Materials Research Proceedings 41 (2024) 603-612 https://doi.org/10.21741/9781644903131-67

Modeling time-dependent anisotropy in MEX component-scale process simulation

DI NARDO Mario Emanuele[1,a *], FRÖLICH Felix[2,b], KÄRGER Luise[2,c] and CARLONE Pierpaolo[1,d]

[1]Department of Industrial Engineering, University of Salerno, 132 Via Giovanni Paolo II, 84084 Fisciano (SA), Italy

[2]Institute of Vehicle System Technology (FAST), Lightweight Design Division, Karlsruhe Institute of Technology, Rintheimer Querallee 2, Building 70.04, 76131 Karlsruhe, Germany

[a] mdinardo@unisa.it, [b] felix.froelich@kit.edu, [c] luise.kaerger@kit.edu, [d] pcarlone@unisa.it

Keywords: Additive Manufacturing, FFF, Process Simulation, FEM, Filament Orientation, Material Anisotropy, Homogenization

Abstract. In this paper, a numerical method is proposed to consider the transient anisotropic orientation state of a material extruded (MEX) part during a macroscale process simulation. To enhance computational efficiency, multiple filament strands are considered within single finite elements. The model is constructed around the utilization of 4th order orientation tensors, obtained by combining the information of the nozzle toolpath and the mesh elements in accordance with the process time. This provides a dynamic mapping of the anisotropic material orientation state within each element, in real-time to the process trajectory. Through the integration of time-dependent orientation tensors, this research provides deeper insights into filaments alignment evolution during the MEX process. This advancement not only enhances predictive capabilities in process simulation but also streamlines computational demands.

Introduction

Material Extrusion (MEX) is a highly versatile method within additive manufacturing, enabling the fabrication of intricate components with improved qualities [1]. Nevertheless, selecting the right process parameters is crucial in determining the ultimate characteristics of these components [2]. The alignment of the filament, which is directly linked to the path taken by the nozzle tool, holds a central significance in designing both the structural robustness and operational performance of produced components [3,4,5]. The pronounced anisotropy [6] observed in additively manufactured structures significantly influences their properties, necessitating thorough consideration and investigation when designing and evaluating components. Instead of the often-used trial and error approach, which stems from the interplay between the process, material and component, process simulation offers a more strategic solution. One of the key challenges in simulating the process is to capture the material orientation determined by the strand deposition. Leveraging computation and the finite element method proves to be a potent approach not just for forecasting the mechanical characteristics at a macroscopic level but also for simulating the manufacturing process itself for the prediction of process-induced distortion and residual stresses. However, a substantial portion of studies has predominantly concentrated on modeling either unidirectional or bidirectional raster orientations by predefining the filament's alignment [7,8,9,10,11]. Moreover, several studies delve into exploring how orientation impacts the mechanical characteristics of parts produced through MEX via experimental methods [8]. Biswas et al. [12] investigated orthotropic elastic properties by crafting computational models employing a representative volume element (RVE). Additionally, classical laminate theory (CLT) has been utilized to simulate orthotropic mechanical attributes [13]. However, it is essential to note that this

study primarily focuses on process simulation, requiring precise orientation data at each process time point, rendering techniques as CLT impractical. Although numerous studies have explored orientation effects using diverse modeling techniques and experimental approaches, there has not been a dedicated investigation into depicting deposition trajectories and their evolution over time. The aim is to introduce a streamlined approach that overcomes the computational intricacies associated with filament orientation. The objective is to enhance accuracy while maintaining computational efficiency. To achieve this, orientation tensors are formulated, giving continual update of the orientation state for every element throughout the process [14,15]. This modeling approach involves the finite element software ABAQUS in conjunction with specific subroutines. The recent development involves incorporating these codes as "internal" user subroutines within an Abaqus plugin called AM Modeler. In this paper, the novel methodology is presented by first explaining the mathematical formulation of tensors and their special properties in the MEX process, followed by a general explanation of the workflow of the method. In addition, a verification scenario is presented and the resulting orientation states within an element are discussed and compared with an analytical solution.

Methodology

Orientation Tensors

The orientation tensor formulation was introduced by Kanatani [14] and further developed by Advani and Tucker [15] serving to map the alignment probability of fibers in composites. This tensor aids in detailing the orientation distribution function (ODF) [15], reflecting how fibers align within a composite. Their approach involves utilizing spherical harmonics to represent the distribution of fiber orientations. By employing these mathematical functions, Advani and Tucker expanded the ODF into a series, creating a representation that encapsulates the diverse orientations of fibers within the material. The resulting 4th order tensor comprises coefficients representing different spherical harmonic terms, offering insights into the distribution and intensity of fibers in various orientations within the composite.

The representation of fiber directions is denoted by unit vectors, symbolized as **p**. These unit vectors **p** correspond to the spherical coordinates defining the direction of fiber alignment in three-dimensional space as shown in Figure 1:

$$\mathbf{p} = \begin{Bmatrix} p_x \\ p_y \\ p_z \end{Bmatrix} = \begin{Bmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \\ \cos\theta \end{Bmatrix} \tag{1}$$

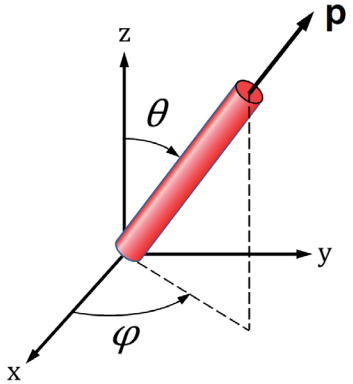

Figure 1. The unit vector **p** *characterized by the angles θ and φ in relation to Cartesian coordinate axes. (Adapted from [15]).*

Due to the unitary nature, traditional scalar or cross products cannot account for a distribution, leading to results lacking comprehensive information [15]. To encapsulate the fiber orientation as a tensor, the dyadic product is employed, which consists in a mathematical operation combining vectors to create tensors by multiplying the vectors components for themselves [16]. This operation involves multiplying the vectors components four times, yielding a 4th order tensor $\mathbb{A}$ that characterizes how the fibers are aligned in relation to the coordinate axes. Furthermore, given the need of a distribution representation, the formulation includes an averaging of multiple orientation vectors denoted within the angle brackets:

$$\mathbb{A} \equiv \langle \mathbf{pppp} \rangle \equiv \begin{bmatrix} A_{1111} & A_{1122} & A_{1133} & A_{1123} & A_{1131} & A_{1112} \\ A_{2211} & A_{2222} & A_{2233} & A_{2223} & A_{2231} & A_{2212} \\ A_{3311} & A_{3322} & A_{3333} & A_{3323} & A_{3331} & A_{3312} \\ A_{2311} & A_{2322} & A_{2333} & A_{2323} & A_{2331} & A_{2312} \\ A_{3111} & A_{3122} & A_{3133} & A_{3123} & A_{3131} & A_{3112} \\ A_{1211} & A_{1222} & A_{1233} & A_{1223} & A_{1231} & A_{1212} \end{bmatrix} \quad (2)$$

The utilization of the 4th order orientation tensor across multiple studies is rooted in its ability to provide a holistic linkage between fiber orientation and material properties. It serves as a pivotal tool to capture directional details that are necessary for comprehending the impact of fiber alignment on material behavior [17,18]. It allows for precise predictions of mechanical or thermal responses along different axes, providing a robust framework for studying material properties in diverse conditions.

Distinctive properties of the orientation vectors and tensors within the MEX process

In the context of this research, the orientation tensor $\mathbb{A}$ is used to describe the distribution of orientation after deposition events at various time steps within a finite element. Consequently, the orientation vectors are describing the alignment of the filament, which are considered straight and rigid as in [15]. The tensor notation, as shown in Eq. 2, represents the visualization in the contracted notation as used in [15], which basically translates into the Voigt notation. It encapsulates information regarding the alignment distribution along the axes, providing insights into the anisotropic properties during the extrusion process.

Indeed, due to the nature of the MEX process characterized by sole deposition on the x-y plane, the angle θ within the orientation vector $\mathbf{p}$ remains consistently at $\pi/2$, resulting in a fixed z-component, leading to $p_z = 0$. Consequently, this characteristic trait shapes the 4th order distribution tensor in a distinct manner, rendering only those tensor components non-zero that arise from the multiplication without the involvement of the z-component:

$$\mathbb{A}^{(t=n)}(NOEL) = \begin{bmatrix} A_{1111} & A_{1122} & 0 & 0 & 0 & A_{1112} \\ A_{2211} & A_{2222} & 0 & 0 & 0 & A_{2212} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ A_{1211} & A_{1222} & 0 & 0 & 0 & A_{1212} \end{bmatrix} \quad (3)$$

In this numerical modeling, for each element of the mesh in the FE Model, an orientation tensor is formulated at every time increment defined in the simulation, leading to tensors dependent on the time increment. The NOEL parameter used in Eq. 3 represents the element numbering defined in Abaqus.

A generalized formula for updating A_{ijkl} incorporates weighted contributions from multiple orientation vectors, as previously denoted in Eq.2 using angular brackets. Thus, the formulation of entries within the orientation tensor A_{ijkl} can be expressed as follows:

$$A_{ijkl}^{(t=n)}(NOEL) = \sum_{m=1}^{N} \left(\frac{w_m}{\sum_{o=1}^{N} w_o}\right) \cdot \left(p_i \cdot p_j \cdot p_k \cdot p_l\right)_m , \quad (4)$$

where:

- $A_{ijkl}^{(t=n)}$ represents the orientation tensor entry after deposition at time $t = n$;
- N signifies the number of orientation vectors $\mathbf{p}_m$ observed during the deposition;
- w_m denotes the weight associated with each orientation vector $\mathbf{p}_m$ reflecting its significance or occurrence frequency.

Modeling Methodology

The method's workflow, as depicted in Figure 2, involves a combination of Python and Fortran programming. To simulate the MEX process, an FE model is generated using Abaqus, leveraging the AM Modeler plugin.

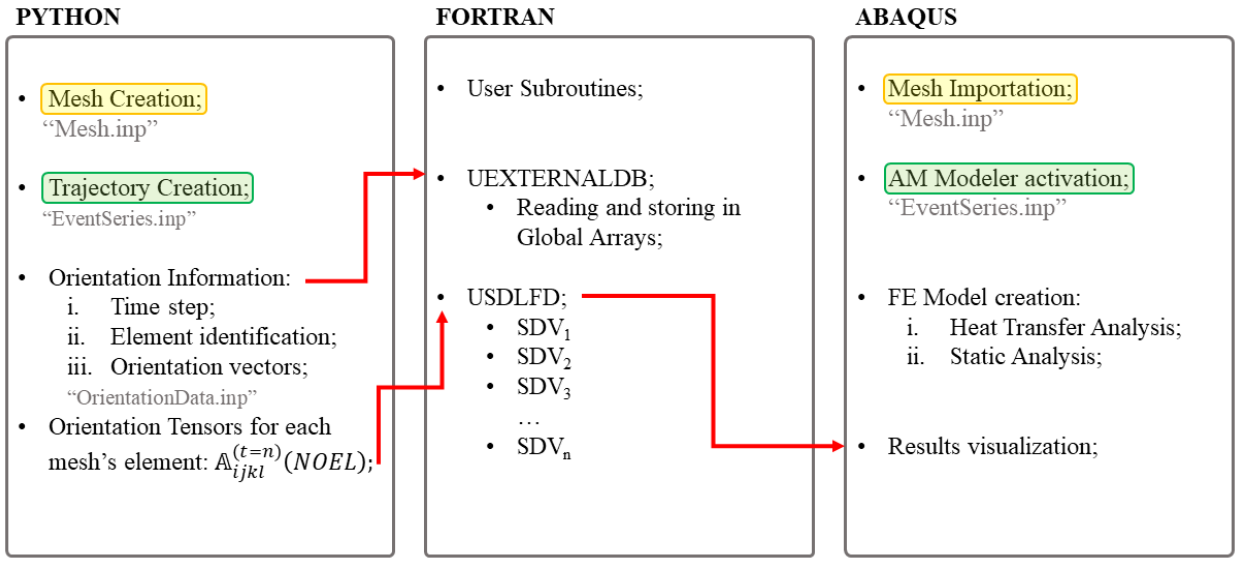

Figure 2. Workflow of the modeling methodology

Python is instrumental in creating import files with ".inp" extension tailored for Abaqus compatibility and generating the analytical solution necessary for validation purposes.

After the slicing phase that generates the G-Code, there's a subsequent step involving extrapolation of data. This phase aims to determine the process time, the coordinates followed by the nozzle, and an ON-OFF variable called EA, indicating when the extruder is activated during the deposition process. This data is organized within a file named "EventSeries.inp", structured as depicted in Table 1.

Table 1. Structure of toolpath data.

Process Time	x	y	z	EA
[s]	[mm]	[mm]	[mm]	[/]

Simultaneously, another phase entails creating the component mesh, which it has been made by using the Meshio package [19], resulting in an import file named "Mesh.inp" containing all the necessary information that can be read by Abaqus in VTK format.

At this stage, merging these datasets allows for pinpointing the exact cell activated as the nozzle progresses along its trajectory. Indeed, after the mesh and trajectory data are loaded, the Python script seeks the time step value. Using this input, the script outputs the necessary information required to formulate the 4th order orientation tensors throughout the time simulation. This results in a new import file named "OrientationData.inp," structured as in Table 2.

Table 2. Structure of the orientation data containing the vectors components and frequency.

Total Time	NOEL	p_x	p_y	p_z	w_m
[s]	[/]	[/]	[/]	[/]	[/]

To give a clearer understanding, as shown in Figure 3, when the printhead traverses the mesh, the script recognizes the element index by manipulating its vertices coordinates and the printhead coordinates outputting the components of the orientation vectors and their corresponding weights, indicating the quantity of beads deposited. These insights hold significant importance for deriving Python results and for seamless integration into the finite element (FE) model via Subroutines.

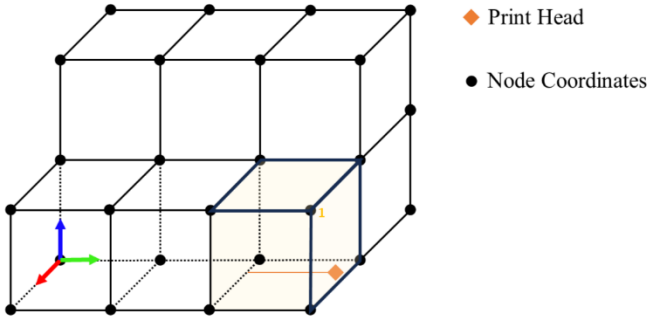

Figure 3. Mesh portion composed of cube elements. The script operates by identifying the print head at element 1, distinguished by its specific node coordinates.

The final step regards the creation of the FE model and the activation of the AM Modeler in Abaqus. The AM Modeler plugin is instrumental for the element activation, representing the addition of material during the simulation and ensuring accurate heat transfer analysis interactions. During activation, non-isothermal properties undergo a homogenization process, wherein the accumulated heat input through the strands is averaged over time. Subsequently, this averaged thermal energy is introduced into the component as a unified effect upon activation. In instances of partial activation, only a specific portion of the element is activated, and the corresponding thermal energy is introduced proportionally. Steps for analysis could involves a heat transfer study followed by a static analysis. This way of modelling is mainly used for aiming at predicting internal stresses and warpage resulting from thermal histories affecting the filament and, consequently, the final structure. Furthermore, a notable feature of the AM Modeler plugin is its ability to utilize the previously created “EventSeries.inp” file. This enables the display of material addition during the simulation in line with process timings. However, it is important to note that it is possible to take the orientation into account, but only as a vector where the x-direction is the extrusion direction. If there are several strands in one element, this is no longer representative. Essential to note is that

this approach focuses exclusively on the orientation distribution over time, which is determined by the trajectory of the nozzle. Indeed, it is crucial to acknowledge that the changing mass of the element also plays a significant role. One effective approach to address this could be achieved by defining a time-dependent mass for each element, taking into consideration its density. The "OrientationData.inp" file is initially integrated using the UEXTERNALDB subroutine [20]. This subroutine is invoked once at the outset of the analysis and facilitates communication with other user subroutines, specifically the USDFLD [20], within ABAQUS/Standard. In this modeling, the subroutine UEXTERNALDB reads the external file with orientation data created in Python once at the beginning of the process simulation. Each column of this file is stored in so-called global arrays. This type of array can be called up for each integration point and is therefore valid for the entire domain. The data of the individual columns can thus be efficiently managed using special pointers and called up in the USDFLD subroutine for each integration point.

After storing each column's data in a global array, the data is processed similarly to the analytical method employed in Python to generate the orientation tensors for every element within the mesh. However, the construction of the algorithm in this context is based on the variables: NOEL (element identification index), NPT (integration point), and DTIME (time increment). This design enabled the formation of orientation tensors at every time increment (DTIME) and for every integration point (NPT). Leveraging the element identification index (NOEL), iteration is made feasible for each cell of the mesh. In practical terms, all tensor entries are connected to a state-dependent variable (SDV) as specified in:

$$\mathbb{A}^{(t=n)}(NOEL) = \begin{bmatrix} A_{1111} & A_{1122} & 0 & 0 & 0 & A_{1112} \\ A_{2211} & A_{2222} & 0 & 0 & 0 & A_{2212} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ A_{1211} & A_{1222} & 0 & 0 & 0 & A_{1212} \end{bmatrix} = \begin{bmatrix} SDV_1 & SDV_3 & 0 & 0 & 0 & SDV_4 \\ SDV_3 & SDV_2 & 0 & 0 & 0 & SDV_5 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ SDV_4 & SDV_5 & 0 & 0 & 0 & SDV_3 \end{bmatrix} \quad (5)$$

This variable SDV is used to store and retrieve additional information about the state of each element at each integration point. It adopts a scalar value for each time increment, effectively representing the values embodied by the tensor elements.

Verification Scenario

A verification test employed a cube as the simulation component, utilizing a model comprised of 27 elements and a trajectory alternating between deposition at 0° and 90°, according to the x-axis.

This toolpath excludes deposition on the perimeter giving the two different orientation vectors:

$$\boldsymbol{p}_1 = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix};\ \boldsymbol{p}_2 = \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix}. \quad (6)$$

Focusing on specific orientations streamlines the formulation of 4th orientation tensors, facilitating a clearer understanding of the final outcomes. These tensors predominantly exhibit non-zero values along the first two positions of the main diagonal, indicating alignment along the x and y axes. The computation of trajectory orientation integrates these details as per guidelines outlined in Section 2.3, based on the chosen time increment. In this MEX simulation, the time increment corresponds to the duration required for depositing a layer. Considering the geometry, trajectory, and printing speed, this time increment approximates 97.14 seconds. Consequently, every 97.14 seconds triggers the deposition of a new layer, leading to a simulation update for each element.

To simplify the representation of tensor filling, the chosen process parameters involve depositing 15 lanes per layer, distributed as 5 lanes per layer and per element, with a total of 27 layers. This designates 9 layers for each element. However, adjusting for shorter time steps may require more frequent tensor formulation and subsequent updates.

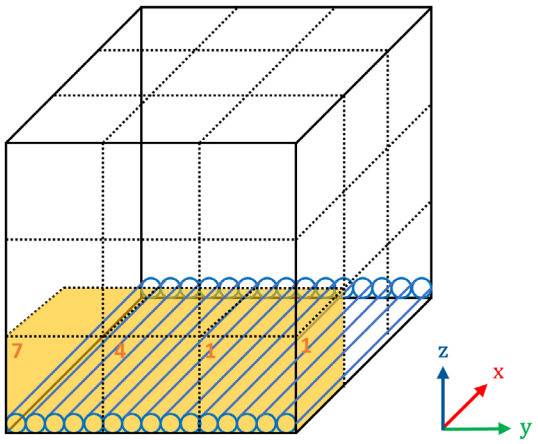

Figure 4. Visualization of the cube mesh during the initial layer deposition along the x-axis. Notably, elements 1, 4, and 7 are activated at this stage of the process.

Table 3. Data snippet generated by the script at the first layer deposition.

Total Time	NOEL	p_x	p_y	p_z	w_m
97.14 [s]	1	1	0	0	5
97.14 [s]	4	1	0	0	5
97.14 [s]	7	1	0	0	5

At the first layer deposition, during the initial time step (Total Time = 97.14s), activation occurs for all the elements at the base of the cube, revealing both the count of deposited filaments (w_m) and the individual components of the orientation vectors (p_x, p_y, p_z). Specifically, let's narrow our focus solely to Elements 1, 4, and 7 as depicted in Figure 4. As the nozzle traverses these elements, they get recognized and activated. Table 3 provides a comprehensive display, showcasing the count of deposited filaments ($w_m = 5$) and the corresponding vector components that define the orientation along the x-axis. This entire process iterates for the remaining elements, updating after each new layer deposition.

Results and discussion

To validate the method, Element 1 is analysed, which is located at the base of the cube, as shown in Figure 4. Given the simple toolpath and the time increment chosen, the tensors are the same for the other elements at the cube's bottom. Indeed, the nine elements at the cube's bottom are the initial ones to be activated. This is due to the process's inherent nature, which constructs objects beginning from the initial layer, responsible for adhesion to the building platform, and culminating in the final layer, adhering to a layer-by-layer strategy. As shown in the orientation data snippet in Table 3, at the initial time increment (97.14 s), as the first layer is laid down, the data indicates 10 lanes deposited in the x direction. At this point it is possible to visualize the orientation sensor of this element. At the first time increment, the tensor corresponds to a null matrix, with the first term on the main diagonal equal to 1. This means that 100% of the filaments are fully aligned in the x-direction. At the second time increment (198.28 s) the algorithm considers also the second layer which is oriented in the y direction. The tensor then is updated with a value of 0.5 for the first two terms on the principal diagonal, signifying that 50% of the filaments are oriented in the x

direction and the other 50% in the y direction. The updating goes on until the last layer is deposited. Figure 5 visualizes the tensor's updates for the Element 1 after the deposition of three layers.

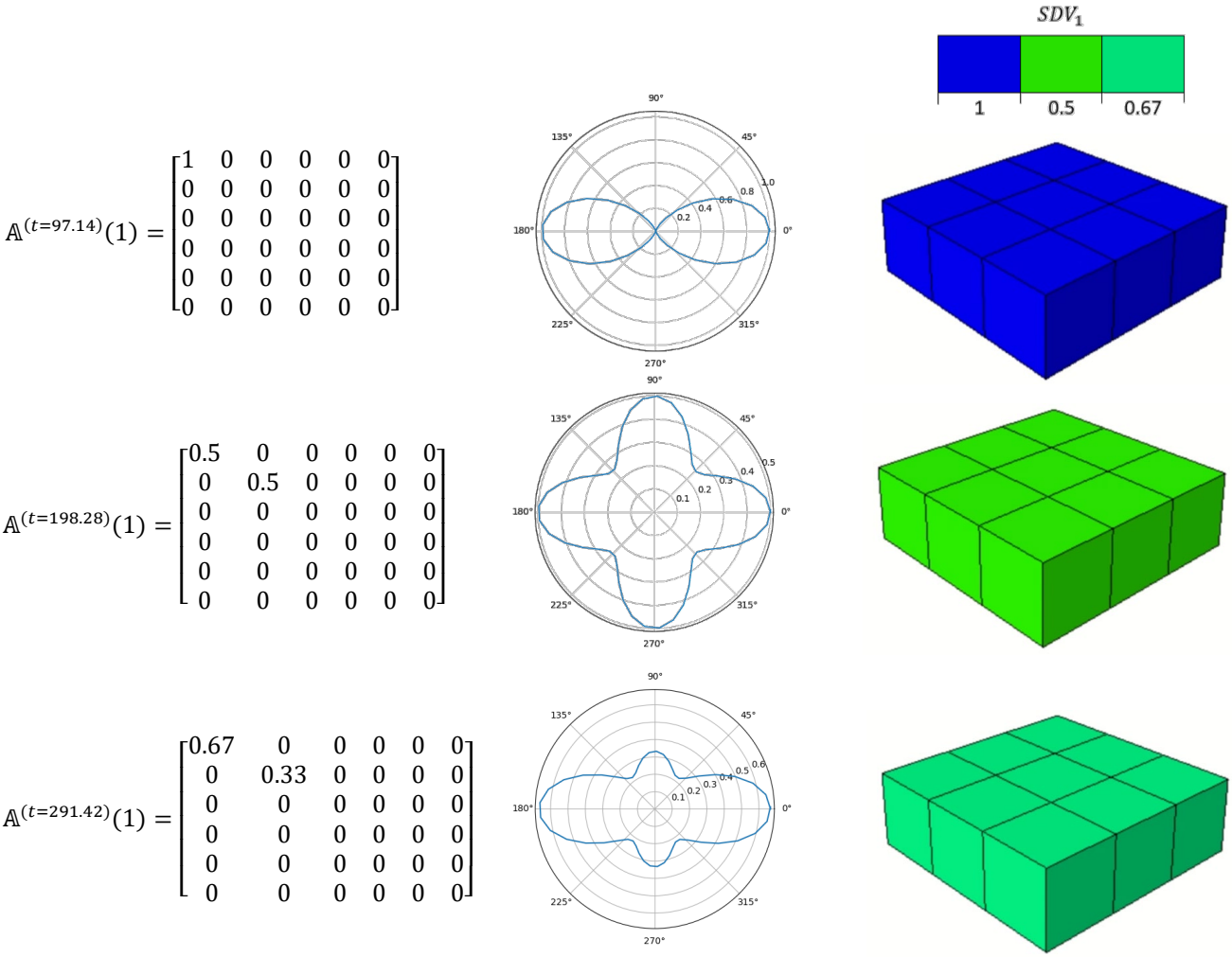

Figure 5. Updates pertaining to Element 1. On the Left: Tensor states throughout the deposition phases. Middle: 2D representation of these tensors. Right: SDV_1 is exhibited using a color-coded scheme as implemented in Abaqus.

The analytical results are then compared with Abaqus results. Given the straightforward deposition trajectory of 0° and 90° in this study, the sole entries undergoing value changes during the simulation are the first two elements on the primary diagonal SDV_1 and SDV_2. As shown in Figure 4, in Abaqus the individual entries of the tensors can be visualised over time in the component by associating a code color. The comparison shows that the implementation in Abaqus produces the same values and is therefore verified by the analytical solution.

Conclusion

The presented methodology demonstrates a systematic approach to simulate the orientation state in the MEX process, considering the interplay between the nozzle toolpath and induced anisotropic material orientation. This transient anisotropic material orientation state is crucial for predicting residual stresses and distortion accurately. Utilizing 4th order orientation tensors provides a holistic perspective of weighted average orientation within each finite element. It is important to note that the verification scenario is simplistic and could be adequately addressed with 2nd rank tensors.

However, using 4th order orientation tensor permits to describe the resulting distribution on stiffness tensors and in more complex components with elaborate trajectories, more complicated orientation distributions may occur, which make the 4th order tensor necessary. This methodology, integrating Python, Fortran programming, and Abaqus subroutines, streamlines the workflow from generating import files to incorporating orientation data, ensuring a representation in the FE model. The alignment depicted in the analytical solution mirrors the orientation changes captured in Abaqus, verifying the method in tracking orientation states for each element and at each time step. Visualization of tensor updates further solidifies the congruence between the analytical and numerical results. Future enhancements could focus on integrating material properties associated with orientation for a more comprehensive predictive model. Indeed, this method stands to enhance predictive capabilities by serving of time-dependent orientation tensors as valuable inputs, enabling a nuanced correlation with material intrinsic properties by using homogenization approaches.

Authors contribution

Conceptualization: Mario Emanuele Di Nardo, Felix Frölich, Luise Kärger; Methodology: Mario Emanuele Di Nardo, Felix Frölich; Software: Mario Emanuele Di Nardo, Felix Frölich; Visualization: Mario Emanuele Di Nardo, Felix Frölich; Writing - original draft preparation: Mario Emanuele Di Nardo, Felix Frölich; Writing - review and editing: Luise Kärger, Pierpaolo Carlone; Funding acquisition: Luise Kärger, Pierpaolo Carlone; Supervision: Felix Frölich, Luise Kärger, Pierpaolo Carlone.

Acknowledgements

The authors thank Baden-Württemberg State Ministry of Science, Research and the Arts (MWK) for funding the project "Basics of a remanufacturing process chain for functional, hybridized polymer components to increase reusability and optimize resource utilization (Restore)" as part of the Innovation Campus Future Mobility (ICM), and the German Research Foundation (DFG) for funding the Heisenberg Professorship of Prof. Kärger.

References

[1] Gibson, I., Rosen, D. W., Stucker, B., Khorasani, M., Rosen, D., Stucker, B., & Khorasani, M. (2021). Additive manufacturing technologies (Vol. 17, pp. 160-186). Cham, Switzerland: Springer. https://doi.org/10.1007/978-3-030-56127-7

[2] Kaur, G., Singari, R. M., & Kumar, H. (2022). A review of fused filament fabrication (FFF): Process parameters and their impact on the tribological behavior of polymers (ABS). Materials Today: Proceedings, 51, 854-860. https://doi.org/10.1016/j.matpr.2021.06.274

[3] Li, J., Yang, S., Li, D., & Chalivendra, V. (2018). Numerical and experimental studies of additively manufactured polymers for enhanced fracture properties. Engineering Fracture Mechanics, 204, 557-569. https://doi.org/10.1016/j.engfracmech.2018.11.001

[4] Rashid, A. A., & Koç, M. (2021). Fused filament fabrication process: a review of numerical simulation techniques. Polymers, 13(20), 3534. https://doi.org/10.3390/polym13203534

[5] Jin, Y. A., He, Y., Fu, J. Z., Gan, W. F., & Lin, Z. W. (2014). Optimization of tool-path generation for material extrusion-based additive manufacturing technology. Additive manufacturing, 1, 32-47. https://doi.org/10.1016/j.addma.2014.08.004

[6] Ahn, S. H., Montero, M., Odell, D., Roundy, S., & Wright, P. K. (2002). Anisotropic material properties of fused deposition modeling ABS. Rapid prototyping journal, 8(4), 248-257. https://doi.org/10.1108/13552540210441166

[7] Cattenone, A., Morganti, S., Alaimo, G., & Auricchio, F. (2019). Finite element analysis of additive manufacturing based on fused deposition modeling: Distortions prediction and comparison with experimental data. Journal of Manufacturing Science and Engineering, 141(1), 011010. https://doi.org/10.1115/1.4041626

[8] Brenken, B., Favaloro, A., Barocio, E., DeNardo, N. M., & Pipes, R. B. (2016, May). Development of a model to predict temperature history and crystallization behavior of 3D printed parts made from fiber-reinforced thermoplastic polymers. In Int. SAMPE Tech. Conf (Vol. 12, p. 704).

[9] Zhou, Y., Lu, H., Wang, G., Wang, J., & Li, W. (2020). Voxelization modelling based finite element simulation and process parameter optimization for Fused Filament Fabrication. Materials & Design, 187, 108409. https://doi.org/10.1016/j.matdes.2019.108409

[10] Ahn, S. H., Baek, C., Lee, S., & Ahn, I. S. (2003). Anisotropic tensile failure model of rapid prototyping parts-fused deposition modeling (FDM). International Journal of Modern Physics B, 17(08n09), 1510-1516. https://doi.org/10.1142/S0217979203019241

[11] Domingo-Espin, M., Puigoriol-Forcada, J. M., Garcia-Granada, A. A., Llumà, J., Borros, S., & Reyes, G. (2015). Mechanical property characterization and simulation of fused deposition modeling Polycarbonate parts. Materials & Design, 83, 670-677. https://doi.org/10.1016/j.matdes.2015.06.074

[12] Biswas, P., Guessasma, S., & Li, J. (2020). Numerical prediction of orthotropic elastic properties of 3D-printed materials using micro-CT and representative volume element. Acta Mechanica, 231(2), 503-516. https://doi.org/10.1007/s00707-019-02544-2

[13] Zhao, Y., Chen, Y., & Zhou, Y. (2019). Novel mechanical models of tensile strength and elastic property of FDM AM PLA materials: Experimental and theoretical analyses. Materials & Design, 181, 108089. https://doi.org/10.1016/j.matdes.2019.108089

[14] Kanatani, K. (1984). Distribution of directional data and fabric tensors. International journal of engineering science, 22(2), 149-164. https://doi.org/10.1016/0020-7225(84)90090-9

[15] Advani, S. G., & Tucker III, C. L. (1987). The use of tensors to describe and predict fiber orientation in short fiber composites. Journal of rheology, 31(8), 751-784. https://doi.org/10.1122/1.549945

[16] Lebedev, L. P., Cloud, M. J., & Eremeyev, V. A. (2010). Tensor analysis with applications in mechanics. World Scientific. https://doi.org/10.1142/7826

[17] Heller, B. P., Smith, D. E., & Jack, D. A. (2017). Simulation of planar deposition polymer melt flow and fiber orientation in fused filament fabrication. In 2017 International Solid Freeform Fabrication Symposium. University of Texas at Austin.

[18] Lupone, F., Padovano, E., Venezia, C., & Badini, C. (2022). Experimental Characterization and Modeling of 3D Printed Continuous Carbon Fibers Composites with Different Fiber Orientation Produced by FFF Process. Polymers, 14(3), 426. https://doi.org/10.3390/polym14030426

[19] Python Documentation, Meshio Package, Available on 08/12/2023, Information on: https://pypi.org/project/meshio/

[20] Abaqus 6.14 documentation, User Subroutines Reference Guide, Available on 08/12/20023, Information on: http://130.149.89.49:2080/v6.14/