The Journal of Open Source Software

FElupe: Finite element analysis for continuum mechanics of solid bodies

Andreas Dutzler [1,2¶] **and Martin Leitner** [1]

1 Institute of Structural Durability and Railway Technology, Graz University of Technology, Austria **2** Siemens Mobility Austria GmbH, Austria ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

FElupe is a Python package for finite element analysis focusing on the formulation and numerical solution of nonlinear problems in continuum mechanics of solid bodies. This package is intended for scientific research, but is also suitable for running nonlinear simulations in general. In addition to the transformation of general weak forms into sparse vectors and matrices, FElupe provides an efficient high-level abstraction layer for the simulation of the deformation of solid bodies.

Highlights

- 100% Python package built with NumPy and SciPy
- easy to learn and productive high-level API
- nonlinear deformation of solid bodies
- interactive views on meshes, fields and solid bodies
- typical finite elements
- cartesian, axisymmetric, plane strain and mixed fields
- hyperelastic material models
- strain energy density functions with automatic differentiation

Efficient NumPy-based math is realized by element-wise operating trailing axes (Gustafsson & McBain, 2020). The finite element method, as used in FElupe, is based on (Bonet & Wood, 2008), (Bathe, 2006) and (Zienkiewicz, 2013). Interactive views are enabled by PyVista (Sullivan & Kaszynski, 2019). The capabilities of FElupe may be enhanced with additional Python packages, e.g. `meshio` (Schlömer, 2024), `matadi` (Dutzler, 2024b), `tensortrax` (Dutzler, 2024c), `hyperelastic` (Dutzler, 2024a) or `feplot` (Mohamed ZAARAOUI, 2023).

The essential high-level parts of solving problems with FElupe include a field, a solid body, boundary conditions and a job. A field for a field container is created by a mesh, a numeric region and a quadrature scheme, see Figure 1. In a solid body, this field container is combined with a constitutive material formulation. Along with constant and ramped boundary conditions a step is created. During job evaluation, the field values are updated in-place as shown in Figure 2.

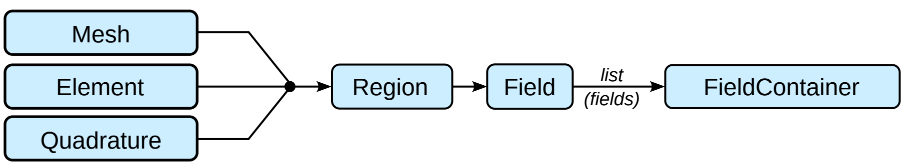

Figure 1: Schematic representation of classes needed to create a field container.

The Journal of Open Source Software

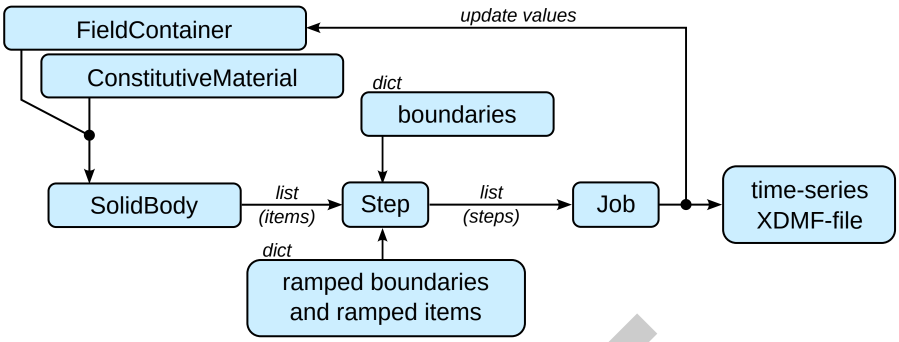

Figure 2: Schematic representation of classes needed to evaluate a job.

For example, consider a quarter model of a solid cube with hyperelastic material behaviour subjected to a uniaxial elongation applied at a clamped end-face. First, a meshed cube out of hexahedron cells is created. A numeric region, pre-defined for hexahedrons, is created on the mesh. The appropriate finite element and its quadrature scheme are chosen automatically. A vector-valued displacement field is initiated on the region and is further added to a field container. A uniaxial load case is applied on the displacement field to create the boundary conditions. This involves setting up symmetry planes as well as the absolute value of the prescribed displacement at the mesh-points on the right-end face of the cube. The right-end face is clamped, i.e. its displacements, except the components in longitudinal direction, are fixed. An isotropic hyperelastic Neo-Hookean material formulation is applied on the displacement field of a solid body. A step generates the consecutive substep-movements of a selected boundary condition. The step is further added to a list of steps of a job. After the job evaluation is completed, the maximum principal values of logarithmic strain of the last completed substep are plotted, see Figure 3.

```python
import felupe as fem

region = fem.RegionHexahedron(mesh=fem.Cube(n=6))
field = fem.FieldContainer([fem.Field(region, dim=3)])
solid = fem.SolidBody(umat=fem.NeoHooke(mu=1, bulk=2), field=field)
boundaries, loadcase = fem.dof.uniaxial(field, clamped=True)

move = fem.math.linsteps([0, 1], num=5)
step = fem.Step([solid], ramp={boundaries["move"]: move}, boundaries=boundaries)
job = fem.Job(steps=[step]).evaluate()

solid.plot("Principal Values of Logarithmic Strain").show()
```

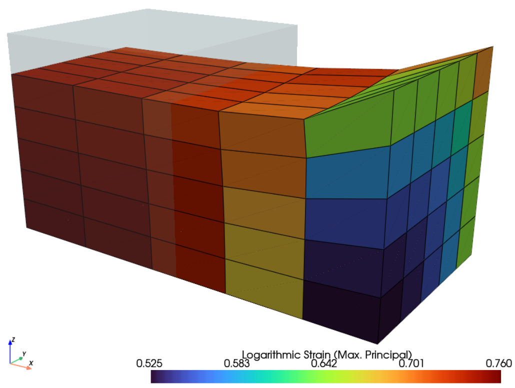

Figure 3: Final logarithmic strain distribution of the deformed hyperelastic solid body at $l/L = 2$.

Any other hyperelastic material model formulation may be used instead of the Neo-Hookean material model given above, most easily by its strain energy density function. The strain energy density function of the Mooney-Rivlin material model formulation, as given in Equation 1, is implemented by a hyperelastic material class in FElupe.

$$\psi(C) = C_{10} \left(\hat{I}_1 - 3\right) + C_{01} \left(\hat{I}_2 - 3\right) \tag{1}$$

```python
import tensortrax.math as tm

def mooney_rivlin(C, C10, C01):
    I1 = tm.trace(C)
    I2 = (I1**2 - tm.trace(C @ C)) / 2
    I3 = tm.linalg.det(C)
    return C10 * (I3**(-1/3) * I1 - 3) + C01 * (I3**(-2/3) * I2 - 3)

umat = fem.Hyperelastic(mooney_rivlin, C10=0.5, C01=0.1)
solid = fem.SolidBody(umat=umat, field=field)
```

Examples

The documentation of FElupe contains interactive tutorials and examples for simulating the deformation of solid bodies. Resulting deformed solid bodies of selected examples are shown in Figure 4. Computational results of FElupe are used in several scientific publications, e.g. (Dutzler et al., 2021), (Torggler et al., 2023).

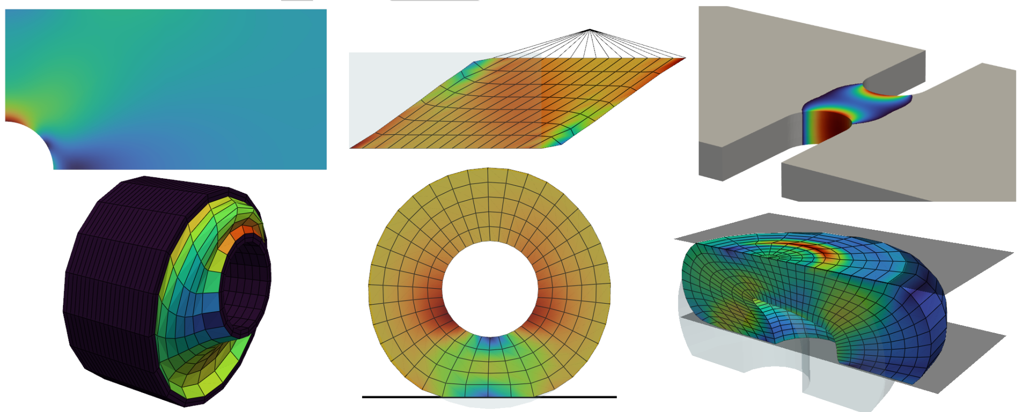

Figure 4: Equivalent stress distribution of a plate with a hole (top left). Shear-loaded hyperelastic block (top middle). Endurable cycles obtained by local stresses (top right). Multiaxially loaded rubber bushing (bottom left). Rotating rubber wheel on a frictionless contact (bottom middle). A hyperelastic solid with frictionless rigid contacts (bottom right).

References

Bathe, K.-J. (2006). *Finite element procedures*. Bathe. ISBN: 9780979004902

Bonet, J., & Wood, R. D. (2008). *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press. https://doi.org/10.1017/cbo9780511755446

Dutzler, A. (2024a). *Hyperelastic: Constitutive hyperelastic material formulations for FElupe*. Zenodo. https://doi.org/10.5281/zenodo.8106469

Dutzler, A. (2024b). *matADi: Material definition with automatic differentiation*. Zenodo. https://doi.org/10.5281/zenodo.5519971

Dutzler, A. (2024c). *Tensortrax: Math on (hyper-dual) tensors with trailing axes*. Zenodo. https://doi.org/10.5281/zenodo.7384105

Dutzler, A., Buzzi, C., & Leitner, M. (2021). Nondimensional translational characteristics of elastomer components. *Journal of Applied Engineering Design and Simulation*, *1*(1). https://doi.org/10.24191/jaeds.v1i1.20

Gustafsson, T., & McBain, G. (2020). Scikit-fem: A python package for finite element assembly. *Journal of Open Source Software*, *5*(52), 2369. https://doi.org/10.21105/joss.02369

Mohamed ZAARAOUI. (2023). *ZAARAOUI999/feplot: v0.1.13*. Zenodo. https://doi.org/10.5281/zenodo.10429691

Schlömer, N. (2024). *Meshio: Tools for mesh files*. Zenodo. https://doi.org/10.5281/zenodo.1173115

Sullivan, C., & Kaszynski, A. (2019). PyVista: 3D plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK). *Journal of Open Source Software*, *4*(37), 1450. https://doi.org/10.21105/joss.01450

Torggler, J., Dutzler, A., Oberdorfer, B., Faethe, T., Müller, H., Buzzi, C., & Leitner, M. (2023). Investigating damage mechanisms in cord-rubber composite air spring bellows of rail vehicles and representative specimen design. *Applied Composite Materials*, *30*(6), 1979–1999. https://doi.org/10.1007/s10443-023-10157-1

Zienkiewicz, O. C. (2013). *Finite element method: Its basis and fundamentals* (R. L. Taylor, J. Zhu, & O. C. Zienkiewicz, Eds.; 7th ed.). Elsevier Science & Technology. ISBN: 9780080951355