ubermag

UNIVERSITY OF Southampton

European XFEL

THE SKYRMION PROJECT

OPEN DREAMKIT

This is a PDF of the presentation and does not contain animations. If you require a full MacOS Keynote version, please ask Marijan.

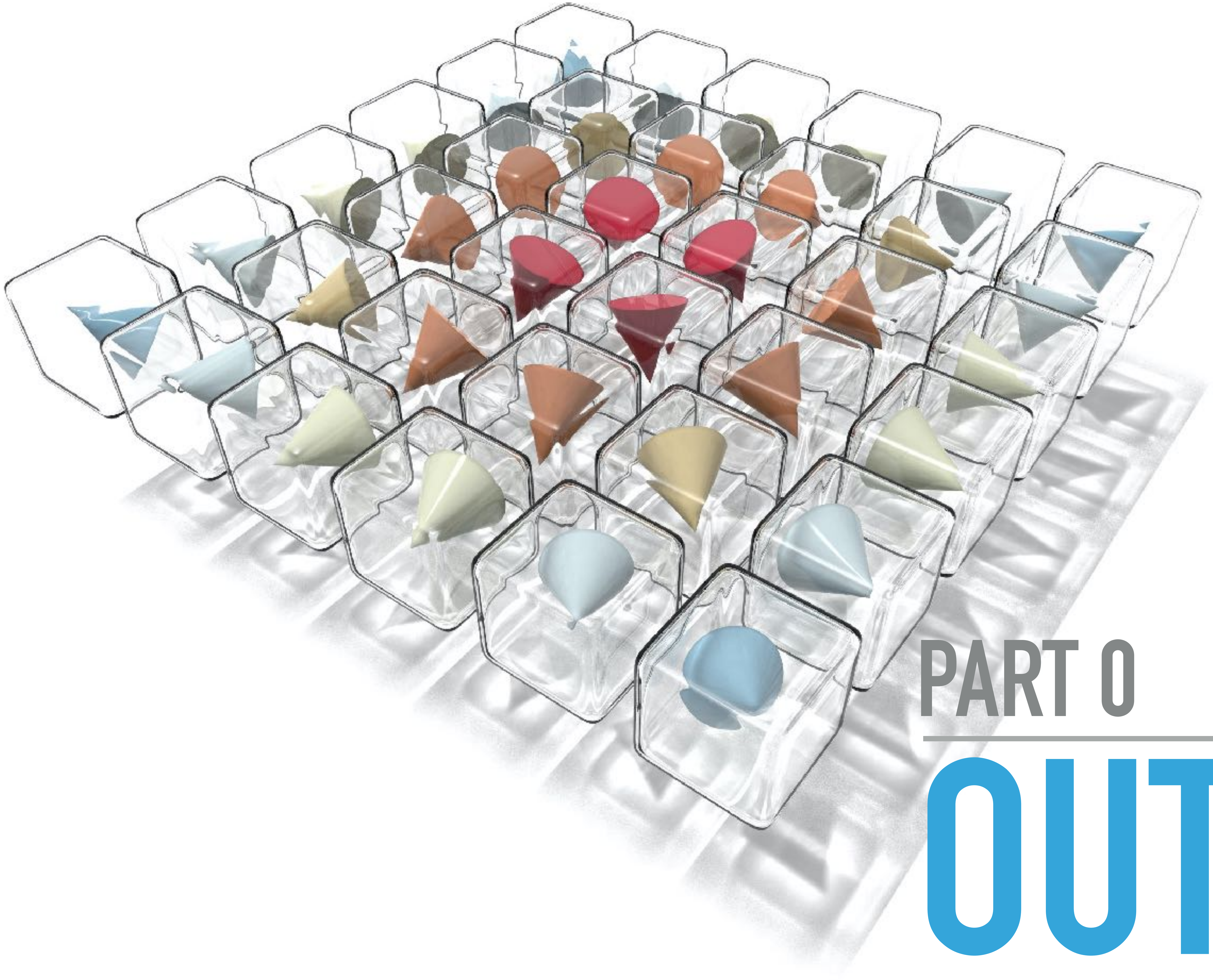
MARIJAN BEG[1]*

UBERMAG: INTERACTIVE MICROMAGNETIC SIMULATIONS IN JUPYTER

[1]University of Southampton, Highfield, SO17 1BJ Southampton, United Kingdom
*m.beg@soton.ac.uk

EPSRC
Engineering and Physical Sciences
Research Council

Warwick Seminar Series, 19 October 2020



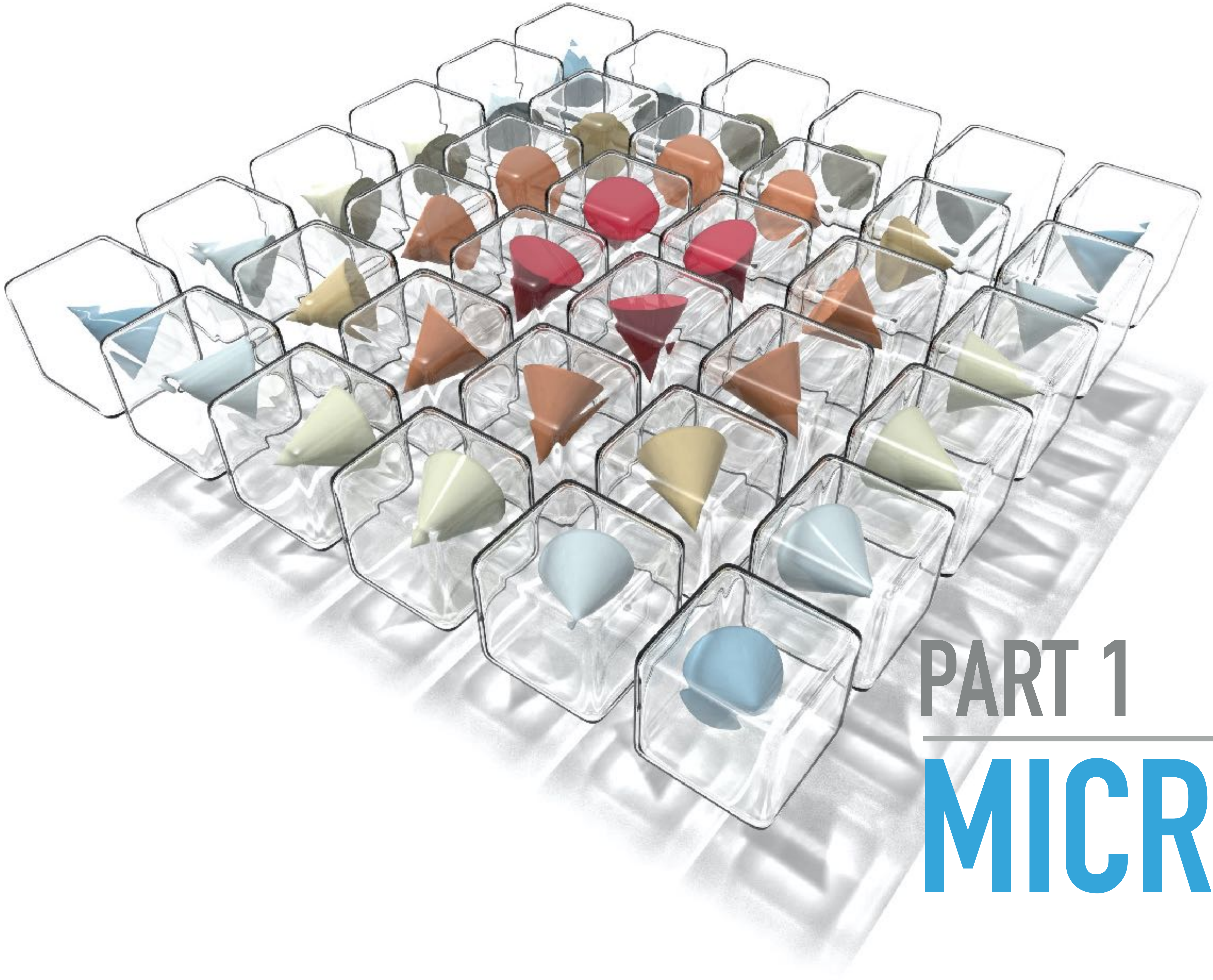
PART 0

OUTLINE

OUTLINE

1. Micromagnetics basics
2. OOMMF
3. Typical computational workflow
4. Ubermag
5. Demo
6. Discussion and Summary

ubermag



PART 1

MICROMAGNETICS

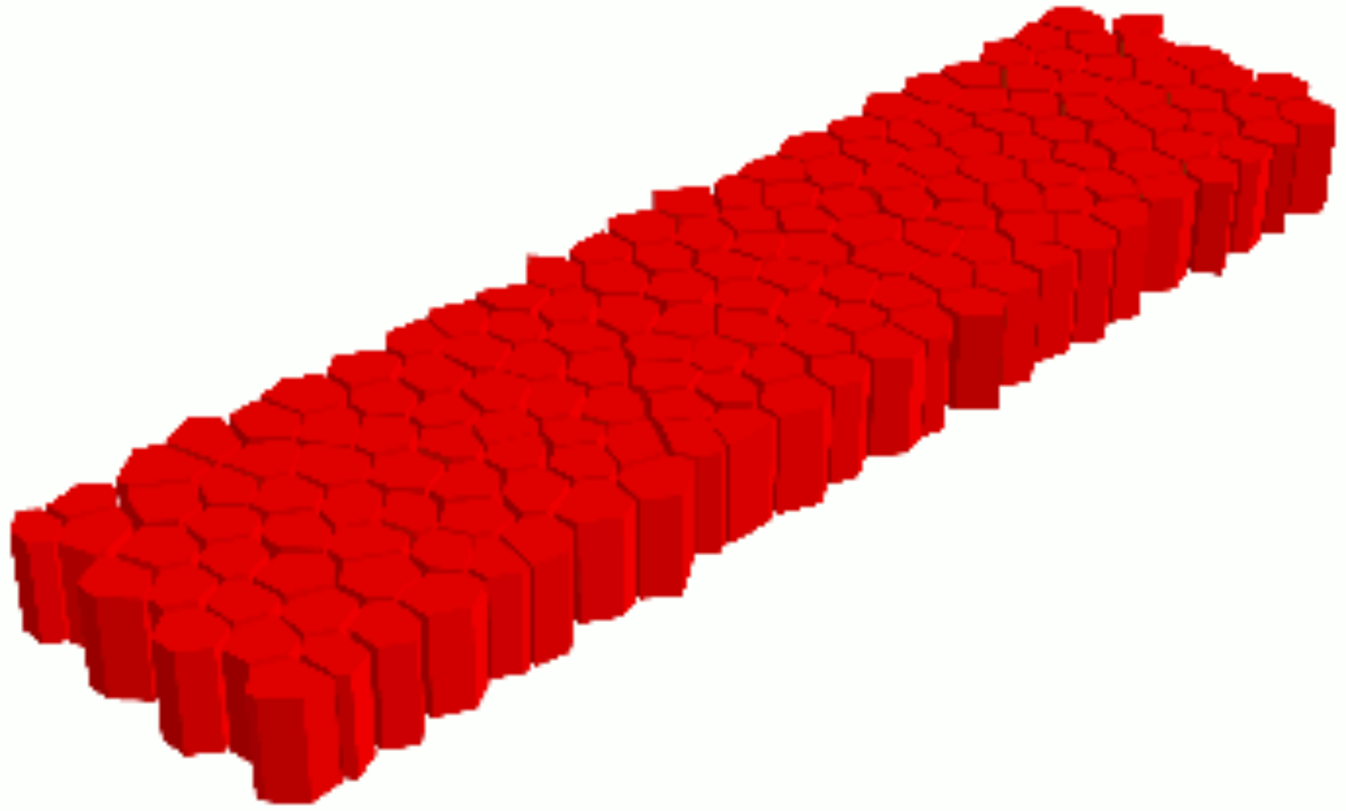
MICROMAGNETICS

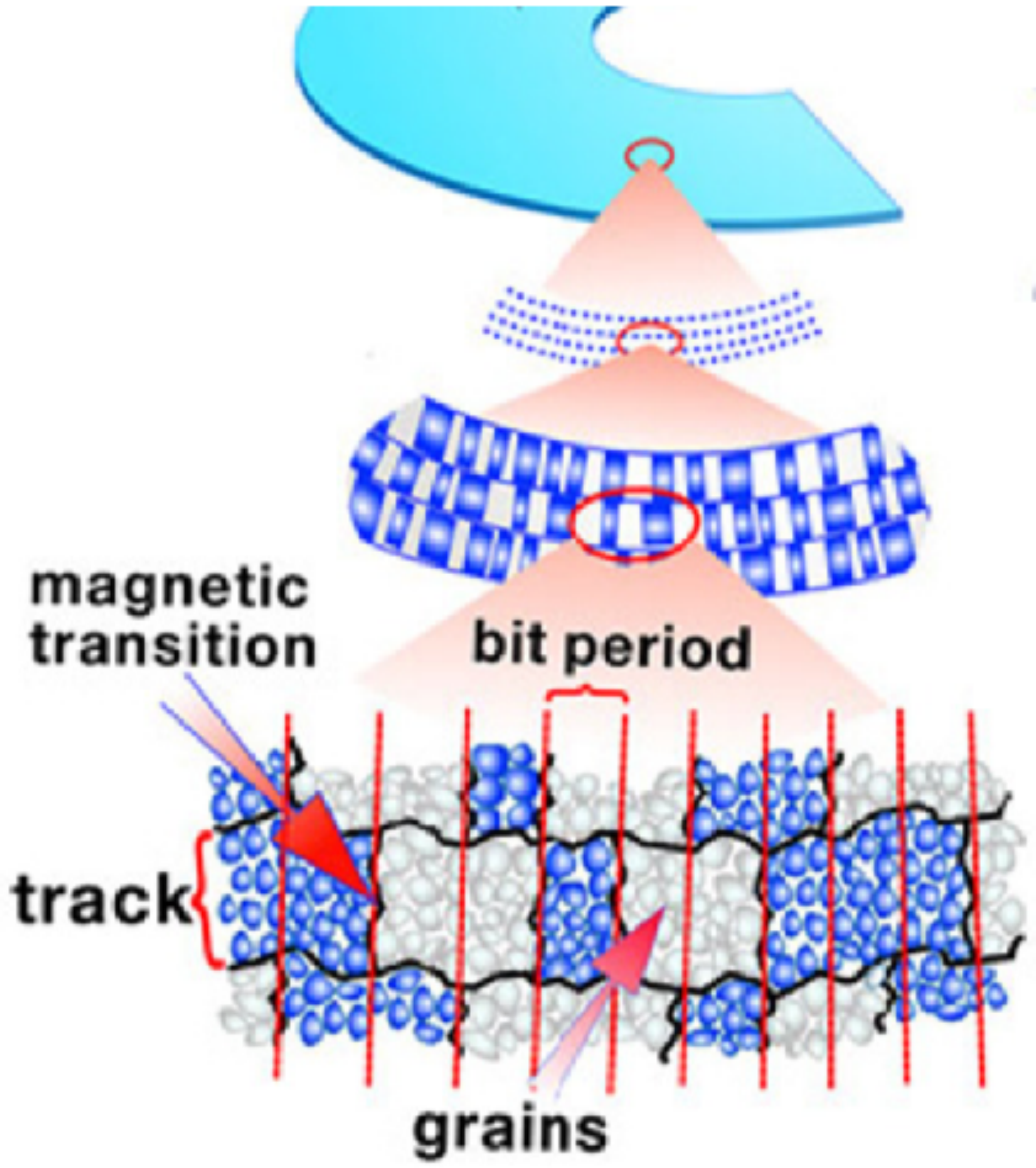
> *"… is a field of physics dealing with the prediction of magnetic behaviours at sub-micrometer length scales."*

Source: Wikipedia

WHY IS IT INTERESTING?

- Micromagnetics deals with complex systems with tuneable parameters.
- It is used to explain experiments as well as to design experiments.
- Real applications, including
  - magnetic data storage,
  - cancer therapy,
  - low-energy magnetic logic (spintronics).





MAGNETISATION FIELD

- Magnetisation field is the **main "unknown"**
- In continuum approximation, magnetisation is considered to be a **continuous vector field.**
- Magnetisation $\mathbf{M}(\mathbf{r}, t)$ is a function of both space $\mathbf{r}$ and time t.

$$\mathbf{M} = \mathbf{M}(\mathbf{r}, t)$$

MICROMAGNETIC ASSUMPTIONS

1. Magnetisation field **M**(**r**, *t*) is **differentiable** (continuous and slowly changing) with respect to both space **r** and time *t*.

2. The magnetisation field **norm is constant** (time invariant).

   - Constant norm |**M**| is represented by **saturation magnetisation** M_s.

$$M_\mathrm{s} = |\mathbf{M}| = \mathrm{const.}$$

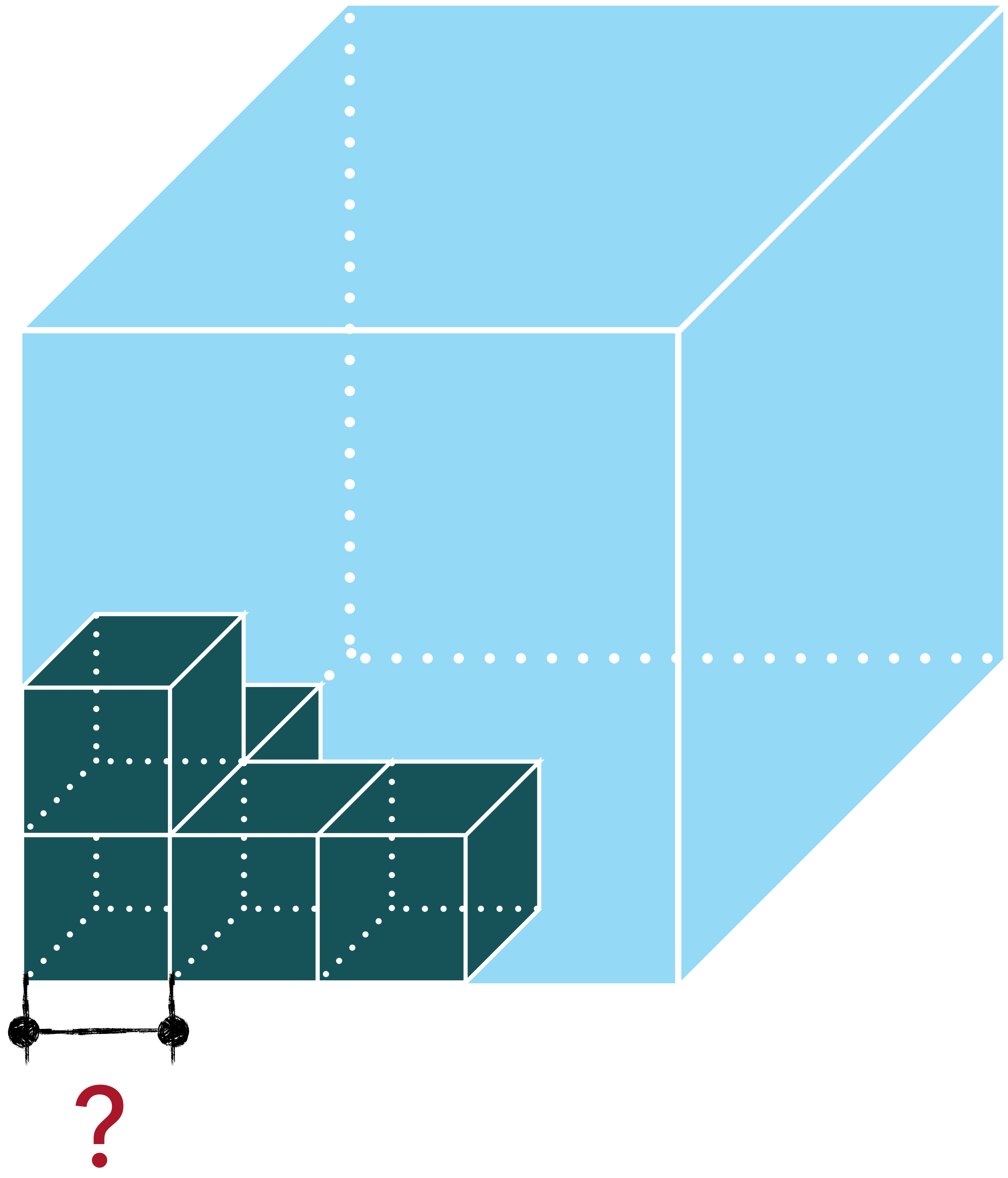
   - Very often, magnetisation is represented by a normalised magnetisation field **m**(**r**, t).

$$\mathbf{m}(\mathbf{r}, t) = \frac{\mathbf{M}(\mathbf{r}, t)}{M_\mathrm{s}}$$

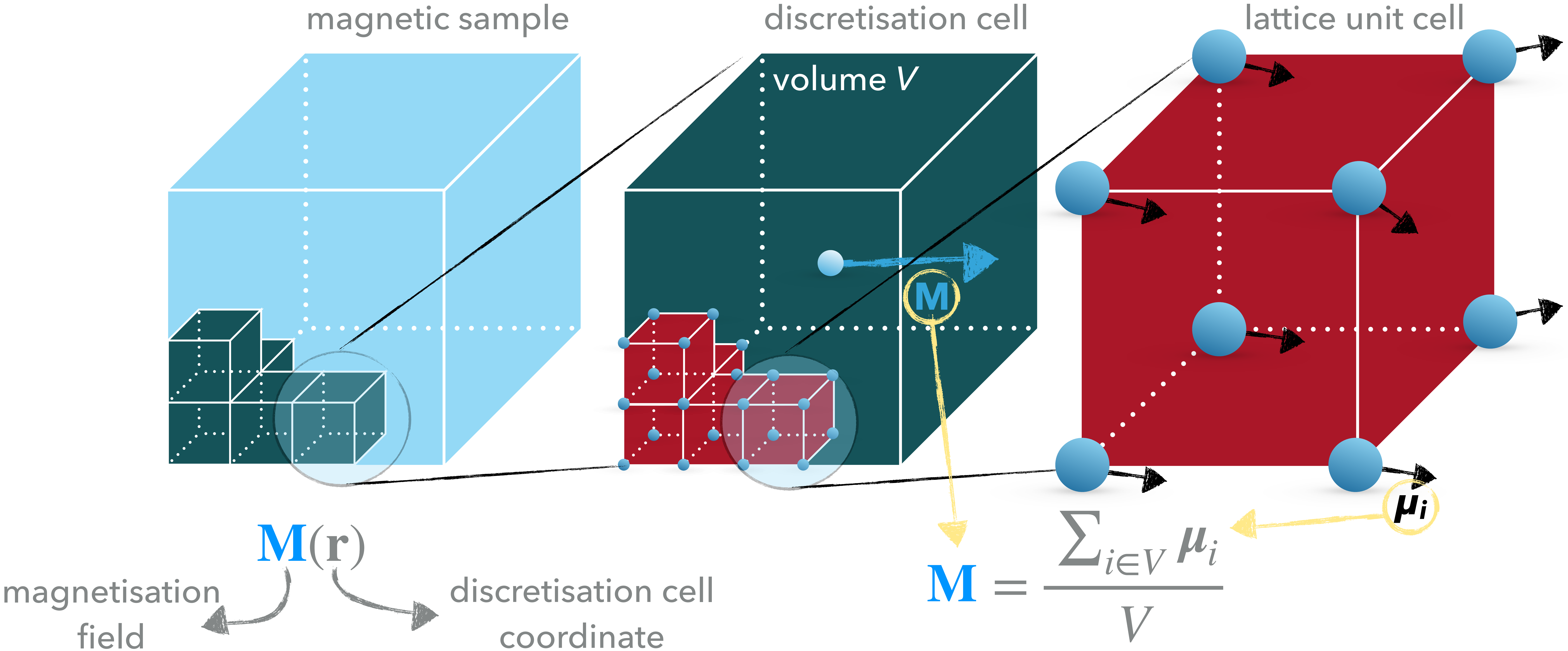
$$|\mathbf{m}| = m_x{}^2 + m_y{}^2 + m_z{}^2 = 1$$

DISCRETISATION

- In order to solve the magnetisation field numerically, we have to **divide it into smaller "chunks"**.
- There are two main ways how we can discretise the field:
  - **finite-differences**
  - finite-elements
- The discretisation must be:
  - **large enough** to ignore the crystal structure of the material (the continuum approximation).
  - **small enough** to spatially resolve different magnetisation configurations



FINITE-DIFFERENCE DISCRETISATION



magnetic sample

discretisation cell

lattice unit cell

volume V

$\mathbf{M}(\mathbf{r})$

magnetisation field

discretisation cell coordinate

$$\mathbf{M} = \frac{\sum_{i \in V} \boldsymbol{\mu}_i}{V}$$

ENERGY EQUATION (HAMILTONIAN)

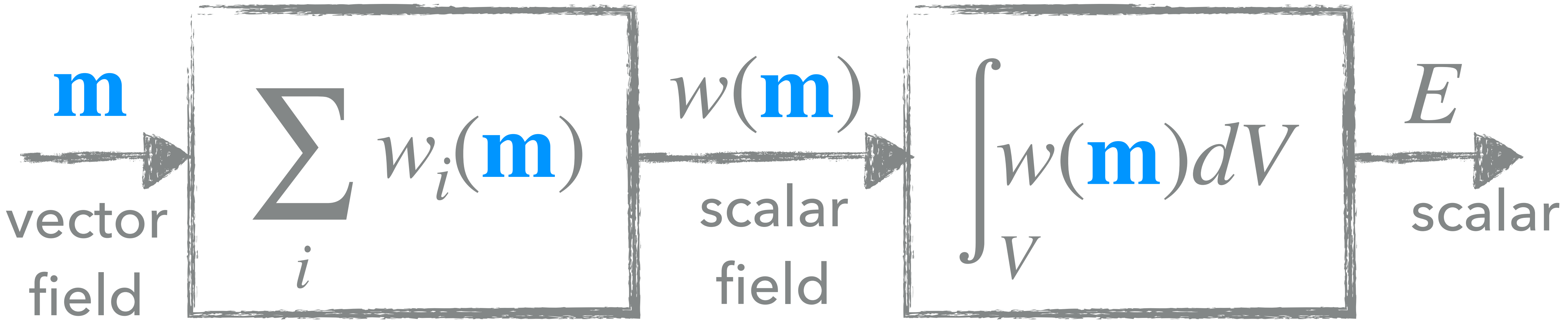
- … is a mapping of magnetisation field **m**=**m**(**r**, *t*) to energy density (scalar) field.

$$w(\mathbf{m}) = w_1(\mathbf{m}) + w_2(\mathbf{m}) + w_3(\mathbf{m}) + \ldots = \boxed{\sum_i w_i(\mathbf{m})}$$

user-defined

- By integrating *w*(**m**) over the entire sample volume *V*, the energy functional is

$$E[\mathbf{m}] = \int_V w(\mathbf{m}) dV$$



$\mathbf{m}$

vector field

$\sum_i w_i(\mathbf{m})$

$w(\mathbf{m})$

scalar field

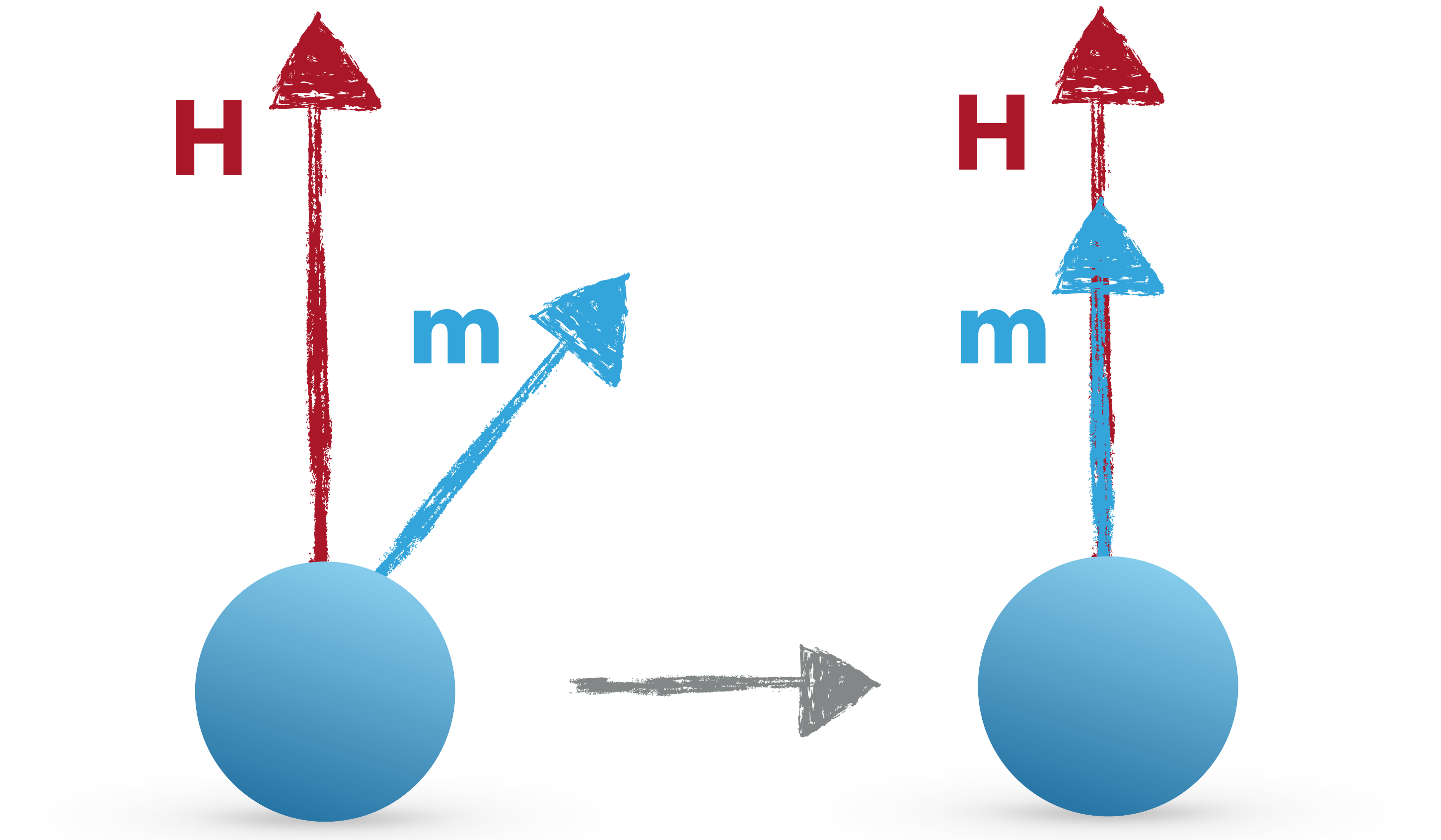
$\int_V w(\mathbf{m}) dV$

E

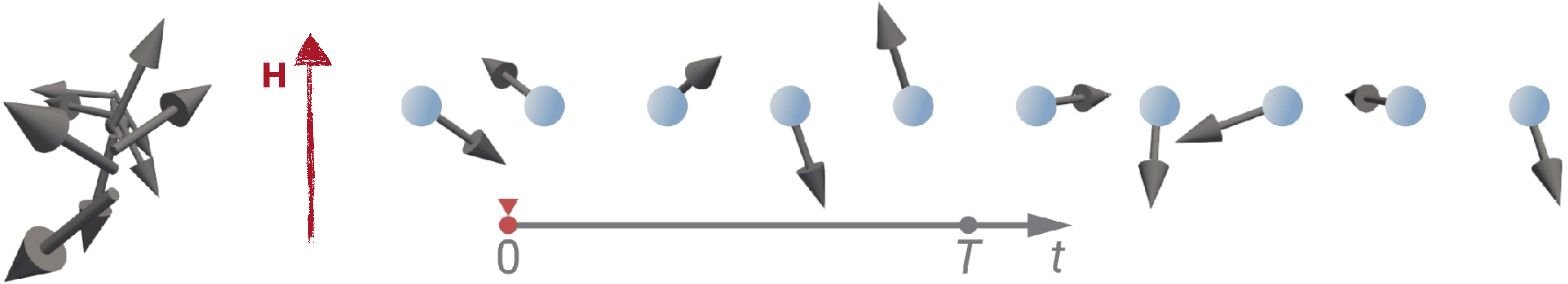
scalar

ZEEMAN

- Aligns **m** parallel to **H**.
- Parameter: **H** (A/m)



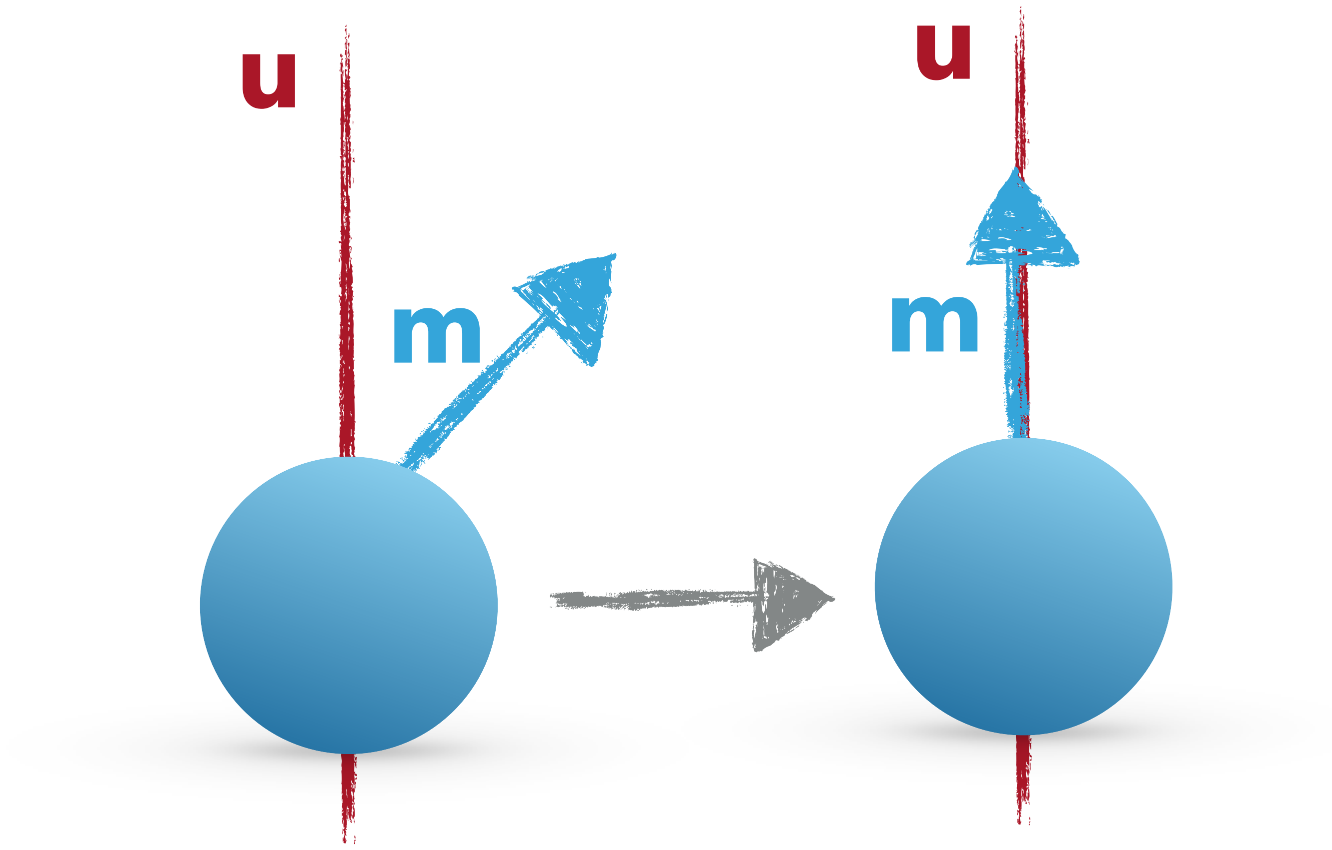
$$w_z = -\mathbf{M}\cdot\mathbf{B} \qquad (\mathbf{B} = \mu_0\mathbf{H},\ \mathbf{M} = M_s\mathbf{m})$$
$$= -\mu_0 M_s \mathbf{m}\cdot\mathbf{H}$$

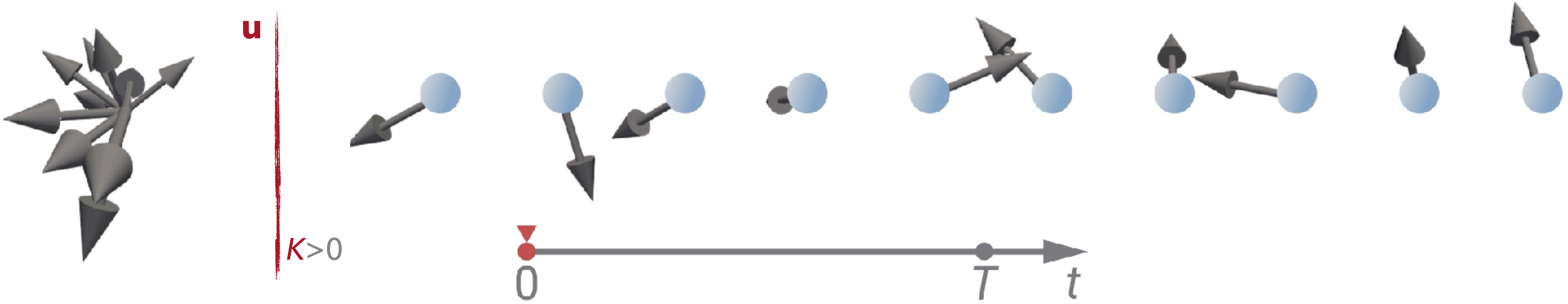


UNIAXIAL ANISOTROPY

- Aligns **m** parallel or antiparallel to **u**.
- Parameters: *K*>0 (J/m^3), **u**

$$w_{\mathrm{ua}} = -K(\mathbf{m}\cdot\mathbf{u})^2 \qquad (|\mathbf{m}| = 1,\ |\mathbf{u}| = 1)$$





(FERROMAGNETIC) EXCHANGE

- Aligns all magnetic moments (in **m**) parallel to each other.
- Parameter: *A* (J/m)

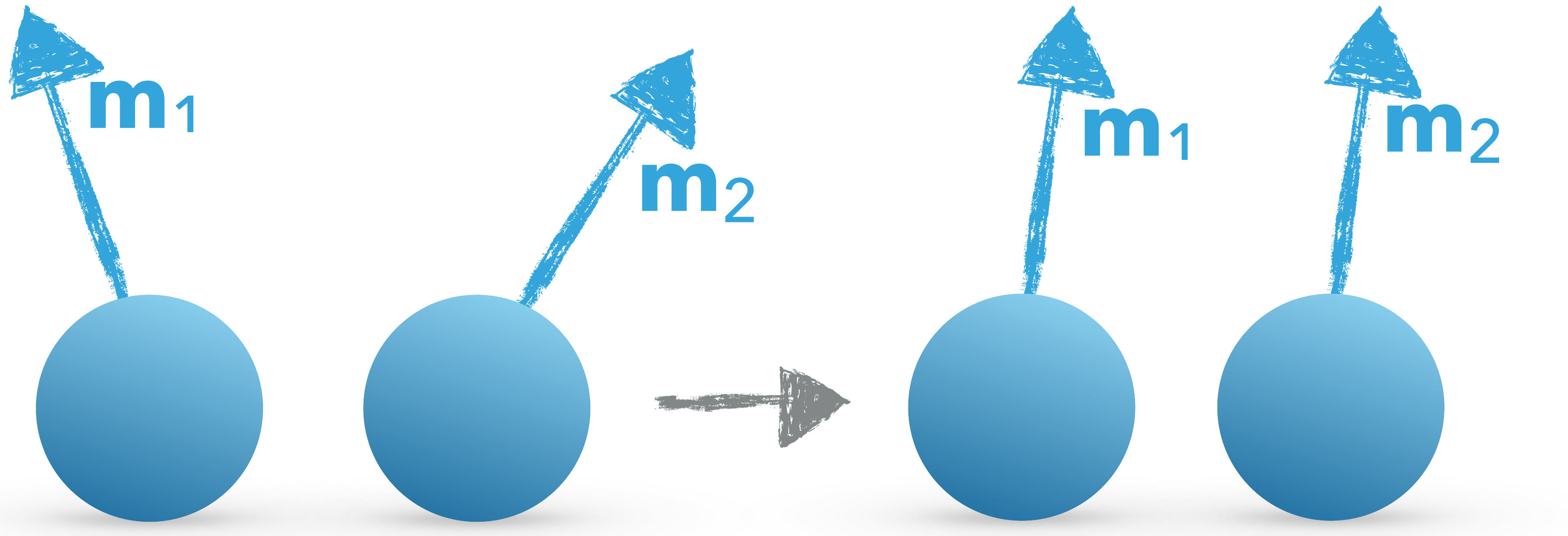
vector Laplacian

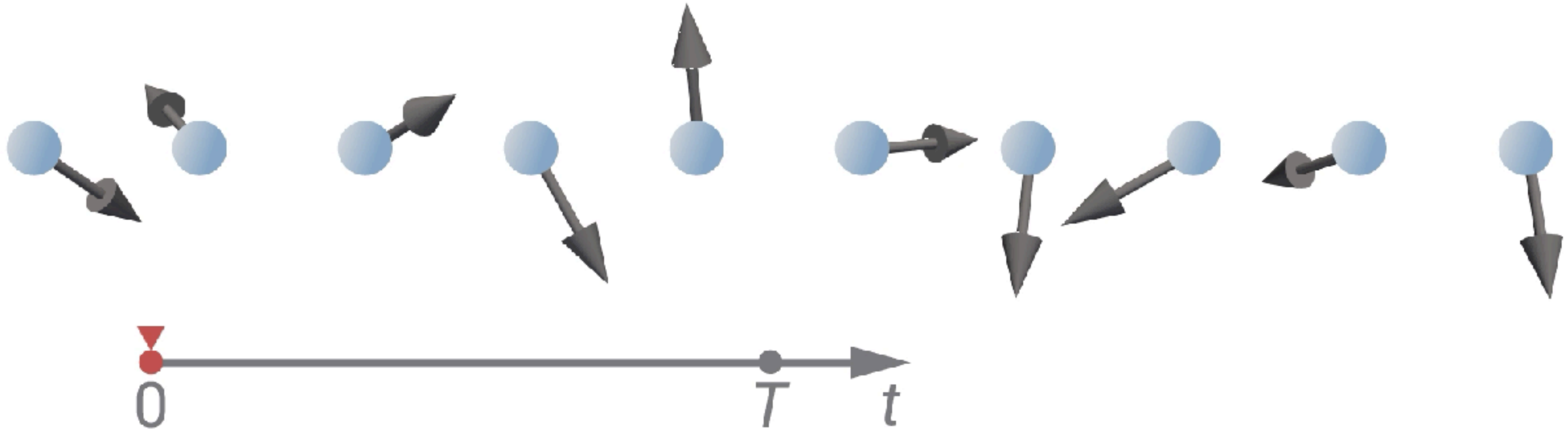
$$w_{\mathrm{ex}} = -A\,\mathbf{m}\cdot\nabla^2\mathbf{m} \quad \left(\nabla^2\mathbf{m} = \nabla^2 m_x\hat{\mathbf{x}} + \nabla^2 m_y\hat{\mathbf{y}} + \nabla^2 m_z\hat{\mathbf{z}}\right)$$

$$= A[(\nabla m_x)^2 + (\nabla m_y)^2 + (\nabla m_z)^2]$$

$$(=) \boxed{A(\nabla\mathbf{m})^2}$$

just a convention

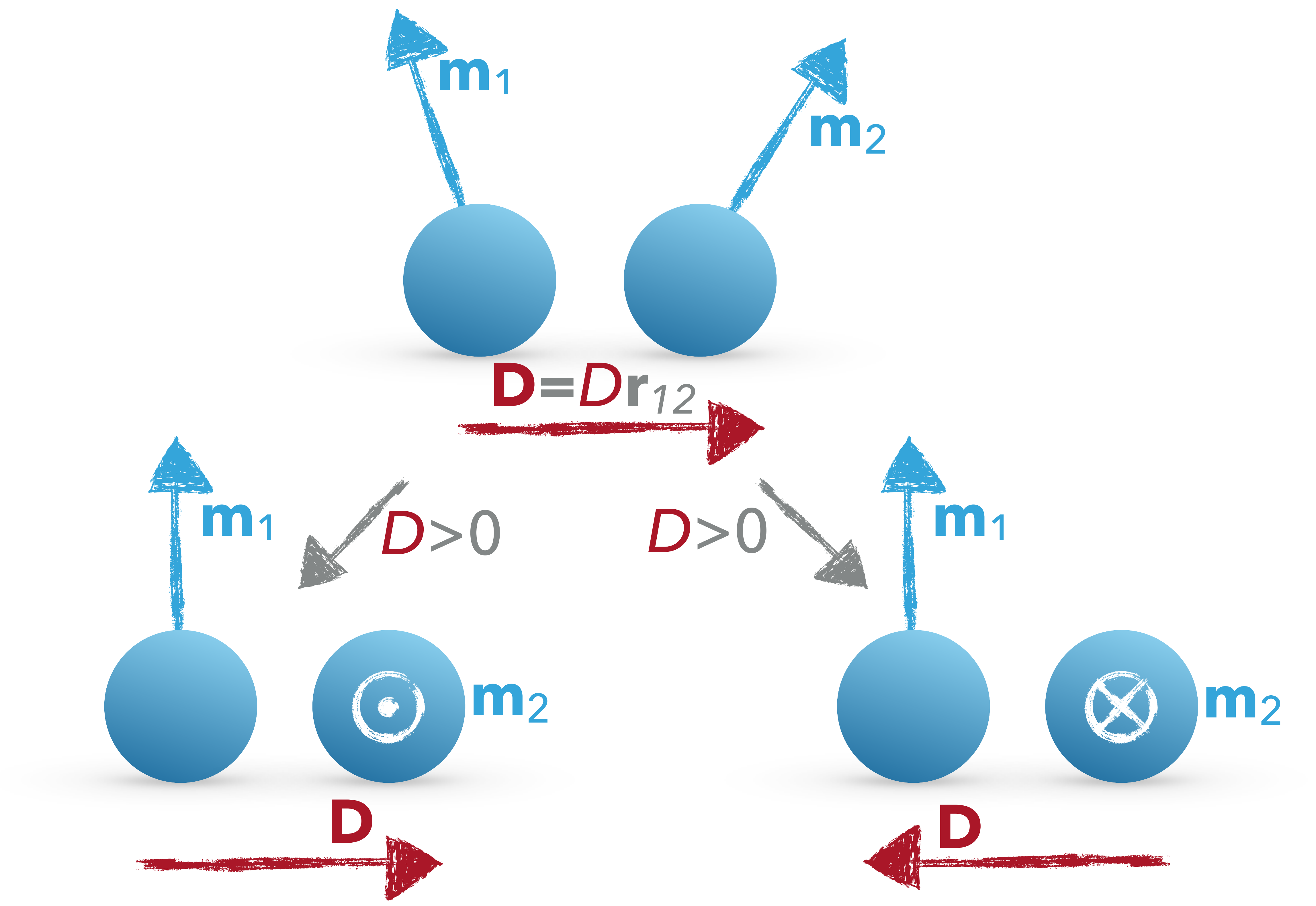


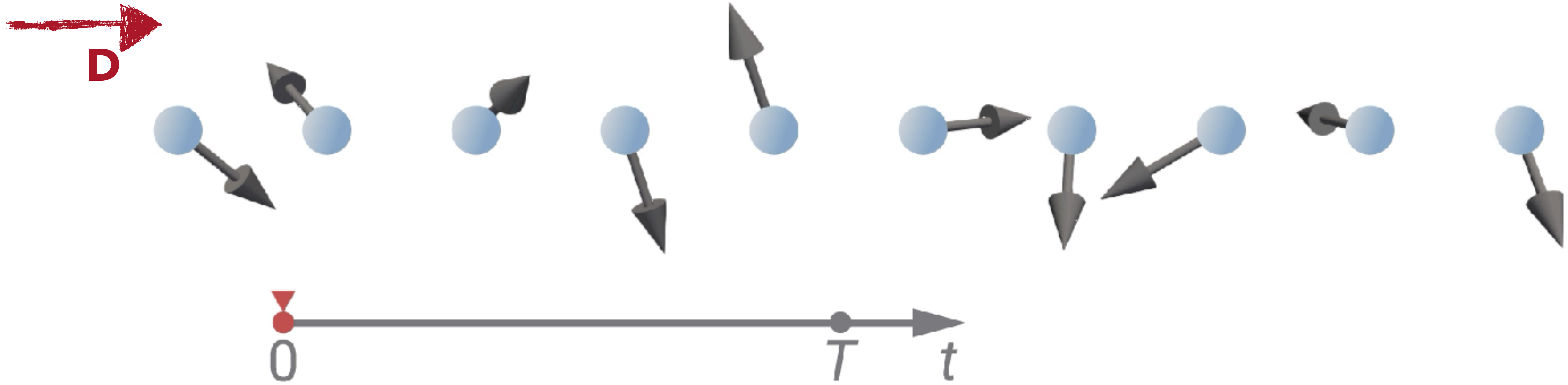


DZYALOSHINSKII-MORIYA

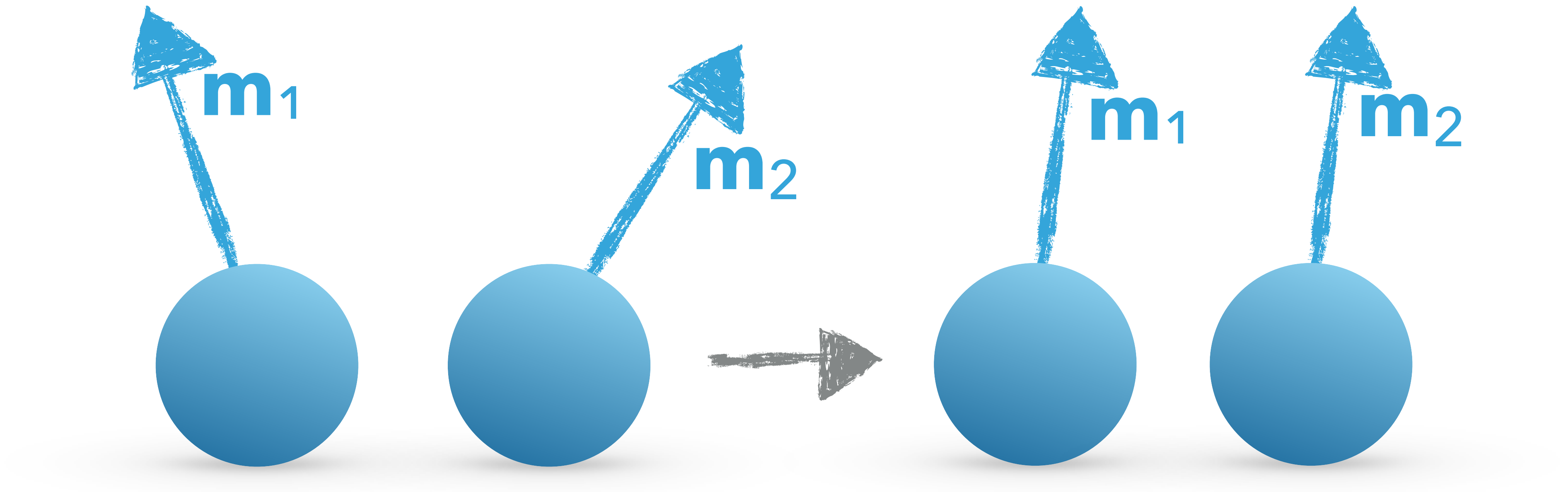
- Aligns neighbouring magnetic moments (in **m**) perpendicular to each other.
- Parameter: *D* (J/m²)

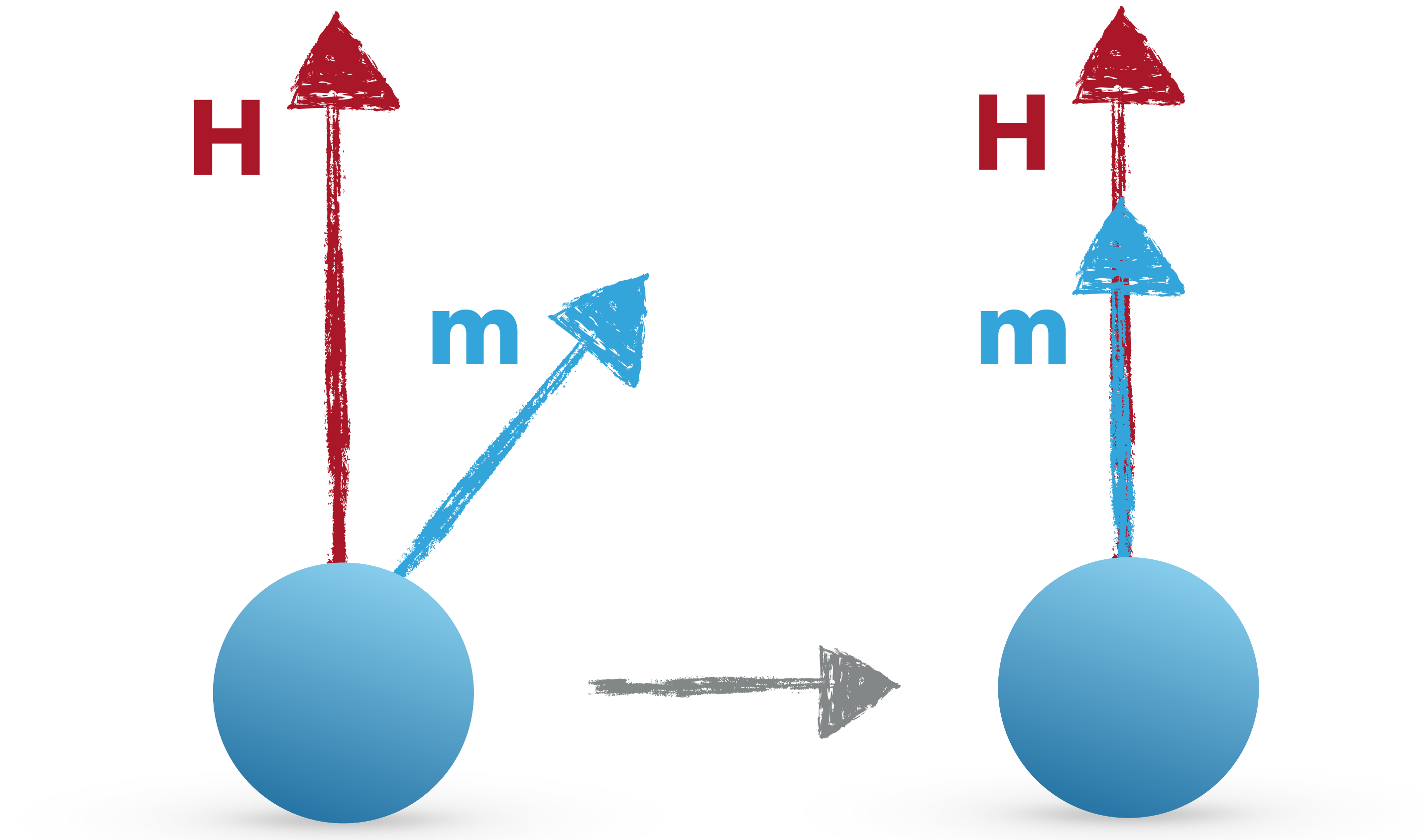
$$w_{\mathrm{dmi}} = (\pm)D\mathbf{m}\cdot(\nabla\times\mathbf{m}) \qquad \left(\mathbf{D} = D\mathbf{r}_{ij}\right)$$





EXCHANGE AND ZEEMAN

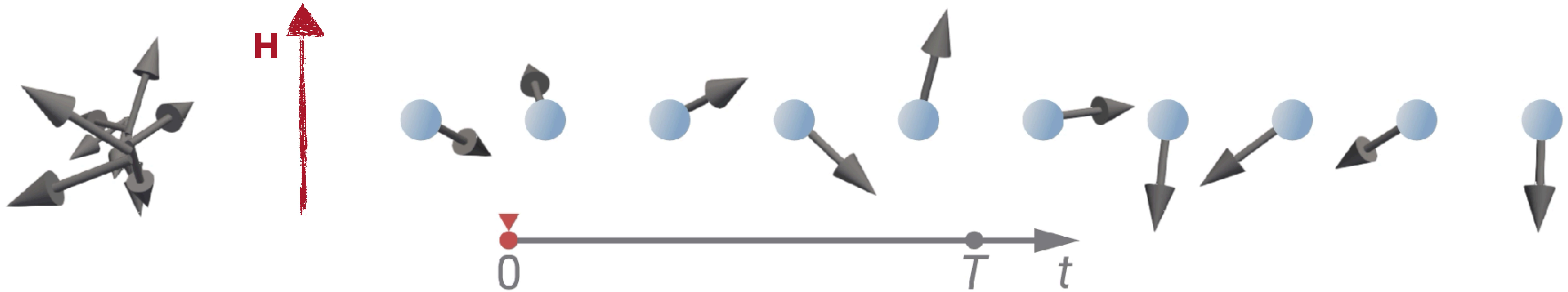




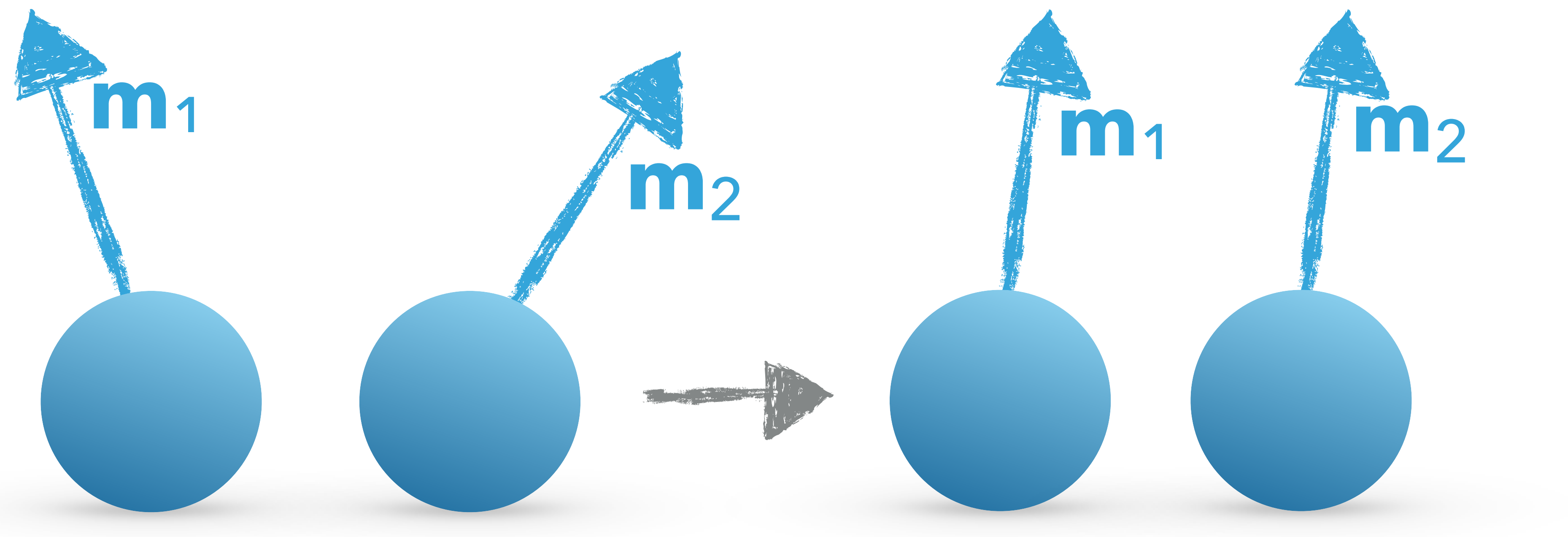
All magnetic moments parallel to each other? ✓

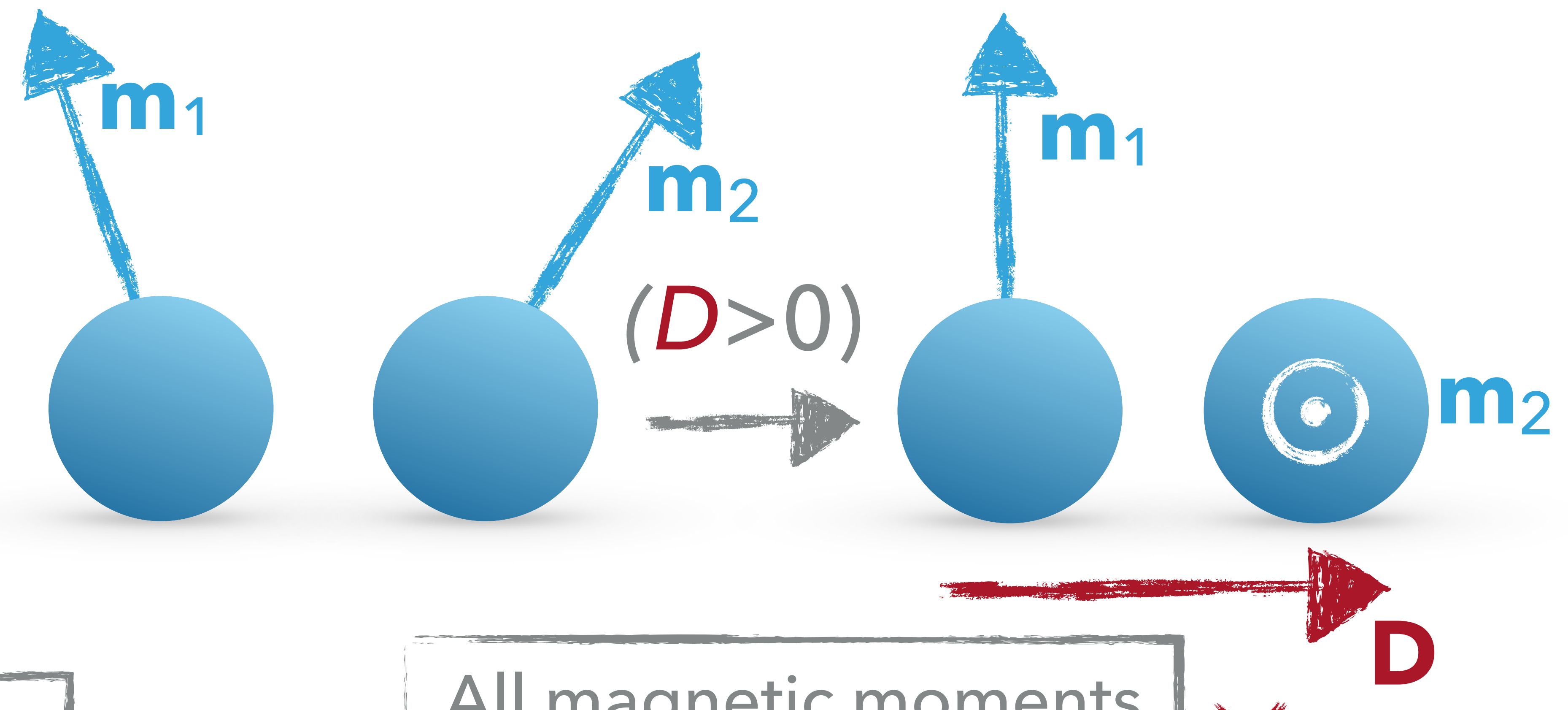
All magnetic moments parallel to **H**? ✓

No compromise is needed.



EXCHANGE AND DMI

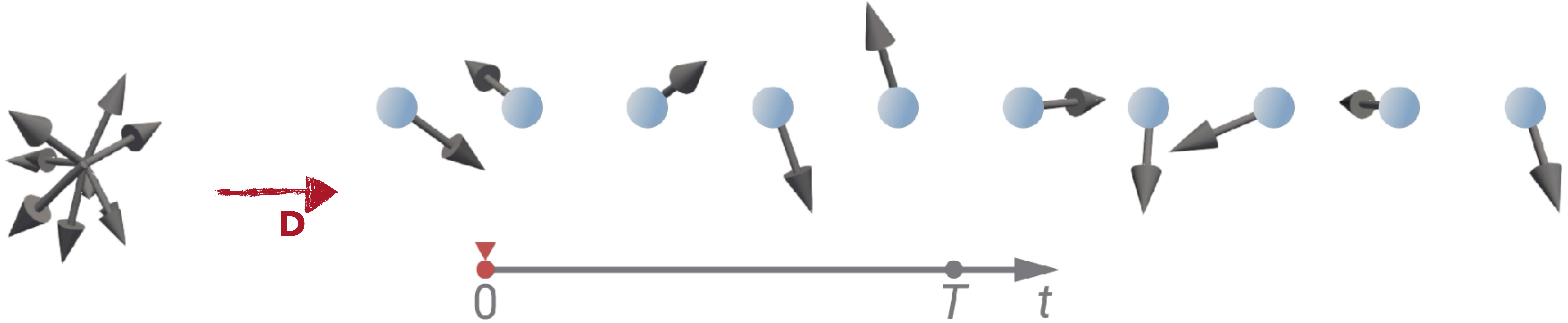




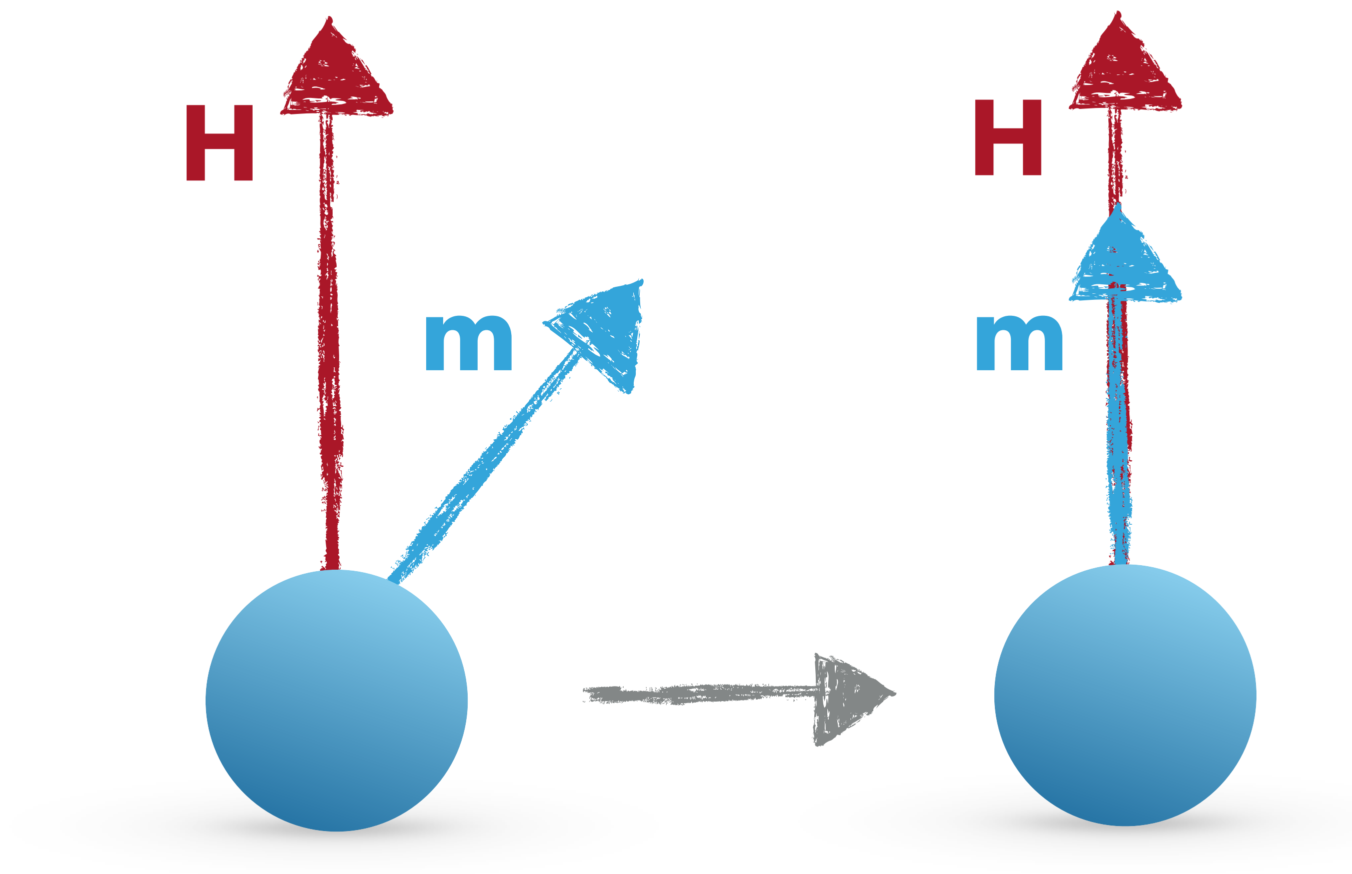
All magnetic moments parallel to each other?

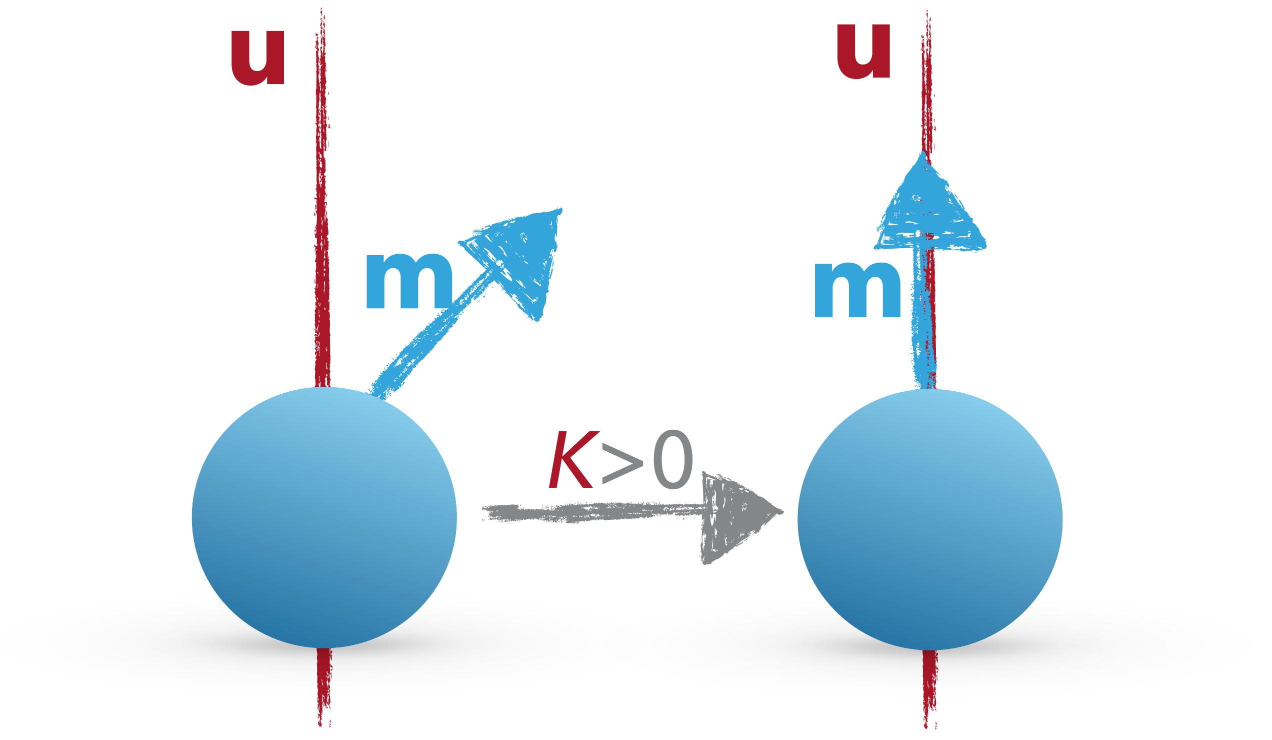
Energies have to compete and reach a compromise.

All magnetic moments perpendicular to each other?

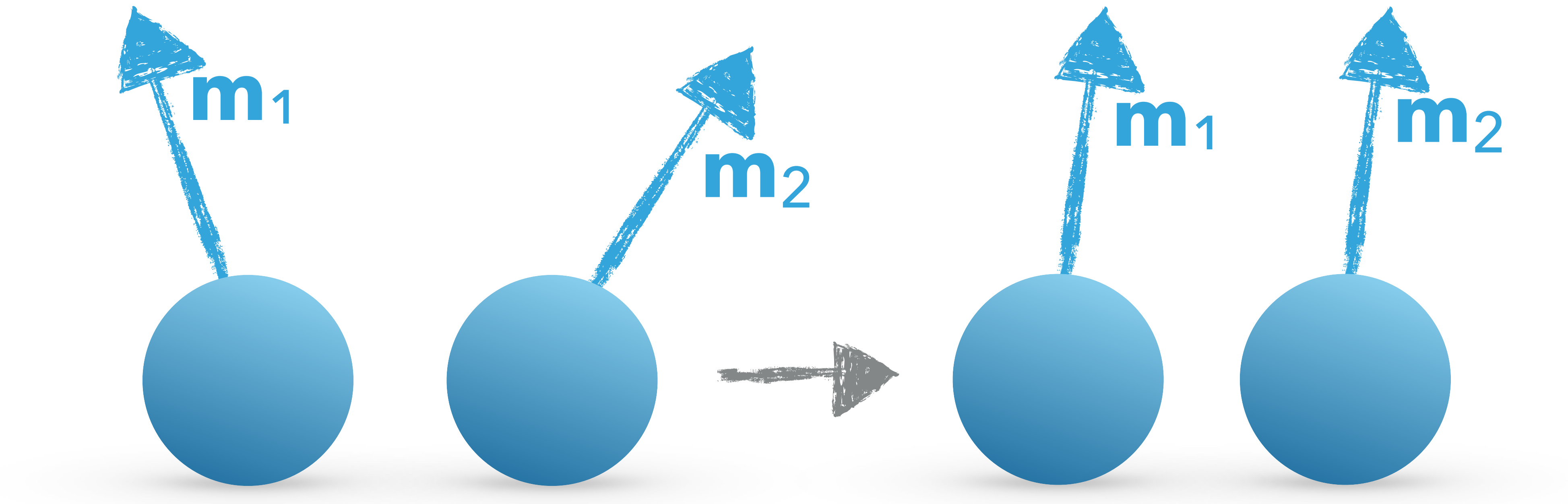


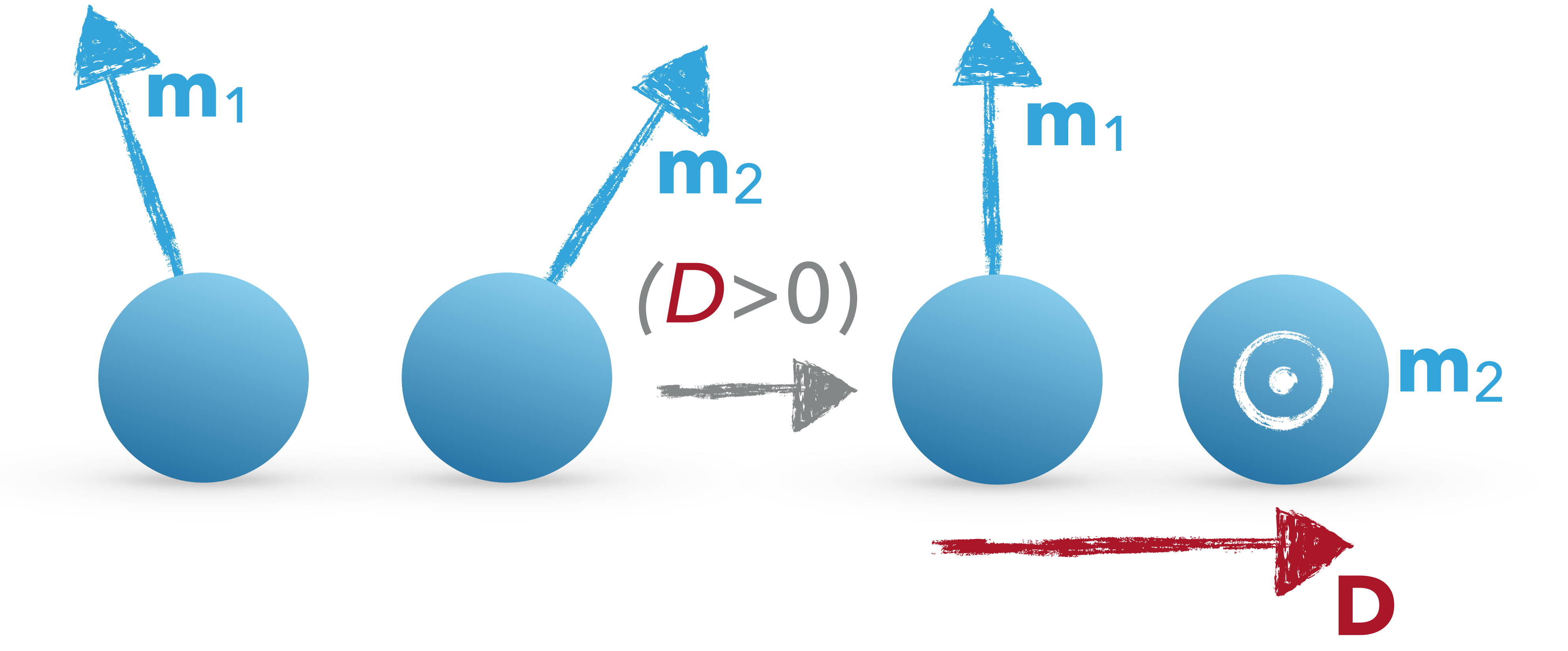
MORE COMPLICATED CASE IN A 2D SAMPLE (1/2)



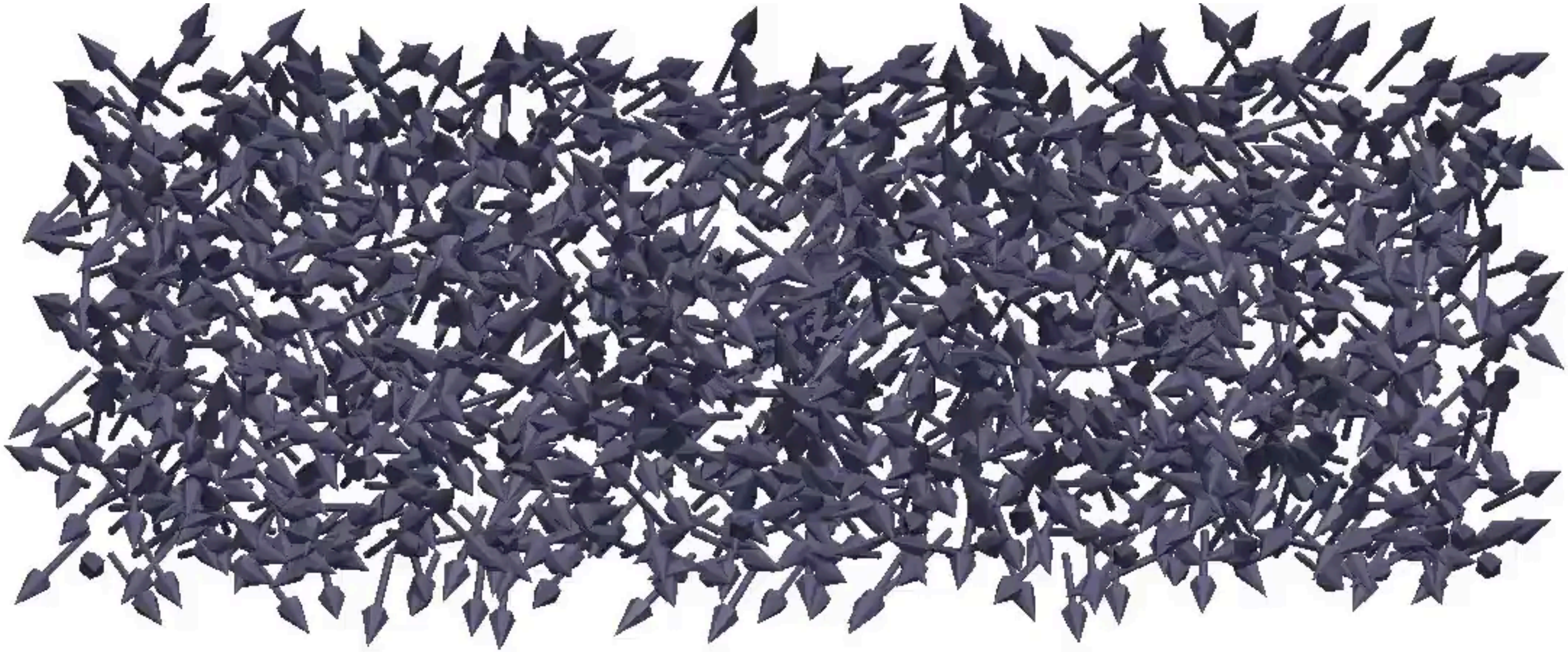


Energies have to compete and reach a compromise.





MORE COMPLICATED CASE IN A 2D SAMPLE (2/2)



DYNAMICS EQUATION

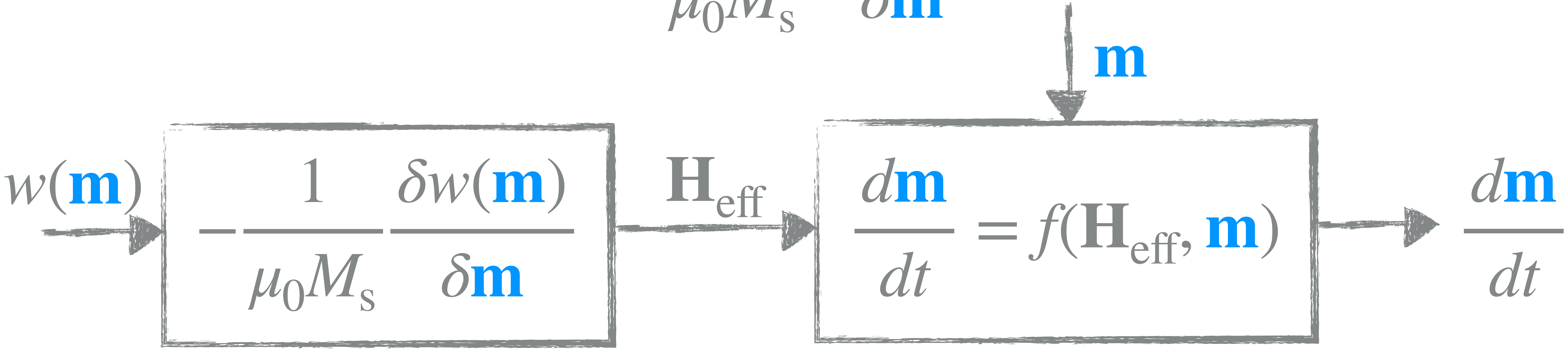
- …tells us how magnetisation **m** wants to change in order to minimise its energy.

$$\frac{d\mathbf{m}}{dt} = f_1(\mathbf{m}, \mathbf{H}_\text{eff}, \ldots) + f_1(\mathbf{m}, \mathbf{H}_\text{eff}, \ldots) + \ldots = \sum_i f_i(\mathbf{m}, \mathbf{H}_\text{eff}, \ldots)$$

user-defined

- Effective field is computed as the first variational derivative of energy density:

$$\mathbf{H}_\text{eff} = -\frac{1}{\mu_0 M_\text{s}} \frac{\delta w(\mathbf{m})}{\delta \mathbf{m}}$$



$\mathbf{m}$

$w(\mathbf{m})$

$$-\frac{1}{\mu_0 M_\text{s}} \frac{\delta w(\mathbf{m})}{\delta \mathbf{m}}$$

$\mathbf{H}_\text{eff}$

$$\frac{d\mathbf{m}}{dt} = f(\mathbf{H}_\text{eff}, \mathbf{m})$$

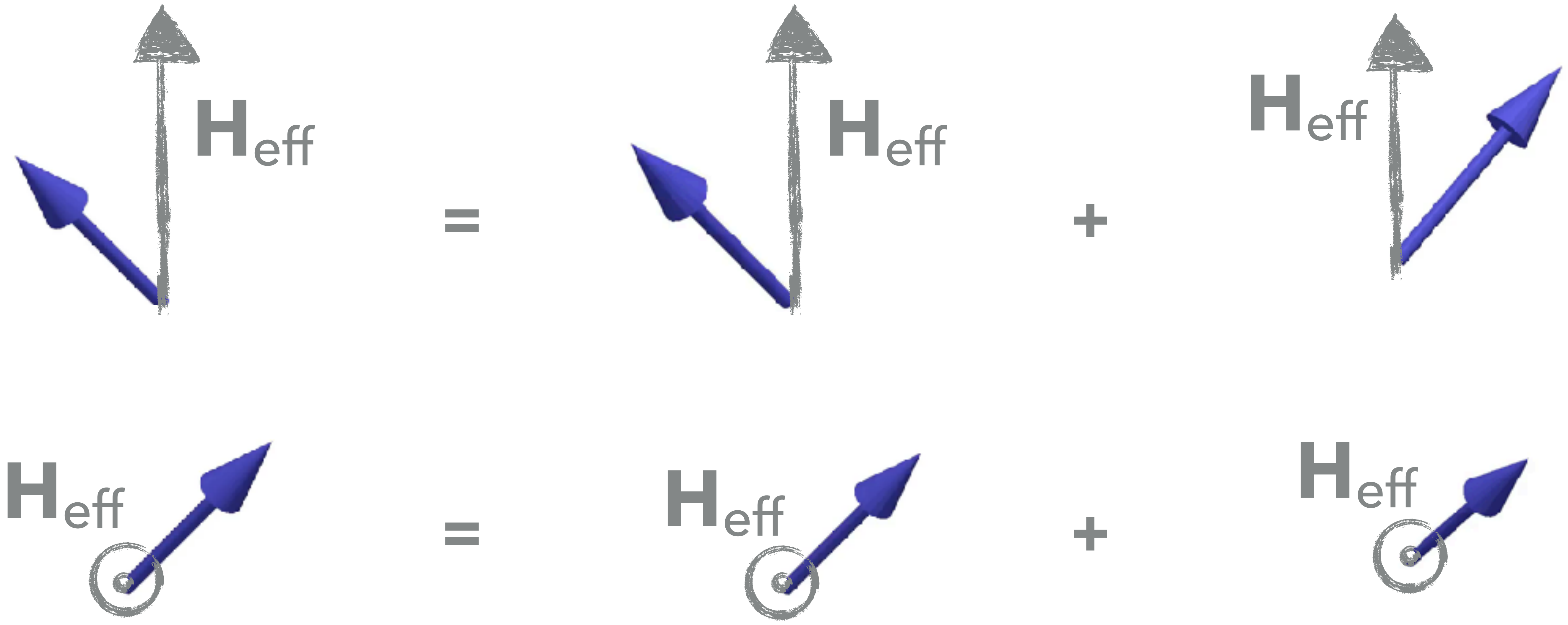
$$\frac{d\mathbf{m}}{dt}$$

LANDAU-LIFSCHITZ-GILBERT EQUATION

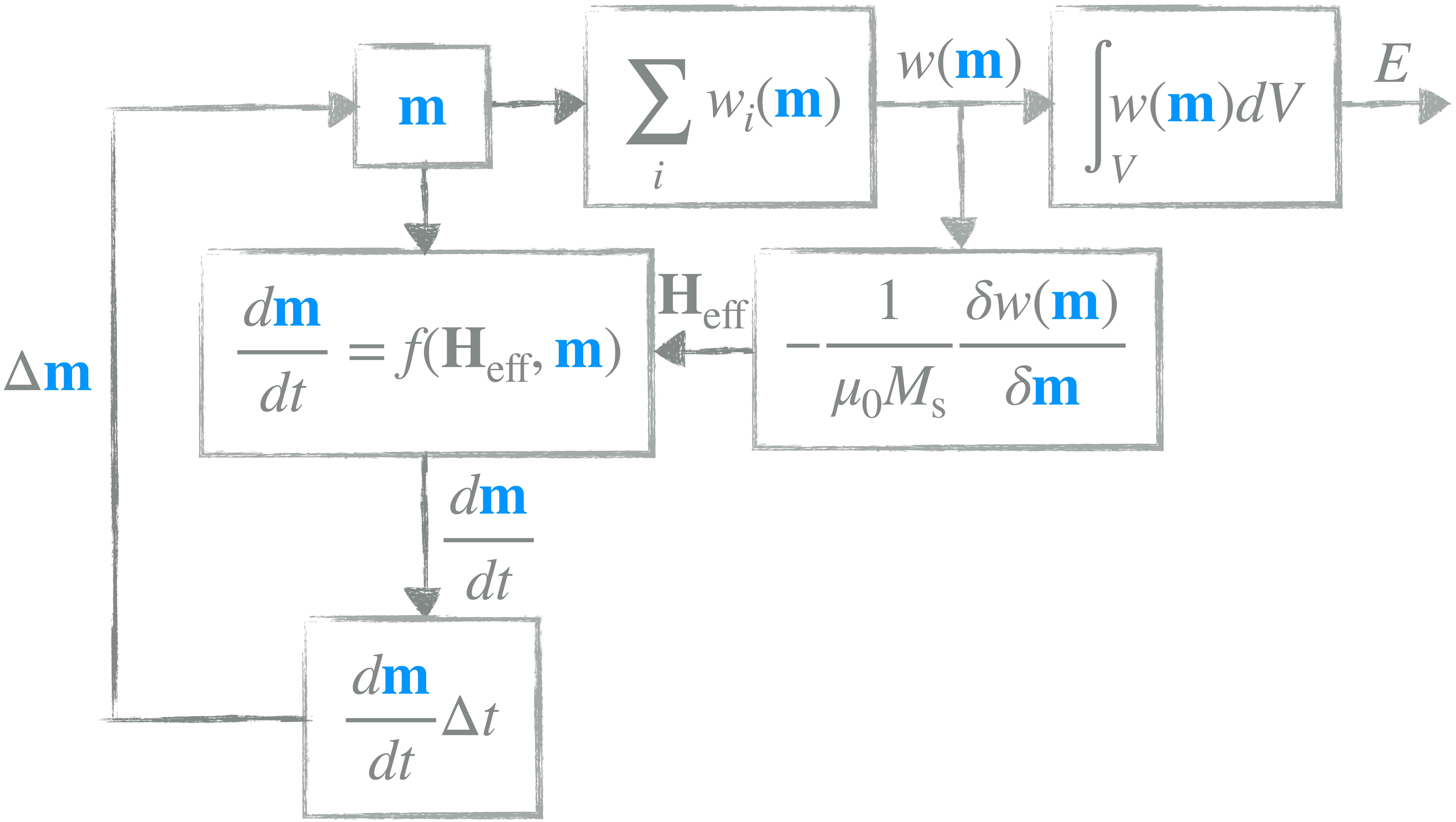
$$\frac{d\mathbf{m}}{dt} = -\gamma_0 \mathbf{m} \times \mathbf{H}_{\text{eff}} + \alpha \left(\mathbf{m} \times \frac{d\mathbf{m}}{dt} \right)$$

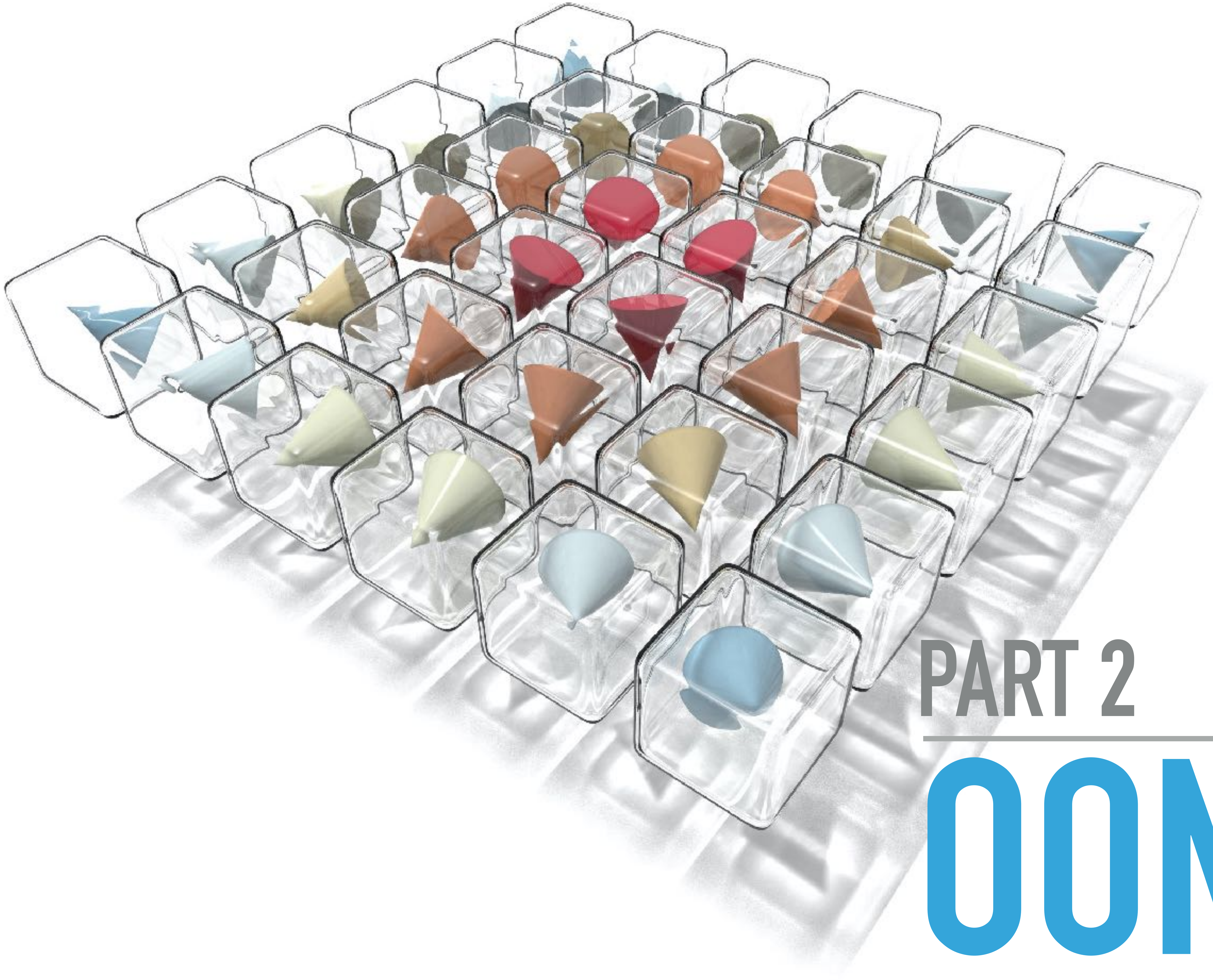
$$\downarrow$$

$$\frac{d\mathbf{m}}{dt} = -\frac{\gamma_0}{1+\alpha^2} \mathbf{m} \times \mathbf{H}_{\text{eff}} - \frac{\gamma_0 \alpha}{1+\alpha^2} \mathbf{m} \times (\mathbf{m} \times \mathbf{H}_{\text{eff}})$$



(OVERSIMPLIFIED) MICROMAGNETIC SIMULATOR



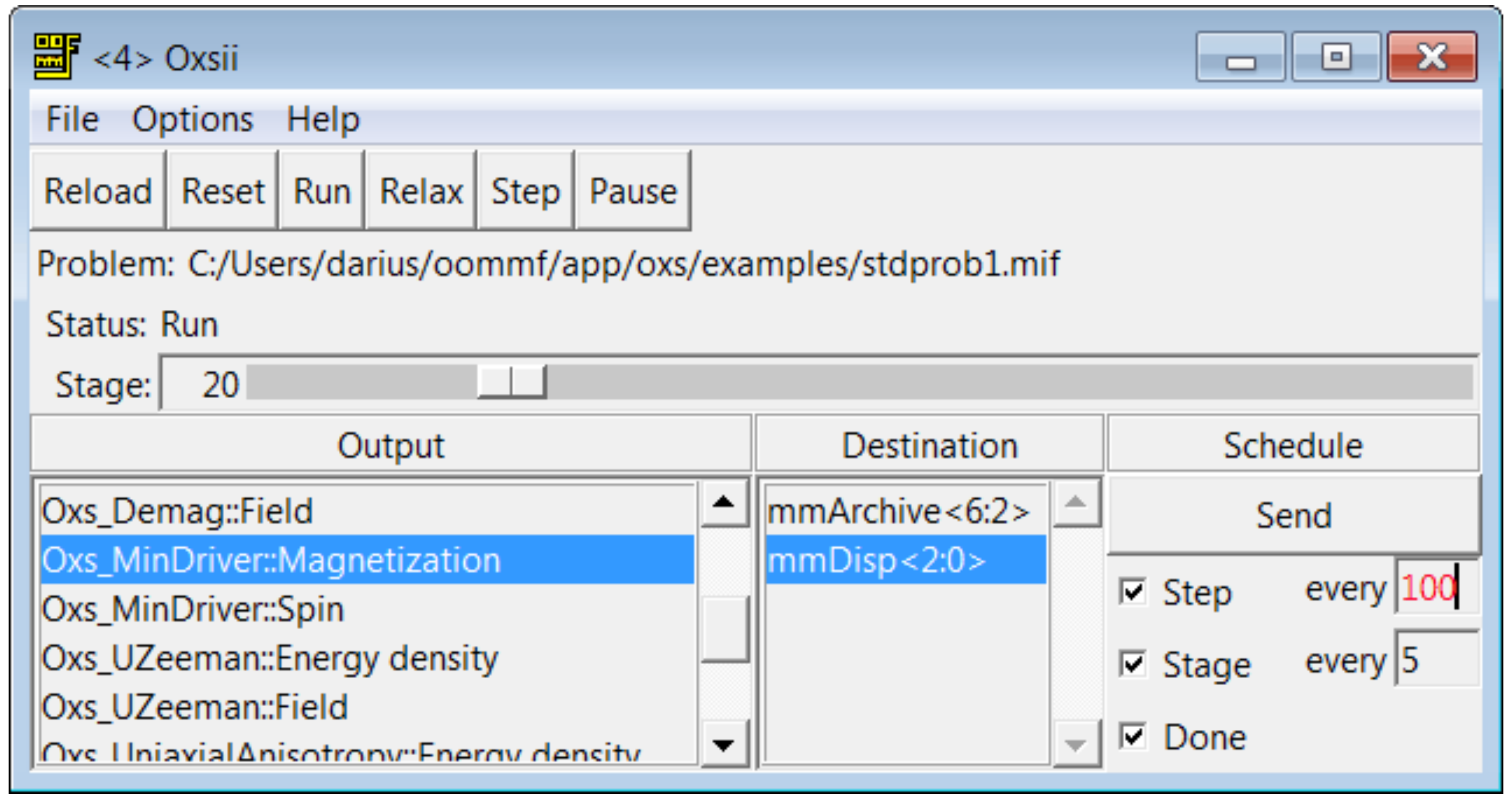


PART 2

OOMMF

OOMMF

- Probably **the most widely used** micromagnetic simulation tool
- Developed at **NIST**, US, **since ~1998** by Michael Donahue & Don Porter
- **Cited over 2200** times in scientific publications
- Written in **C++** & **Tcl**
- **Finite-difference** code
- Very often **used for comparisons** between codes
- https://math.nist.gov/oommf/



Problem: C:/Users/darius/oommf/app/oxs/examples/stdprob1.mif

Status: Run

Stage: 20

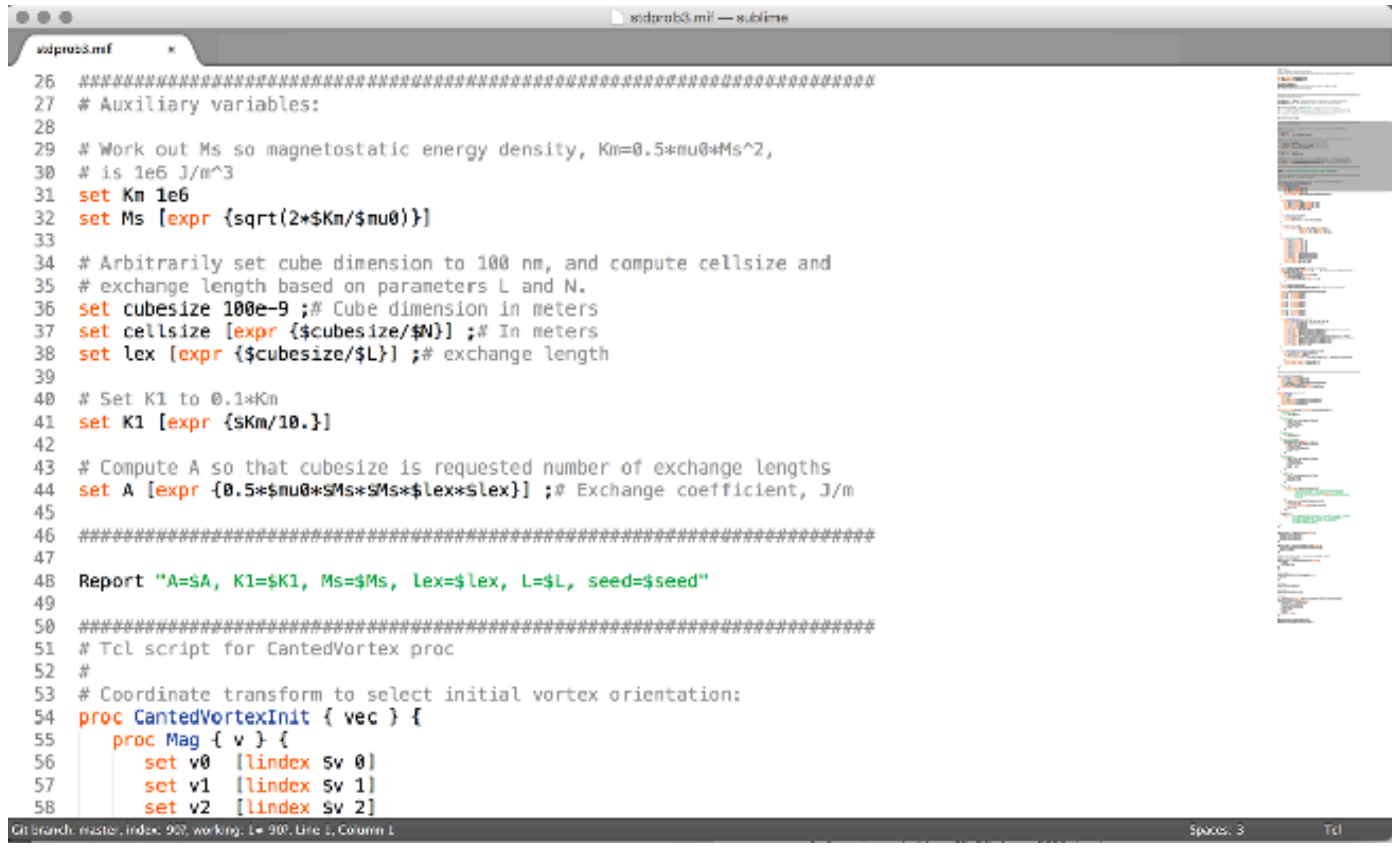
- Oxs_Demag::Field
- Oxs_MinDriver::Magnetization
- Oxs_MinDriver::Spin
- Oxs_UZeeman::Energy density
- Oxs_UZeeman::Field
- Oxs_UniaxialAnisotropy::Energy density

- mmArchive<6:2>
- mmDisp<2:0>

☑ Step every 100

☑ Stage every 5

☑ Done

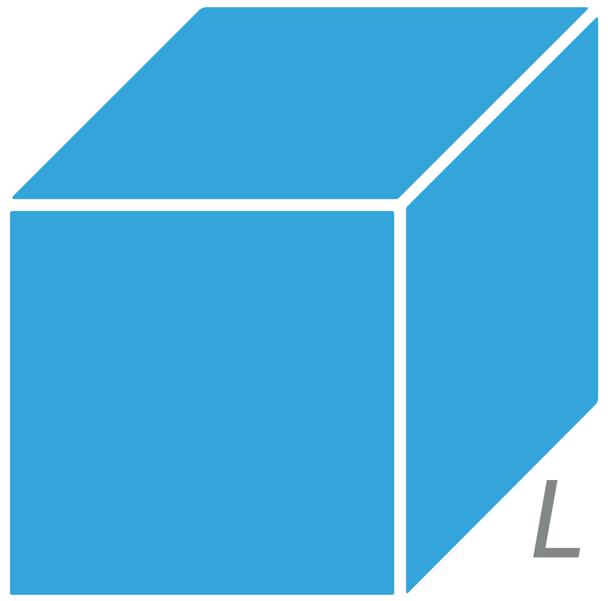


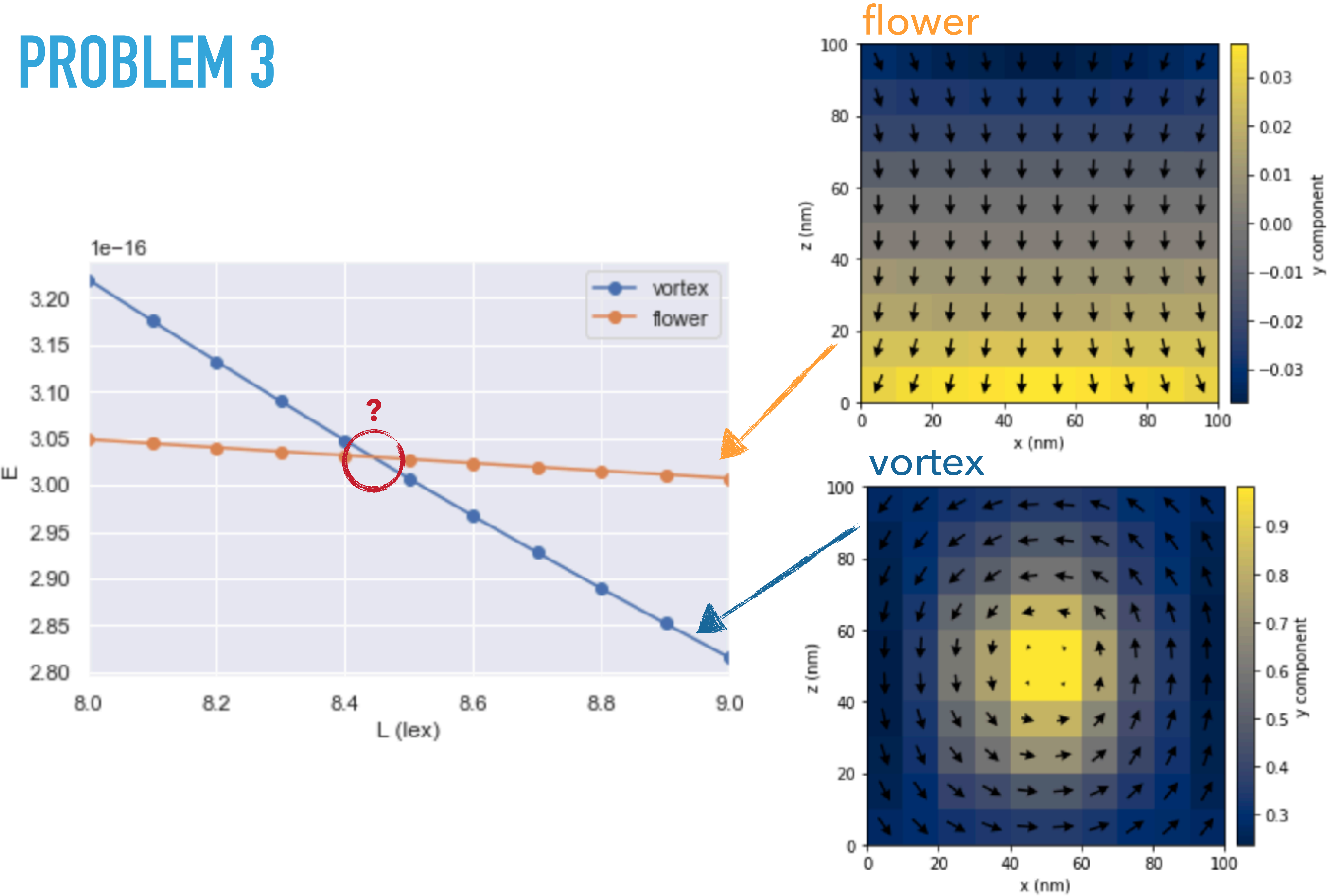
```
##############################################################################
# Auxiliary variables:

# Work out Ms so magnetostatic energy density, Km=0.5*mu0*Ms^2,
# is 1e6 J/m^3
set Km 1e6
set Ms [expr {sqrt(2*$Km/$mu0)}]

# Arbitrarily set cube dimension to 100 nm, and compute cellsize and
# exchange length based on parameters L and N.
set cubesize 100e-9 ;# Cube dimension in meters
set cellsize [expr {$cubesize/$N}] ;# In meters
set lex [expr {$cubesize/$L}] ;# exchange length

# Set K1 to 0.1*Km
set K1 [expr {$Km/10.}]

# Compute A so that cubesize is requested number of exchange lengths
set A [expr {0.5*$mu0*$Ms*$Ms*$lex*$lex}] ;# Exchange coefficient, J/m

##############################################################################

Report "A=$A, K1=$K1, Ms=$Ms, lex=$lex, L=$L, seed=$seed"

##############################################################################
# Tcl script for CantedVortex proc
#
# Coordinate transform to select initial vortex orientation:
proc CantedVortexInit { vec } {
    proc Mag { v } {
        set v0  [lindex $v 0]
        set v1  [lindex $v 1]
        set v2  [lindex $v 2]
```

PART 3

COMPUTATIONAL WORKFLOW

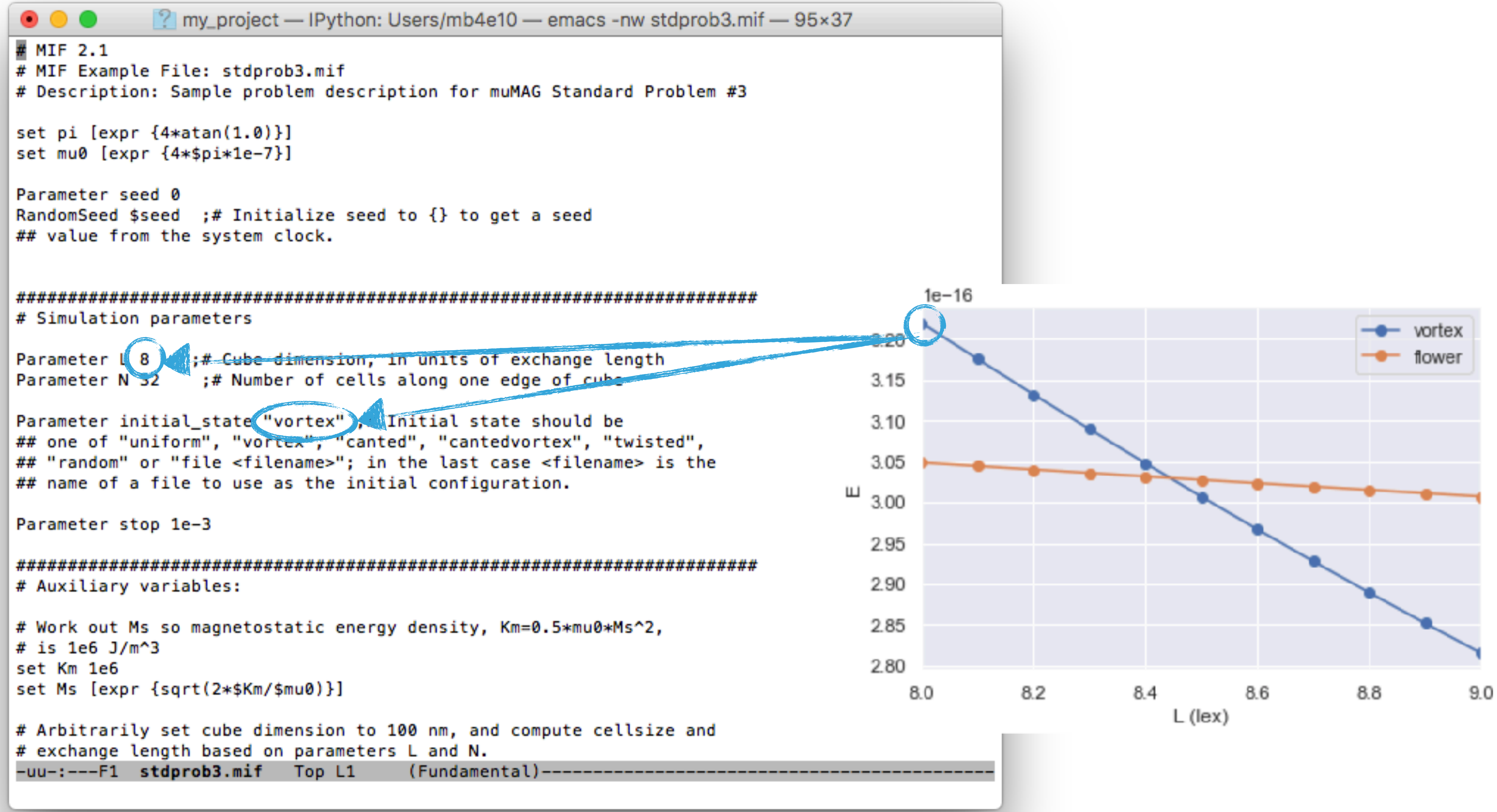
CASE STUDY: STANDARD PROBLEM 3





Research question: For what cube edge length *L*, vortex and flower states have the same energy?

STEP 1: WRITE CONFIGURATION FILE



```
# MIF 2.1
# MIF Example File: stdprob3.mif
# Description: Sample problem description for muMAG Standard Problem #3

set pi [expr {4*atan(1.0)}]
set mu0 [expr {4*$pi*1e-7}]

Parameter seed 0
RandomSeed $seed  ;# Initialize seed to {} to get a seed
## value from the system clock.


########################################################################
# Simulation parameters

Parameter L 8   ;# Cube dimension, in units of exchange length
Parameter N 32   ;# Number of cells along one edge of cube

Parameter initial_state "vortex" ;# Initial state should be
## one of "uniform", "vortex", "canted", "cantedvortex", "twisted",
## "random" or "file <filename>"; in the last case <filename> is the
## name of a file to use as the initial configuration.

Parameter stop 1e-3

########################################################################
# Auxiliary variables:

# Work out Ms so magnetostatic energy density, Km=0.5*mu0*Ms^2,
# is 1e6 J/m^3
set Km 1e6
set Ms [expr {sqrt(2*$Km/$mu0)}]

# Arbitrarily set cube dimension to 100 nm, and compute cellsize and
# exchange length based on parameters L and N.
-uu-:---F1  stdprob3.mif   Top L1     (Fundamental)-----------------------------------------
```

STEP 2: RUN SIMULATION

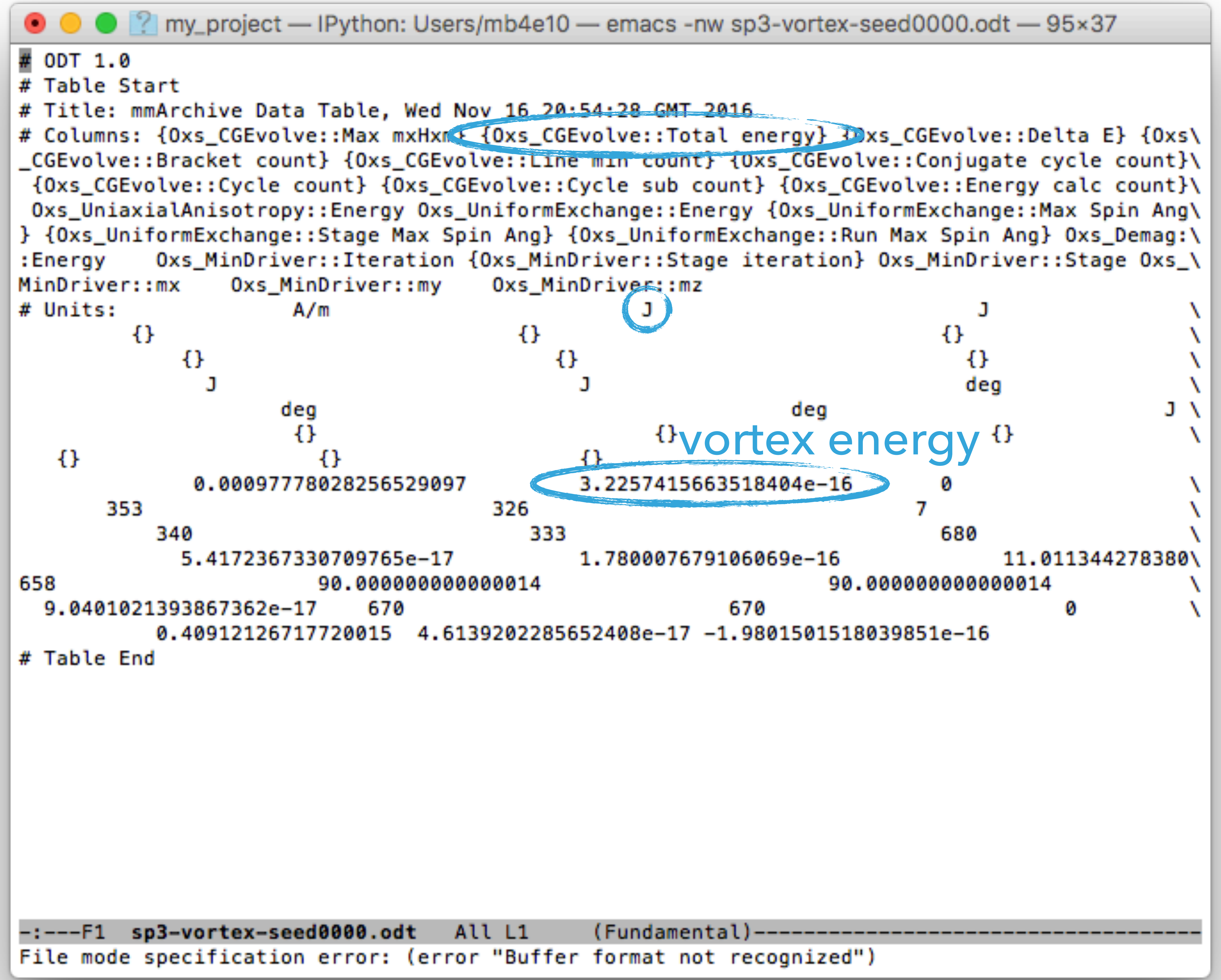
```
Marijans-MBP:my_project mb4e10$ ls
stdprob3.mif
Marijans-MBP:my_project mb4e10$ tclsh $OOMMFTCL boxsi +fg stdprob3.mif -exitondone 1
Start: "/Users/mb4e10/my_project/stdprob3.mif"
Options: -exitondone 1 -threads 2
Boxsi version 1.2.1.0
Running on: marijans-macbook-pro.local
OS/machine: Darwin/x86_64
User: mb4e10    PID: 72176
Number of threads: 2
Mesh geometry: 32 x 32 x 32 = 32 768 cells
Checkpoint file: /Users/mb4e10/my_project/sp3-vortex-seed0000.restart
Boxsi run end.
Marijans-MBP:my_project mb4e10$ ls
sp3-vortex-seed0000.odt stdprob3.mif
Marijans-MBP:my_project mb4e10$
```

1. configuration file

2. run OOMMF

3. output file

STEP 3: READ RESULTS



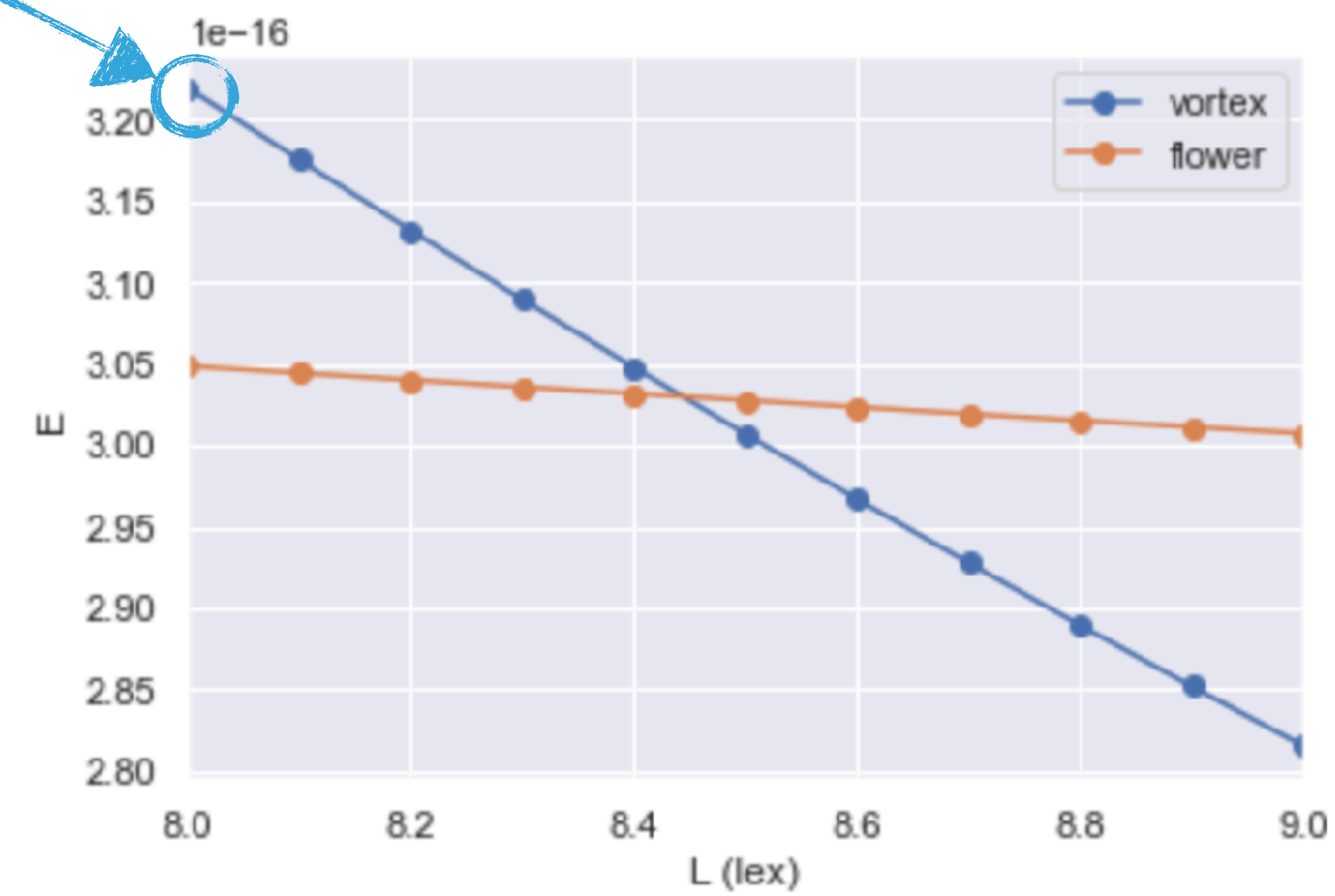
```
# ODT 1.0
# Table Start
# Title: mmArchive Data Table, Wed Nov 16 20:54:28 GMT 2016
# Columns: {Oxs_CGEvolve::Max mxHxm} {Oxs_CGEvolve::Total energy} {Oxs_CGEvolve::Delta E} {Oxs\
_CGEvolve::Bracket count} {Oxs_CGEvolve::Line min count} {Oxs_CGEvolve::Conjugate cycle count}\
 {Oxs_CGEvolve::Cycle count} {Oxs_CGEvolve::Cycle sub count} {Oxs_CGEvolve::Energy calc count}\
 Oxs_UniaxialAnisotropy::Energy Oxs_UniformExchange::Energy {Oxs_UniformExchange::Max Spin Ang\
} {Oxs_UniformExchange::Stage Max Spin Ang} {Oxs_UniformExchange::Run Max Spin Ang} Oxs_Demag:\
:Energy    Oxs_MinDriver::Iteration {Oxs_MinDriver::Stage iteration} Oxs_MinDriver::Stage Oxs_\
MinDriver::mx    Oxs_MinDriver::my    Oxs_MinDriver::mz
# Units:              A/m                         J                          J                \
         {}                             {}                                {}                  \
             {}                            {}                               {}                \
               J                             J                              deg               \
                     deg                                      deg                           J \
                      {}                           {}                         {}              \
   {}                   {}                   {}
              0.00097778028256529097         3.2257415663518404e-16       0                   \
       353                            326                               7                     \
           340                           333                              680                 \
             5.4172367330709765e-17          1.780007679106069e-16             11.011344278380\
658                     90.000000000000014                       90.000000000000014           \
  9.0401021393867362e-17    670                          670                        0         \
           0.4091212671772015  4.6139202285652408e-17 -1.9801501518039851e-16
# Table End
```

```
-:---F1  sp3-vortex-seed0000.odt    All L1     (Fundamental)-----------------------------------
File mode specification error: (error "Buffer format not recognized")
```

vortex energy

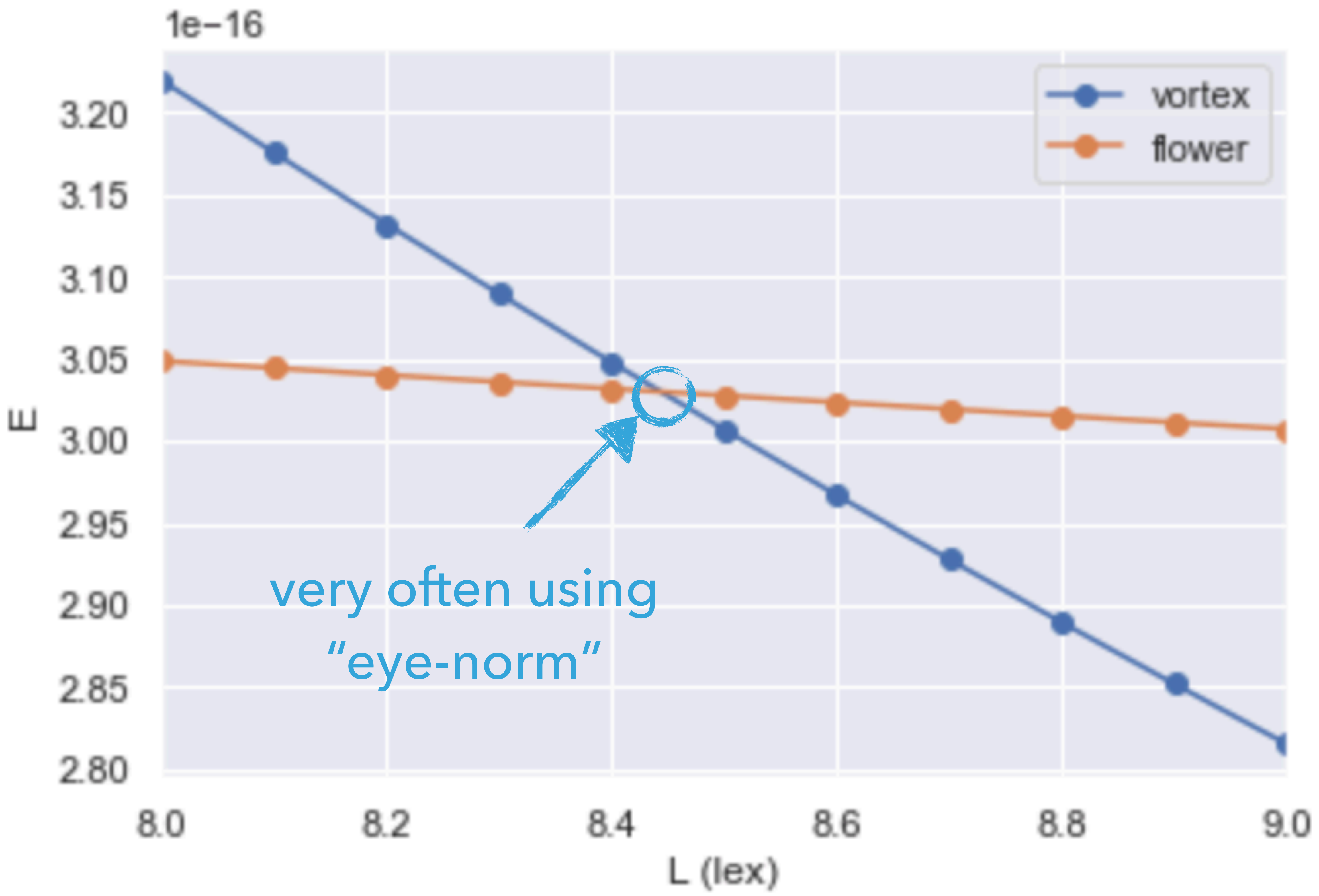
LOOP THROUGH STEPS 1, 2, 3…

L	flower	vortex
8.0	?	$3.23\text{x}10^{-16}$
8.1	?	?
8.2	?	?
8.3	?	?
8.4	?	?
8.5	?	?
8.6	?	?
8.7	?	?
8.8	?	?
8.9	?	?
9.0	?	?



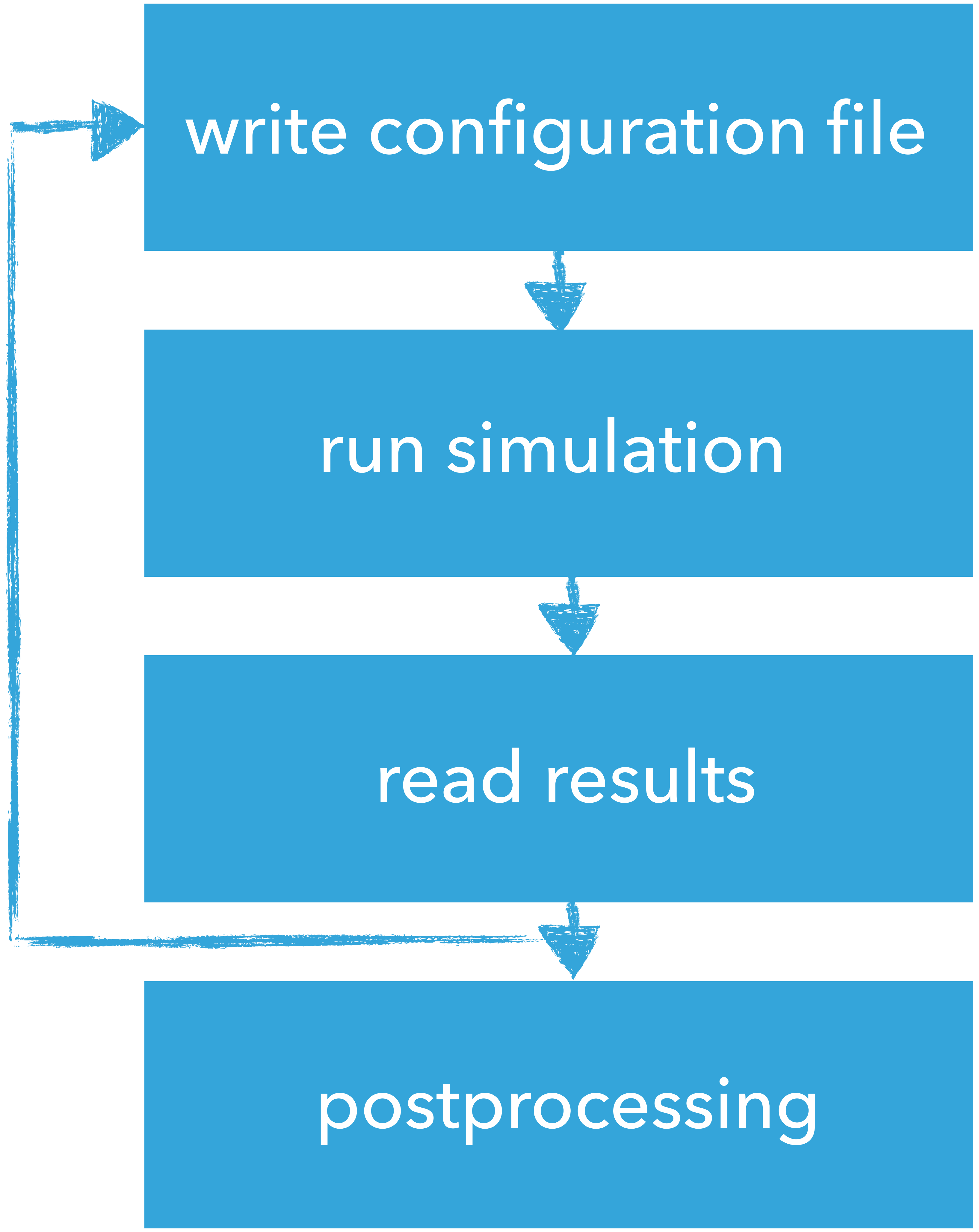
LAST STEP: POSTPROCESSING

- After we obtained all data points, we **plot the results and find crossing**.
- For this step, we often use **separate** plotting scripts or graphical user interface (GUI).



very often using
“eye-norm”

WORKFLOW SUMMARY



write configuration file

run simulation

read results

postprocessing

WHAT COULD BE THE PROBLEMS WITH THIS WORKFLOW?

1. **Time** consuming
    - It requires a lot of user input - **many manual steps**
2. **Keeping log** of all steps that were run and in what order
    - I clicked here, then I changed that, then I fixed that…
3. **Difficult to re-execute** automatically
4. Separate **postprocessing scripts**
    - Every group has their own scripts with different dependencies
5. **Sharing** the exact workflow
6. **Reproducibility**
7. **Very difficult to automate**
8. Very **steep learning curve**

ubermag

PART 4

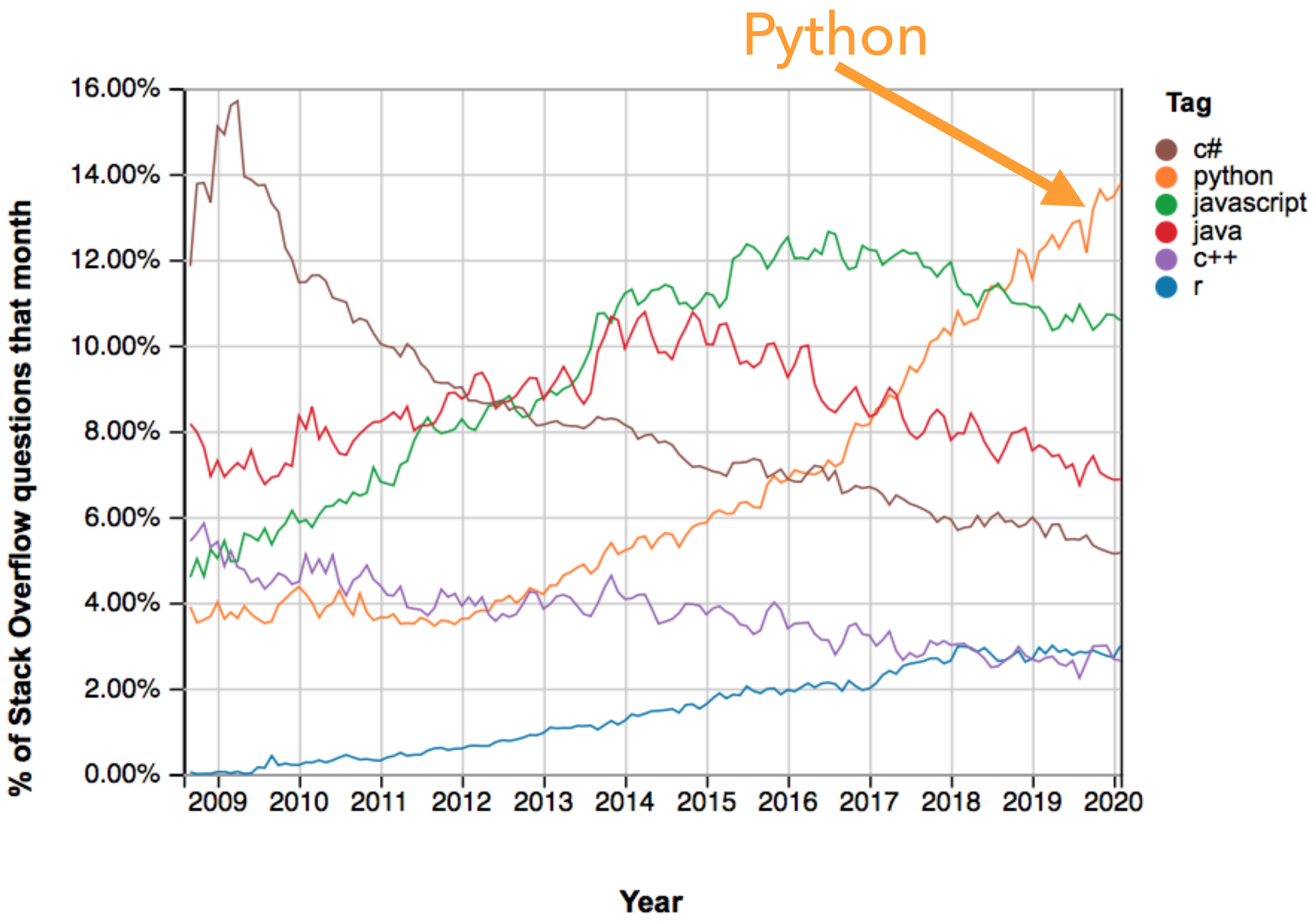
UBERMAG

UBERMAG

> *"… provides Python interface to OOMMF and mumax3 (for now), exposes micromagnetic simulations to Python's scientific ecosystem, and embeds them into Jupyter notebook."*

WHY DID WE CHOOSE PYTHON?

- **Modern** programming language
- The language core is **easy to read and easy to learn**
- Increasingly popular in **software engineering**
- The most popular in **computational and data science**
- Very **well documented** and well supported
- Interpreted language
- www.python.org



Source: https://towardsdatascience.com

SCIENTIFIC PYTHON ECOSYSTEM

- **numpy**: linear algebra
- **scipy**: numerical analysis
- **matplotlib**: 2d (and some 3d) plotting
- **pandas**: big data for Python
- **scikit-learn**: machine learning
- **Jupyter Notebook**

- No need to reinvent the wheel.

JUPYTER

- **Executable document**
- Text, equations, images, code, and results in a **single document**
- Easily **shared**
- Easily **reproducible**
- Hosted in **web browser**
- Can be **run in the cloud** (Binder)
- www.jupyter.org

ubermag

PART 5

DEMO

ubermag

PART 6

SUMMARY

WHAT ARE THE BENEFITS OF USING UBERMAG?

- Ability to drive micromagnetic simulations from **Python**
  - **Scriptability** of computational studies
  - Use of the **Python ecosystem** for computational and data science (numpy, scipy, pandas, …)
- Integration with Jupyter Notebook
  - **Rich media representation** of equations, meshes, fields
  - **Widgets** to explore data sets interactively in notebook
  - Easier **reproducibility**: Notebook contains complete simulation study
  - **Sharing** of interactive documents through MyBinder

COMPUTER SCIENCE PERSPECTIVE ON THE USER INTERFACE

- Python libraries created are a **Domain Specific Language (DSL)** for micromagnetic science
- This DSL is **embedded in general purpose programming language** (Python)
  - More powerful than (i) hard coded parameters, or (ii) config files
  - But also high complexity: users can combine library functions in all possible ways
- **Framework to include more micromagnetic computational solvers** (for example mumax3, micromagnum, fidimag)
- **Publication**: M. Beg, R. A. Pepper, and H. Fangohr. User interfaces for computational science: A domain specific language for OOMMF embedded in Python. *AIP Advances* **7**, 56025 (2017). https://doi.org/10.1063/1.4977225

WORKSHOPS

- Generally **well received**
- **Scientists without programming experience struggle** with Python in Notebook setup: many new concepts at the same time
- **Ubermag in the cloud** (JupyterHub, MyBinder) very effective for workshop delivery

RESOURCES

- **Website**: ubermag.github.io
- **How to start**:
  - Ubermag YouTube channel
  - Workshop repository: https://github.com/ubermag/workshop
- **Publication**: M. Beg, R. A. Pepper, and H. Fangohr. User interfaces for computational science: A domain specific language for OOMMF embedded in Python. AIP Advances 7, 56025 (2017). https://doi.org/10.1063/1.4977225

ACKNOWLEDGEMENTS

- **Contributors**: Marijan Beg, Martin Lang, Sergii Mamedov, Ryan A. Pepper, David Cortés-Ortuño, Thomas Kluyver, Hans Fangohr
- **Financial support**:
  - **OpenDreamKit Horizon 2020**, European Research Infrastructures project (#676541),
  - EPSRC's **Skyrmion Programme Grant** (EP/N032128/1),
  - EPSRC's Centre for Doctoral Training in **Next Generation Computational Modelling**, (#EP/L015382/1), and
  - **The Gordon and Betty Moore Foundation** through Grant GBMF #4856, by the Alfred P. Sloan Foundation and by the Helmsley Trust.

EPSRC
Engineering and Physical Sciences
Research Council