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

DOI: 10.21105/joss.09160

Software

- Review
- Repository
- Archive

Editor: Kyle Niemeyer

Reviewers:

- @kyleniemeyer

Submitted: 28 September 2025
Published: 07 October 2025

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. The finite element method, as used in FElupe, is generally based on the preliminary works by Bonet & Wood (2008), Bathe (2006) and Zienkiewicz (2013).

Highlights

- pure Python package built with NumPy and SciPy
- easy-to-learn and productive high-level API
- nonlinear deformation of solid bodies with interactive views
- hyperelastic material models with automatic differentiation

Statement of need

There are well-established Python packages available for finite element analysis. These packages are either distributed as binary packages or need to be compiled on installation, like FEniCSx (Baratta et al., 2023), GetFEM (Renard & Poulios, 2020) or SfePy (Cimrman et al., 2019). JAX-FEM (Xue et al., 2023), which is built on JAX (Bradbury et al., 2018), is a pure Python package but requires many dependencies in its recommended environment. `scikit-fem` (Gustafsson & McBain, 2020) is a pure Python package with minimal dependencies but with a more general scope (Gustafsson & McBain, 2020). FElupe is both easy-to-install as well as easy-to-use in its target domain of hyperelastic solid bodies.

The performance of FElupe is good for a non-compiled package and it is well-suited for up to mid-sized problems, i.e., up to 10^5 degrees of freedom, when hyperelastic model formulations are used. A performance benchmark for times spent on stiffness matrix assembly is included in the documentation. Internally, efficient NumPy (Harris et al., 2020) based math is realized by element-wise operating trailing axes (Gustafsson & McBain, 2020). An all-at-once approach per operation is used instead of a cell-by-cell evaluation loop. The constitutive material formulation class is backend agnostic: FElupe provides NumPy-arrays as input arguments and requires NumPy-arrays as return values. This enables backends like JAX (Bradbury et al., 2018) or PyTorch (Ansel et al., 2024) to be used. Interactive views of meshes, fields and solid bodies 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).

Features

The essential high-level parts of solving problems with FElupe include a field, a solid body, boundary conditions and a job. With the combination of a mesh, a finite element formulation and a quadrature rule, a numeric region is created. A field for a field container is further created on top of this numeric region, see Figure 1.

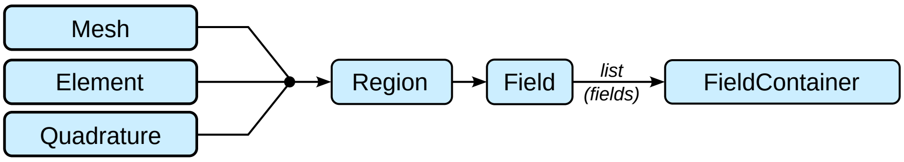

Figure 1: Schematic representation of classes needed to create a field container.

In a solid body, a constitutive material formulation is applied on this field container. Along with constant and ramped boundary conditions a step is created. During job evaluation, the field values are updated in-place after each completed substep as shown in Figure 2.

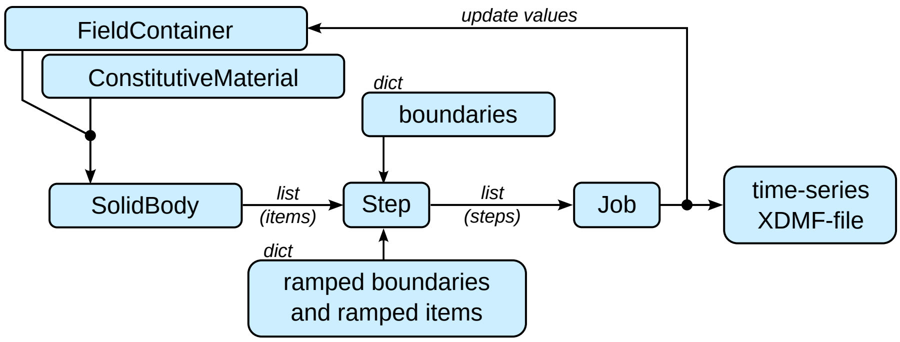

Figure 2: Schematic representation of classes needed to evaluate a job.

For example, consider a quarter model of a solid cube with nearly-incompressible hyperelastic material behavior 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 are fixed, except for the components in longitudinal direction. An isotropic hyperelastic Neo-Hookean material formulation (Bonet & Wood, 2008; Treloar, 1943) 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)])
umat = fem.NeoHooke(mu=1)
solid = fem.SolidBodyNearlyIncompressible(umat=umat, field=field, bulk=5000)
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()
```

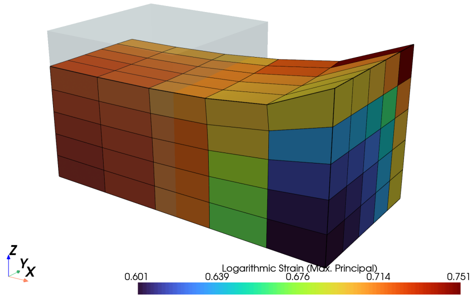

Figure 3: Final logarithmic strain distribution of the deformed hyperelastic solid body at a stretch $l/L = 2$, where l is the deformed length and L the undeformed length of the solid body in longitudinal direction. The undeformed configuration is shown in transparent grey.

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 (Mooney, 1940; Rivlin & Saunders, 1951), as given in Equation 1, is implemented by a hyperelastic material class in FElupe with the help of tensortrax (bundled with FElupe).

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

```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.SolidBodyNearlyIncompressible(umat=umat, field=field, bulk=5000)
```

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 (Buzzi et al., 2022; Dutzler et al., 2021; Torggler et al., 2023).

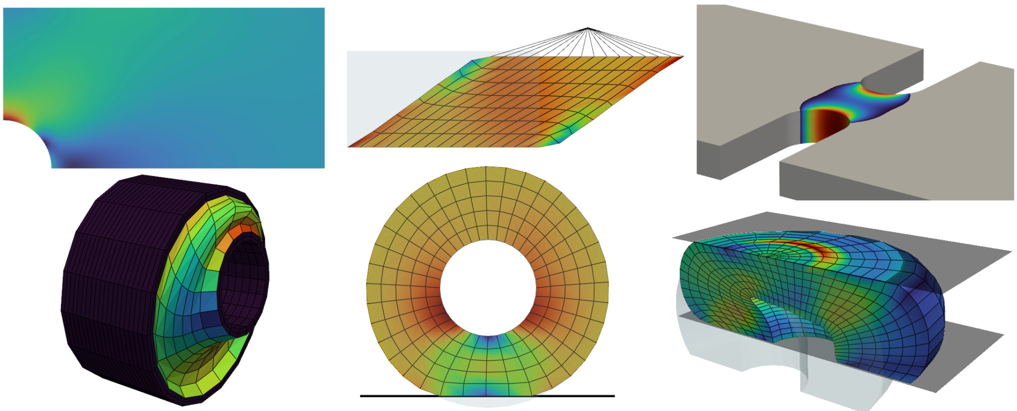

Figure 4: Equivalent stress distribution of a plate with a hole (top left). Shear-loaded hyperelastic block (top middle). Endurable cycles obtained by local stresses (top right). Multiaxially loaded rubber bushing (bottom left). Rotating rubber wheel on a frictionless contact (bottom middle). A hyperelastic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

References

Ansel, J., Yang, E., He, H., Gimelshein, N., Jain, A., Voznesensky, M., Bao, B., Bell, P., Berard, D., Burovski, E., Chauhan, G., Chourdia, A., Constable, W., Desmaison, A., DeVito, Z., Ellison, E., Feng, W., Gong, J., Gschwind, M., … Chintala, S. (2024). PyTorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, *5*, 929–947. https://doi.org/10.1145/3620665.3640366

Baratta, I. A., Dean, J. P., Dokken, J. S., Habera, M., Hale, J. S., Richardson, C. N., Rognes, M. E., Scroggs, M. W., Sime, N., & Wells, G. N. (2023). *DOLFINx: The next generation FEniCS problem solving environment*. Zenodo. https://doi.org/10.5281/zenodo.10447666

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

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/jax-ml/jax

Buzzi, C., Dutzler, A., Faethe, T., Lassacher, J., Leitner, M., & Weber, F.-J. (2022). Development of a tool for estimating the characteristic curves of rubber-metal parts. In A. Szabó (Ed.), *Proceedings of the 12th international conference on railway bogies and running gears* (pp. 191–200). Scientific Society for Mechanical Engineering. ISBN: 978-963-9058-46-0

Cimrman, R., Lukeš, V., & Rohan, E. (2019). Multiscale finite element calculations in python using SfePy. *Advances in Computational Mathematics*, *45*(4), 1897–1921. https://doi.org/10.1007/s10444-019-09666-0

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

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

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

Mooney, M. (1940). A theory of large elastic deformation. *Journal of Applied Physics*, *11*(9), 582–592. https://doi.org/10.1063/1.1712836

Renard, Y., & Poulios, K. (2020). *GetFEM: Automated FE modeling of multiphysics problems based on a generic weak form language*. https://hal.science/hal-02532422

Rivlin, R. S., & Saunders, D. W. (1951). Large elastic deformations of isotropic materials VII. Experiments on the deformation of rubber. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, *243*(865), 251–288. https://doi.org/10.1098/rsta.1951.0004

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

Treloar, L. R. G. (1943). The elasticity of a network of long-chain molecules—II. *Transactions of the Faraday Society*, *39*(0), 241–246. https://doi.org/10.1039/tf9433900241

Xue, T., Liao, S., Gan, Z., Park, C., Xie, X., Liu, W. K., & Cao, J. (2023). JAX-FEM: A differentiable GPU-accelerated 3D finite element solver for automatic inverse design and mechanistic data science. *Computer Physics Communications*, *291*, 108802. https://doi.org/10.1016/j.cpc.2023.108802

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