

¹ Why develop twice? User material interface for the
² Peridynamic Peridigm framework

³ **Christian Willberg¹, Jan-Timo Hesse^{*1}, Marc Garbade¹, Martin Rädel¹,**
⁴ **Falk Heinecke¹, Andreas Schuster¹, and Anna Pernatii²**

⁵ **1** Department of Structural Mechanics, Institute for Composite Structures and Adaptive Systems,
⁶ German Aerospace Center **2** Institute of Mechanics, Otto von Guericke University Magdeburg

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review ↗](#)
- [Repository ↗](#)
- [Archive ↗](#)

Editor: ↗

Submitted: 14 April 2022

Published: unpublished

License

Authors of papers retain
copyright and release the work
under a Creative Commons
Attribution 4.0 International
License ([CC BY 4.0](#)).

⁷ Summary

⁸ In order to analyze complex and large scale structural mechanical problems, simulation
⁹ frameworks are required. These require material models to assess the response of a medium to
¹⁰ loading conditions. The type of material model necessary depends on the physical problem, its
¹¹ implementation and idealization in a structural model. Therefore, an efficient way of using
¹² different kinds of material models in a peridynamic simulation is required. In this paper a
¹³ generalized user material interface is presented that allows the usage of existing as well as
¹⁴ novel material models. In order to implement this interface, the material modeling of the
¹⁵ correspondence formulation in Peridigm was restructured. Local coordinates were added and
¹⁶ the unit tests were extended.

¹⁷ Statement of need

¹⁸ The user material interface allows the simplified use of already existing material routines in the
¹⁹ peridynamic framework Peridigm. The interface is based on the Abaqus UMAT definition
²⁰ ([UMAT, 2022](#)) and allows the integration of Fortran routines directly into Peridigm. The
²¹ integration of UMAT routines based on finite elements in Peridigm eliminates the need
²² for parallel development of existing material models from classical continuum mechanics
²³ theory. Thus, the same material model implementations are applicable in finite element as
²⁴ well as peridynamic simulations. This opens up new possibilities for analysis, verification
²⁵ and comparison. With this interface many material routines can be reused and applied to
²⁶ progressive failure analysis. The source code is stored in a GitHub repository ([2022](#)).
²⁷

²⁸ Introduction

²⁹ Peridynamics is a non-local theory which overcomes the problems of classical continuum
³⁰ mechanics at discontinuities due to the necessity of spatial derivations of the underlying partial
³¹ differential equations. The latter are not defined at discontinuities such as cracks. It is solved
³² by using an integral formulation instead of a differential one. As a result, existing material
³³ models have to be rewritten. To improve the usability of the Peridynamic theory in 2007 the
³⁴ so-called correspondence formulation was developed by Silling et al. ([Silling et al., 2007](#)).
³⁵ This formulation introduces a non-local integral deformation gradient which allows the use of
³⁶ classical continuum mechanical models in Peridynamics. The non-local deformation gradient
³⁷ allows the calculation of classical strain and stress measures.
³⁸ Since, Peridynamics is motivated by the analysis of crack propagation processes, a mesh-free

*Co-first author

39 method is usually used for the numerical solving process. One of the more advanced frameworks
 40 is provided by Sandia National Labs and is called Peridigm ([Parks et al., 2012; Rädel & Willberg,](#)
 41 [2018](#)). The framework allows the parallelization of large scale models and has a post processing
 42 interface to the open source software ParaView¹. Some extensions were introduced in the
 43 recent years. Within this software publication, e.g. energy based ordinary state-based damage
 44 model, anisotropy, correspondence energy damage model ([Willberg et al., 2019, 2022; Willberg](#)
 45 [& Heinecke, 2021](#)) will be added to Peridigm. However, the current structure of Peridigm
 46 does not allow the direct use of already existing material models. Material models have to
 47 be rewritten to use them in Peridigm. This holds true for classical peridynamic as well as
 48 correspondence material models. Abaqus is a general finite element solver that can be used
 49 to model different material behaviors. This software provides an interface to include user
 50 materials (UMAT). UMATs usually are written in Fortran and it is the quasi standard in this
 51 research domain, although alternative formats exist. Therefore, the goal of this publication is
 52 the provision of a direct Peridigm - Abaqus UMAT interface. This interface reduces the hurdle
 53 of material modeling in Peridynamics and increases the advantages of the Peridigm framework
 54 significantly. The approach presented here follows a setup that allows easy extension to other
 55 material models as well as languages.

56 Implementation and architecture

57 The user material interface is motivated by the Abaqus UMAT interface. Before the material
 58 routine can be used, the strain values have to be transformed into local coordinates. The stress
 59 values, which are calculated via the fortran routine, are transformed back into the original
 60 coordinates. The [Table 1](#) gives an overview about the interface and supported parameters.
 61 Obviously not all parameters are valid, because they are specific to the finite element format.

Table 1: Interface parameter of the UMAT in Peridigm (PD). In Dogbone_Umat.yaml it is shown
 how to call a user material with a user defined number of properties and state variables.

Name	Type	Size	Description	Supported
sigmaNP1	double[]	ntens	Mechanical stresses	yes
statev	double[]	nstatev	User defined state variables	yes
DDSDDE	double[]	ntens × ntens	Jacobian matrix of the constitutive model $\partial\sigma/\partial\varepsilon$	no
SSE	double	1	Specific elastic strain energy	no
SPD	double	1	Specific plastic dissipation	no
SCD	double	1	Specific creep dissipation energy	no
RPL	double	1	Volumetric heat generation per unit time	no
DDSDDT	double[]	ntens	Variation of the stress increments with respect to the temperature	no

Name	Type	Size	Description	Supported
DRPLDE	double[]	ntens	Variation of RPL with respect to the strain increment.	no
DRPLDT	double	1	Variation of RPL with respect to the temperature	no
stran dstran time(1)	double[] double[] double	ntens × ntens ntens × ntens 1	strain Strain increment Step time at the beginning of the current increment	yes yes no
time(2)	double	1	Total time at the beginning of the current increment	yes
dtime temp dtemp	double double double	1 1 1	Time increment Temperature Temperature increment	yes yes yes
PREDEF DPRED	double[] double[]	- -	Predefined fields Array of increments of predefined field variables	no no
CMNAME ndi	string int	80 2 or 3	Material name Number of direct stress components at this point	yes yes
nshr	int	1 or 3	Number of engineering shear stress components	yes
ntens	int	ndi + nshr	Size of the stress or strain component array	yes
nstatev	int		Number of state variables	yes
props nprops	double[] int	nprops 1	Property values Number of properties	yes yes
coords drot	double[] double[]	2 or 3 3 × 3	Coordinates Rotation increment matrix	yes yes
PNEWDT	double	1	Ratio of suggested new time increment	no

Name	Type	Size	Description	Supported
CELENT	double	1	Characteristic element length	no
DFGRD0	double[]	3×3	Deformation gradient N	yes
DFGRD1	double[]	3×3	Deformation gradient N+1	yes
NOEL	int	1	Element number	
NPT	int	1	Integration point number	
KSLAY	int	1	Layer number	
KSPT	int	1	Section point number	no
JSTEP	int	1	Step number	no
KINC	int	1	Increment number	no

- 62 The overall interface architecture is given in [Figure 1](#). The Peridigm user material is structured
 63 like a typical material used in Peridigm. In order to be able to use a material routine, the
 64 UMAT file must be precompiled and copied to a specific folder. An additional interface layer is
 65 introduced. It is used to transform or calculate specific parameters for the UMAT. Because, the
 66 material name is not transferable directly from Peridigm which is written in C++ to Fortran,
 67 a dedicated Fortran routine is provided. This routine transforms the string definition of C++
 68 to the character field definition of Fortran.
 69 The interface allows an arbitrary number of properties and state variables. The definition is
 70 shown in the Listing below or in the example, which is given in the repository. The state
 71 variables allow specific calculations, e.g. the history of a discrete material response or property.
 72 These state variables can be saved and requested for output in the output section of the
 73 Peridigm input file.

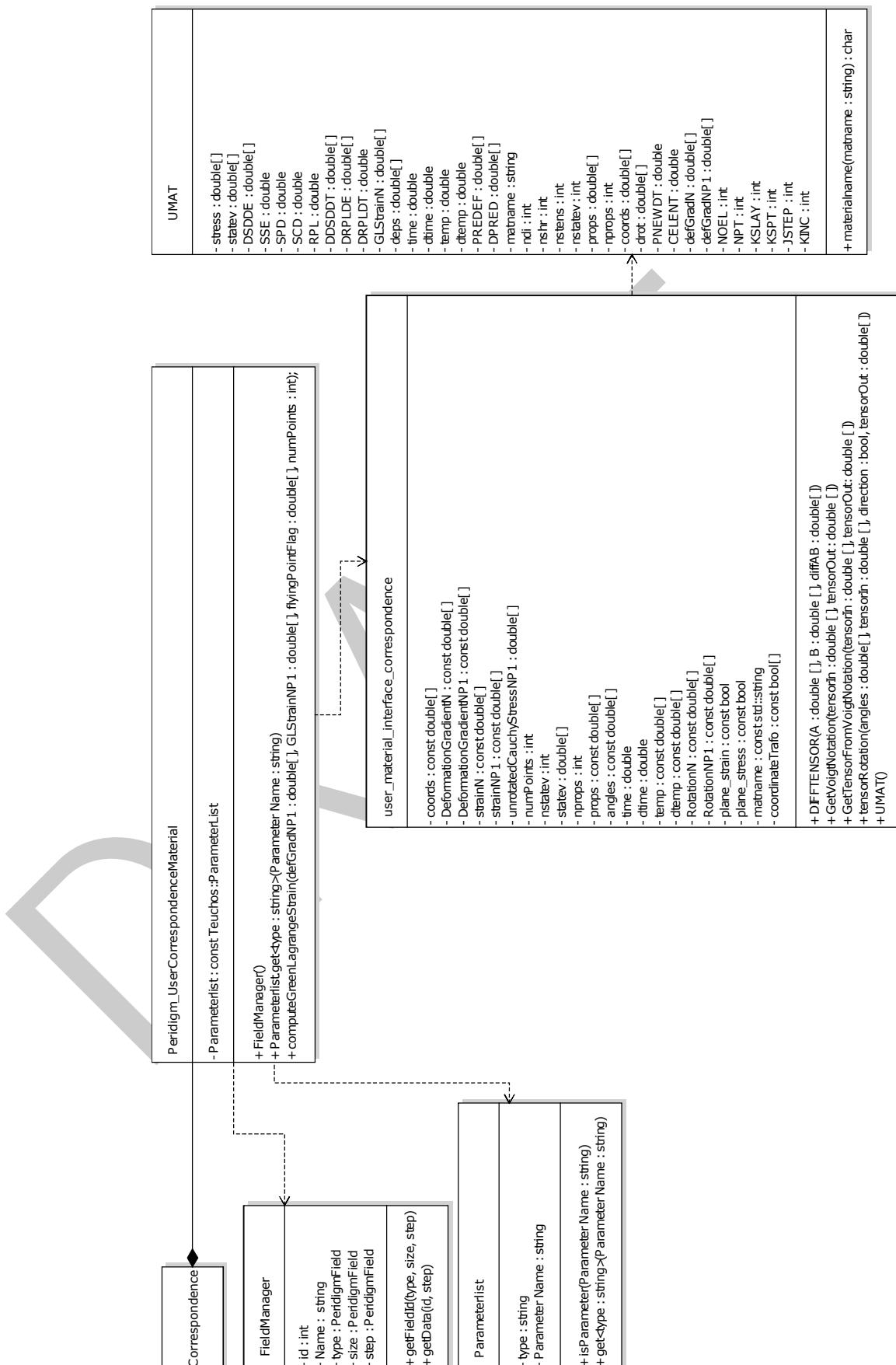


Figure 1: UML schema

74 Listing 1: Yaml interface to call the UMAT with name (User Material Name), three properties
 75 and three state variables.

```

76 Materials:
77   User Material Name:
78     Material Model: "User Correspondence"
79     Plane Strain: false
80     Plane Stress: True
81     Density: 2.7e+03
82     Young's Modulus: 7.24e+10
83     Poisson's Ratio: 3.3e-01
84     Number of Properties: 3
85     Prop_1: 1.0727111897390535e+11
86     Prop_2: 5.2835028748341446e+10
87     Prop_3: 2.721804511278195e+10
88     Number of State Vars: 3

```

89 Listing 2: Yaml interface export state parameter.

```

90 Output:
91   Output File Type: "ExodusII"
92   Output Filename: "Example"
93   Output Frequency: 1
94   Output Variables:
95     State_Parameter_Field_1: true
96     State_Parameter_Field_2: true
97     State_Parameter_Field_3: false
98     Displacements: true

```

99 Example

100 [Figure 2](#) shows the example provided in the repository. The user material is defined with
 101 properties the above properties. Property one is the P-wave modulus, property 2 is Lames first
 102 parameter and the third property is the Shear Modulus. The dogbone is loaded under tension
 103 by applying a $u_1(x_1 = l) = 0.01m$ displacement at the right-hand side. All translations on the
 104 boundary condition application region on the left of the specimen are fixed.

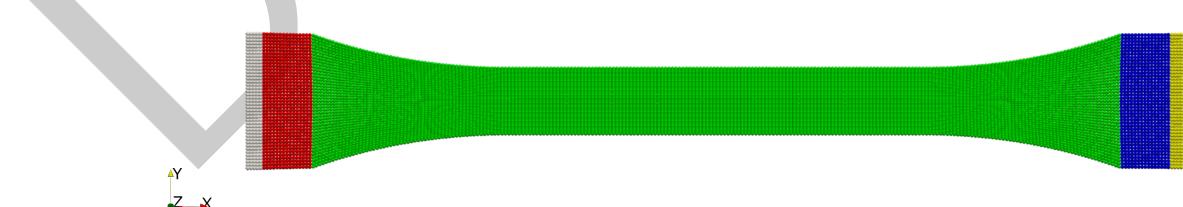


Figure 2: Dogbone block definition

105 The material routine is utilized in Peridigm and Abaqus. [Figure 3](#) and [Figure 4](#) shows
 106 the resulting displacement. As expected the results are identical. This is because the u_1
 107 displacement was applied as boundary condition. [Figure 5](#) and [Figure 6](#) illustrate the σ_{11} stress
 108 distribution. There are some differences between both results. The numerical representation
 109 between Peridigm and Abaqus is different. Boundary conditions cannot be applied in the same
 110 way, because each point in the peridynamic model represents a volume. Surface boundary
 111 conditions lead to an error. Another difference is not of numerical nature. The classical

¹¹² continuum mechanics theory and Peridynamics are different formulations and also if fully
¹¹³ converged will lead to minor differences.

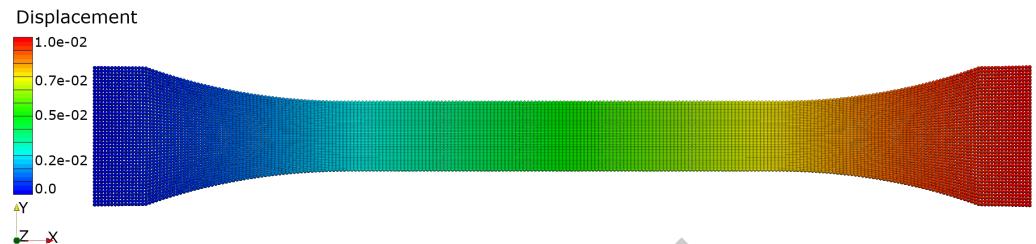


Figure 3: Resulting displacements using Peridigm

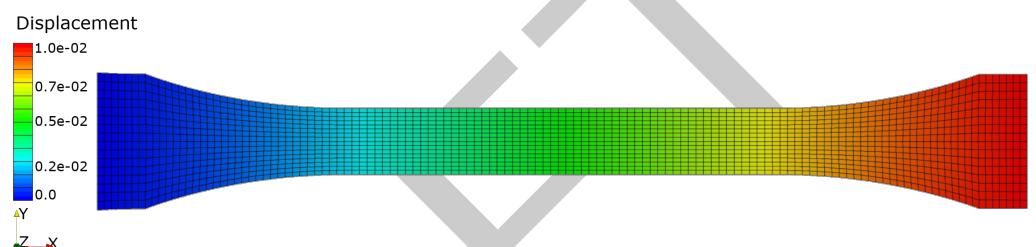


Figure 4: Resulting displacements using Abaqus

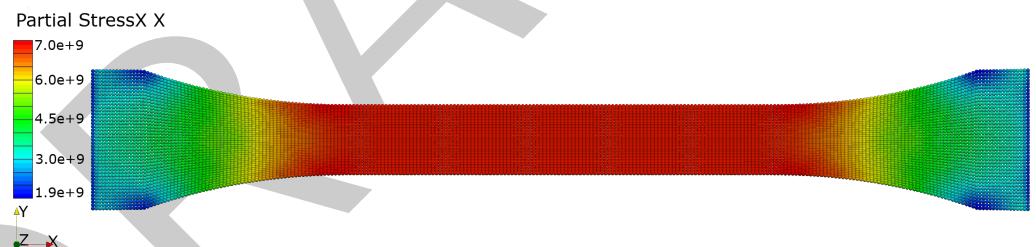


Figure 5: Resulting S11 stresses using Peridigm

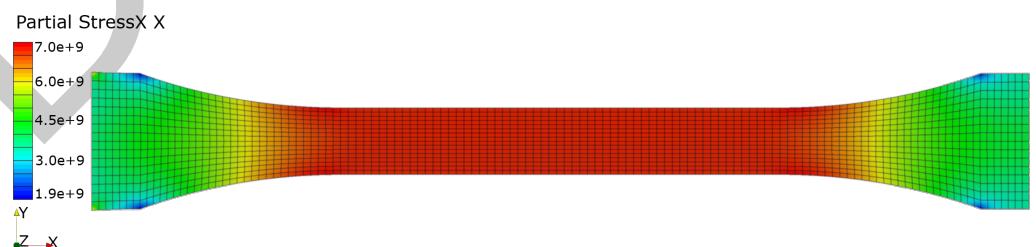


Figure 6: Resulting S11 stresses using Abaqus

¹¹⁴ Quality control

¹¹⁵ Multiple test routines are provided for the interface. They are based on the existing Peridigm
¹¹⁶ CMake test environment. In combination with the use of CTest the user can make sure that
¹¹⁷ the Fortran interface is working as expected.

¹¹⁸ Two unit tests and one functional test are implemented. The first unit test is required to
¹¹⁹ ensure that all tensors are translated into a Voigt notation and returned as a full tensor. The
¹²⁰ second unit test is able to control the correct passing of variables in and from the Fortran
¹²¹ interface. Therefore, a test user material library, which modifies every parameter by a defined
¹²² value, was compiled. If the returned values are as expected the Fortran interface and the
¹²³ property definition is working.

¹²⁴ In addition, full testing ensures that the user material implementation works across the Peridigm
¹²⁵ framework. Hence, the test will compare an exodus result file which is based on a predefined
¹²⁶ Peridigm material model and a file which is the result of the user material model. Both material
¹²⁷ models are similar. If the exodus results are within a defined tolerance the test will pass. As
¹²⁸ reference a dogbone model as shown in [Figure 2](#), the compiled fortran routine and the exodus
¹²⁹ result file can be found in the referenced Zenodo archive.

¹³⁰ Availability

¹³¹ Operating system

¹³² Linux

¹³³ Programming language

¹³⁴ C++, Fortran 90

¹³⁵ Additional system requirements

¹³⁶ 2 GHz (Dual Core), 2 GB RAM, 5 GB available space

¹³⁷ Dependencies

¹³⁸ Trilinos 13.2.0, HDF5 1.12.0, NetCDF 4.8.0, CMake 3.20.5

¹³⁹ List of contributors

¹⁴⁰ Christian Willberg, christian.willberg@dlr.de, German Aerospace Center
¹⁴¹ Jan-Timo Hesse, jan-timo.hesse@dlr.de, German Aerospace Center

¹⁴²

¹⁴³ Software location:

¹⁴⁴ Name:

¹⁴⁵ ▪ Zenodo

¹⁴⁶ Persistent identifier:

¹⁴⁷ ▪ <https://doi.org/10.5281/zenodo.6418265>

¹⁴⁸ License:

¹⁴⁹ ▪ BSD

¹⁵⁰ **Publisher:**

- ¹⁵¹ ▪ Jan-Timo Hesse

¹⁵² **Version published:**

- ¹⁵³ ▪ 0.1

¹⁵⁴ **Date published:**

- ¹⁵⁵ ▪ 06/04/2022

¹⁵⁶ **Code repository**

¹⁵⁷ **Name:**

- ¹⁵⁸ ▪ GitHub

¹⁵⁹ **Persistent identifier:**

- ¹⁶⁰ ▪ <https://github.com/PeriHub/Peridigm>

¹⁶¹ **License:**

- ¹⁶² ▪ BSD

¹⁶³ **Date published:**

- ¹⁶⁴ ▪ 04/04/2022

¹⁶⁵ **Language**

- ¹⁶⁶ ▪ English

¹⁶⁷ **Reuse potential**

¹⁶⁸ A possibility to directly apply user defined material models in Peridigm eliminates the need
¹⁶⁹ for duplicate development of existing material models from classical continuum mechanics.
¹⁷⁰ Many material models can now be used both in finite element methods and in Peridynamics.
¹⁷¹ This opens up new analysis, verification and comparison possibilities. The application of the
¹⁷² UMAT interface is done by a pre-compiled Fortran material routine. This is deposited in the
¹⁷³ calculation folder. This library is copied to the correct place and used for the calculation of
¹⁷⁴ the material.

¹⁷⁵ **Acknowledgements**

¹⁷⁶ The authors like to acknowledge the development team of the original Peridigm framework
¹⁷⁷ (David J. Littlewood, djlittl@sandia.gov, John A. Mitchell, jamitch@sandia.gov, Michael L.
¹⁷⁸ Parks, mlparks@sandia.gov, Stewart A. Silling, sasilli@sandia.gov) ([Parks et al., 2012](#)).

¹⁷⁹ **Funding statement**

¹⁸⁰ The work was funded by the German Research Foundation funded project: "Gekoppelte
¹⁸¹ Peridynamik-Finite-Elemente-Simulationen zur Schädigungsanalyse von Faserverbundstruktu-
¹⁸² ren" Grant number: WI 4835/5-1 and the M-ERA.NET funded project Exploring Multi-
¹⁸³ Method Analysis of composite structures and joints under consideration of uncertainties

¹⁸⁴ engineering and processing (EMMA)



¹⁸⁵
¹⁸⁶ This measure is co-financed with tax funds on the basis of the budget passed by the Saxon
¹⁸⁷ state parliament. Grant number: 3028223. The authors like to thank for the funding.

¹⁸⁸ Competing interests

¹⁸⁹ The authors declare that they have no competing interests.

¹⁹⁰

¹⁹¹ Copyright Notice

¹⁹² Authors who publish with this journal agree to the following terms:

¹⁹³ Authors retain copyright and grant the journal right of first publication with the work simulta-
¹⁹⁴ neously licensed under a [Creative Commons Attribution License](#) that allows others to share the
¹⁹⁵ work with an acknowledgement of the work's authorship and initial publication in this journal.
¹⁹⁶ Authors are able to enter into separate, additional contractual arrangements for the non-exclusive
¹⁹⁷ distribution of the journal's published version of the work (e.g., post it to an institutional
¹⁹⁸ repository or publish it in a book), with an acknowledgement of its initial publication in this
¹⁹⁹ journal.

²⁰⁰ By submitting this paper you agree to the terms of this Copyright Notice, which will apply to
²⁰¹ this submission if and when it is published by this journal.

²⁰² (2022, March). online.

²⁰³ Parks, M. L., Littlewood, D. J., Mitchell, J. A., & Silling, S. A. (2012). *Peridigm users' guide*.
²⁰⁴ Report SAND2012-7800, Sandia National Laboratories.

²⁰⁵ Rädel, M., & Willberg, C. (2018). *PeriDoX*. GitHub repository; GitHub. <https://doi.org/10.5281/zenodo.1403015>

²⁰⁶
²⁰⁷ Silling, S. A., Epton, M., Weckner, O., Xu, J., & Askari, E. (2007). Peridynamic states
²⁰⁸ and constitutive modeling. *Journal of Elasticity*, 88, 151–184. <https://doi.org/10.1007/s10659-007-9125-1>

²⁰⁹
²¹⁰ UMAT. (2022, March). Dassault; online. <https://abaqus-docs.mit.edu/2017/English/SIMACAESUBRefMap/simasub-c-umat.htm#simasub-c-umat>

²¹¹
²¹² Willberg, C., & Heinecke, F. (2021). Evaluation of manufacturing deviations of composite
²¹³ materials. *PAMM*, 20(1), e202000345. <https://doi.org/10.1002/pamm.202000345>

²¹⁴
²¹⁵ Willberg, C., Hesse, J.-T., & Heinecke, F. (2022). Peridynamic simulation of a mixed-mode frac-
²¹⁶ ture experiment in PMMA utilizing an adaptive-time stepping for an explicit solver. *Journal of Peridynamics and Nonlocal Modeling*. <https://doi.org/10.1007/s42102-021-00079-6>

²¹⁷
²¹⁸ Willberg, C., Wiedemann, L., & Rädel, M. (2019). A mode-dependent energy-based damage
²¹⁹ model for peridynamics and its implementation. *Journal of Mechanics of Materials and Structures*, 14(2), 193–217. <https://doi.org/10.2140/jomms.2019.14.193>