp. 175–192.

Zhang, Pu, Jikai Liu, and Albert C To (2017). "Role of anisotropic properties on topology optimization of additive manufactured load bearing structures". In: *Scripta Materialia* 135, pp. 148–152.

Ziemian, Constance W, Ronald D Ziemian, and Kellen V Haile (2016). "Characterization of stiffness degradation caused by fatigue damage of additive manufactured parts". In: *Materials & Design* 109, pp. 209–218.