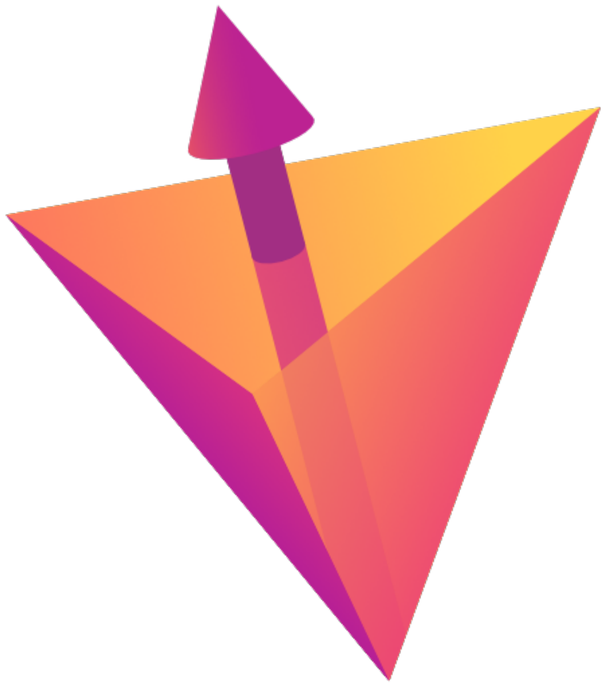


TetraX

tetrax.readthedocs.org/

TetraX Micromagnetic Modeling Package

Lukas Körber[1,2], Gwendolyn Quasebarth[1,2], Alexander Hempel[1,2], Andreas Otto[2],
Jürgen Fassbender[1,2] and Attila Kákay[1]

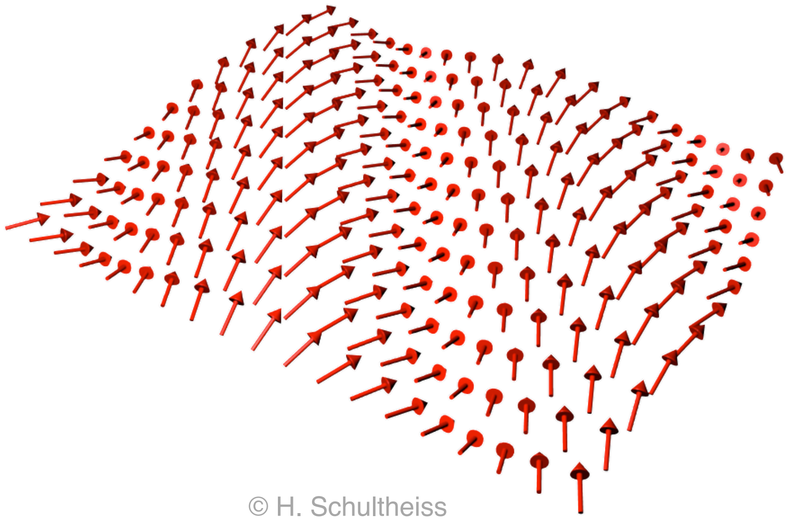
[1]Helmholtz-Zentrum Dresden - Rossendorf, Institut of Ion Beam Physics and Materials Research; [2]Faculty of Physics, Technical University Dresden

TetraX: An open-source finite-element package for (micro)magnetism

TetraX
tetrax.readthedocs.org

HZDR RSE

spin wave in a ferromagnet

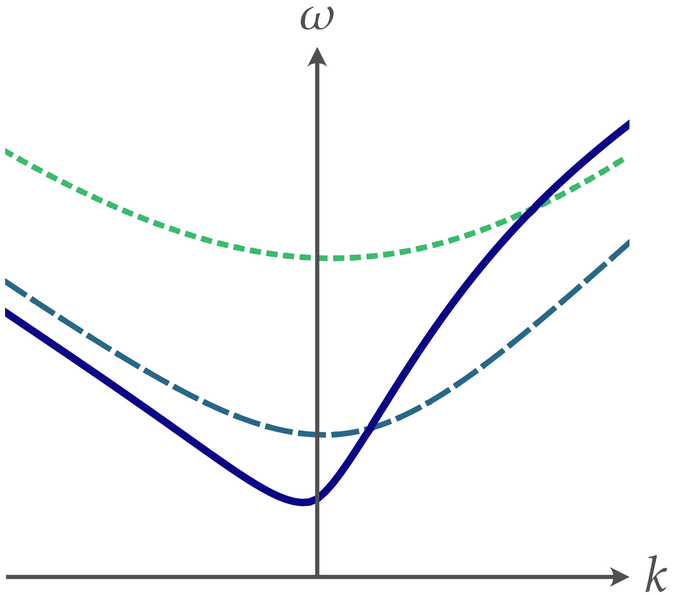


© H. Schultheiss

TetraX

Simulation
Evaluation
Visualization

Energies/dispersion



… and many other things

l.koerber@hzdr.de

Where to find our code

TetraX tetrax.readthedocs.org

HZDR RSE

TetraX Getting started Examples Publications User Guide API Reference More

Section Navigation

User Guide

1. Introduction
2. Installation
3. Creating and defining a sample
4. Numerical Experiments
5. Visualization and evaluation
6. Appendix

Note

During the course of this User Guide, volumentric $\mathbf{m}_\nu$ and lateral profiles $\mathbf{m}_{\nu k}$ are often interchangeably denoted simply as "mode profiles", unless stated otherwise.

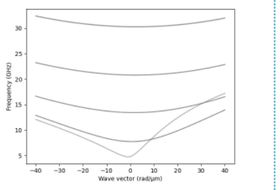
In *TetraX*, the eigensystem of a given `sample` in an experimental setup `exp` can be calculated by

```
>>> dispersion = exp.eigenmodes(...)
```

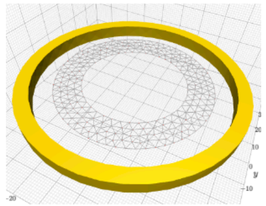
or simply

```
>>> exp.eigenmodes(...)
```

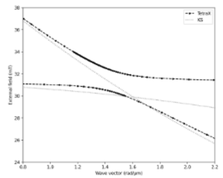
By default, the oscillation frequencies and the mode profiles are saved into the directory of the experimental setup.



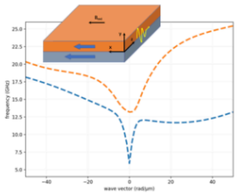
Dispersion of a nanotube in vortex state



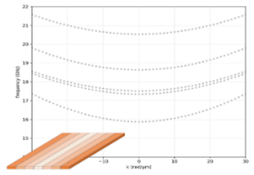
Absorption of a nanotube with antennae



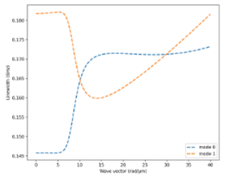
Thick YIG film - spin-wave hybridization



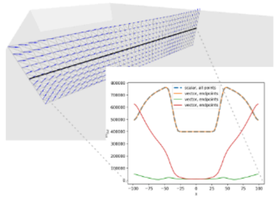
Double layers of Py / CoFeB



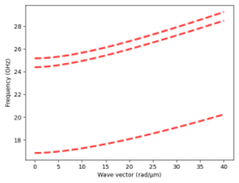
Magnetization-graded waveguides



Thick film dispersion and linewidth analysis



Linescan example on the magnetization-



Rectangular waveguide disp. in

Source: codebase.helmholtz.cloud/micromagnetic-modeling/tetrax

Docs: tetrax.readthedocs.org

Map of *confirmed* users

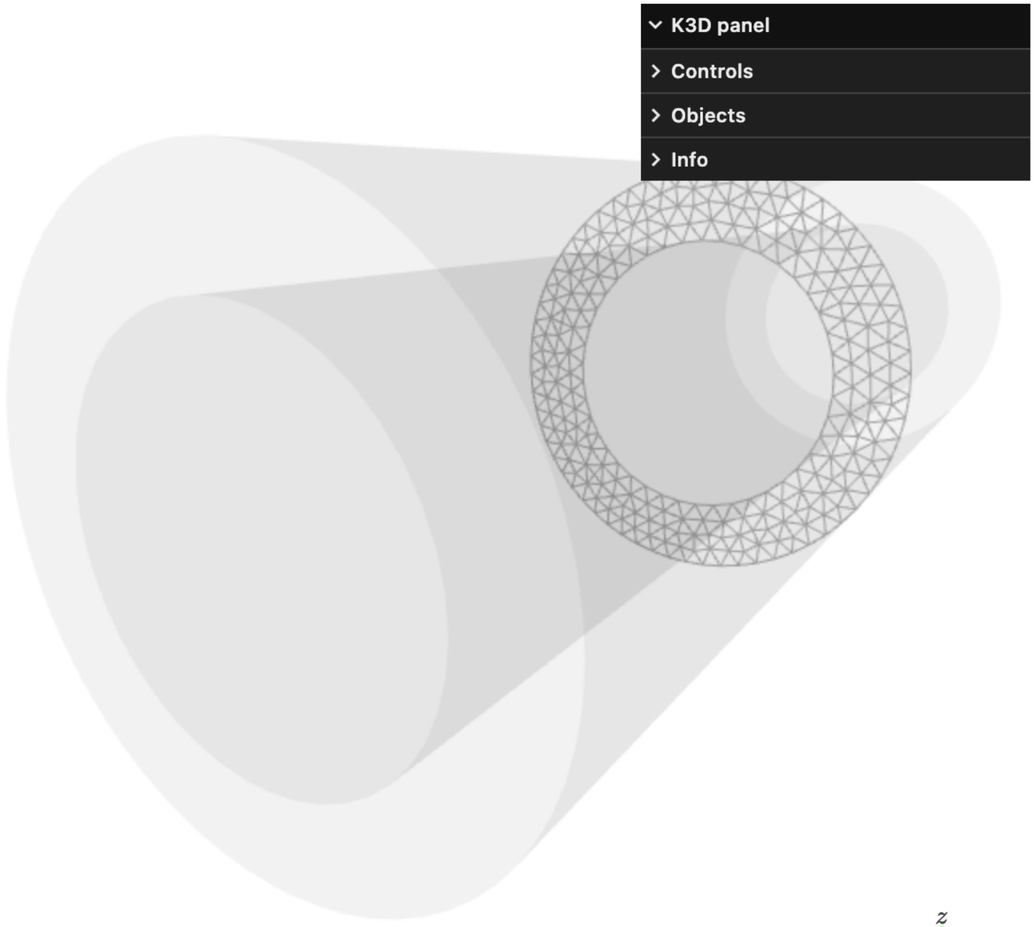
Hifis is helping us already!
(continuous integration, deployment, …)

Basics

- languages:
  - Python (numpy, scipy, pandas, meshio, pygmsh, joblib, etc.)
  - C, Cython
- FEM discretization all self-written
- object-oriented

Example workflow (Jupyter notebook)

```
1 | sample = tetrax.create_sample(geometry_type,
2 |                               magnetic_order,
3 |                               …)
3 |
4 | sample.Msat = 830e3 # material parameters
5 | sample.Aex = 13e-12
7 |
8 | sample.set_geom(…)
9 | sample.show(…)
```



Challenge

- Number of features have grown a lot
- Refactoring necessary to make code cleaner, more maintainable and expandable
- Avoid duplicate code, sleeker classes, etc…

TetraX
tetrax.readthedocs.org

HZDR RSE

Example problem:

- Sample class is bulky and has too many responsibilities
  - …
  - meshing/setting geometry /preprocessing
  - handling material parameters
  - visualization

```
1  class AbstractSample():
2
3      def set_geom(): …
3
4          self.nx
5          self.mesh
7          …
8
9      @property
10     def param1():
11         self._param1 = …
12
13
14     def show(): …
```

delegate responsibilities

→ `class SampleMesh()`

→ `class Material()`

l.koerber@hzdr.de

Challenge: How do you guys manage material parameters?

TetraX
tetrax.readthedocs.org

HZDR RSE

```
@property
def param1():
    self._param1 = …
```

???

```
class Material()
```

First intuition: dict… However…

- set of parameters depends on whether ferromagnet or antiferromagnet
- scalar or vector
- `vector_param1` can be supplied as single vector or vector field
- some are derived quantities (read-only)

should be easily accessible by the user (e.g. `sample[„param1“] = …`)