OpenDreamKit
Virtual Research Environment

Marijan Beg* and H. Fangohr

Faculty of Engineering and the Environment, **University of Southampton**, Southampton, UK

* email: m.beg@soton.ac.uk

Ljubljana, Slovenia (November 2016)

OpenDreamKit

- Horizon 2020 European Research Infrastructure project
- 4 years
- Started in September 2015 (just over a year now)
- 50 people in over 16 European cities
- Goal: Develop **Virtual Research Environments** in pure mathematics and applications, supporting the full research cycle

VIRTUAL RESEARCH ENVIRONMENT

OUTLINE

1. Computational science
2. Conventional workflow (example and problems)
3. Virtual Research Environment Example (Jupyter, JOOMMF, benefits)
4. Summary

COMPUTATIONAL SCIENCE

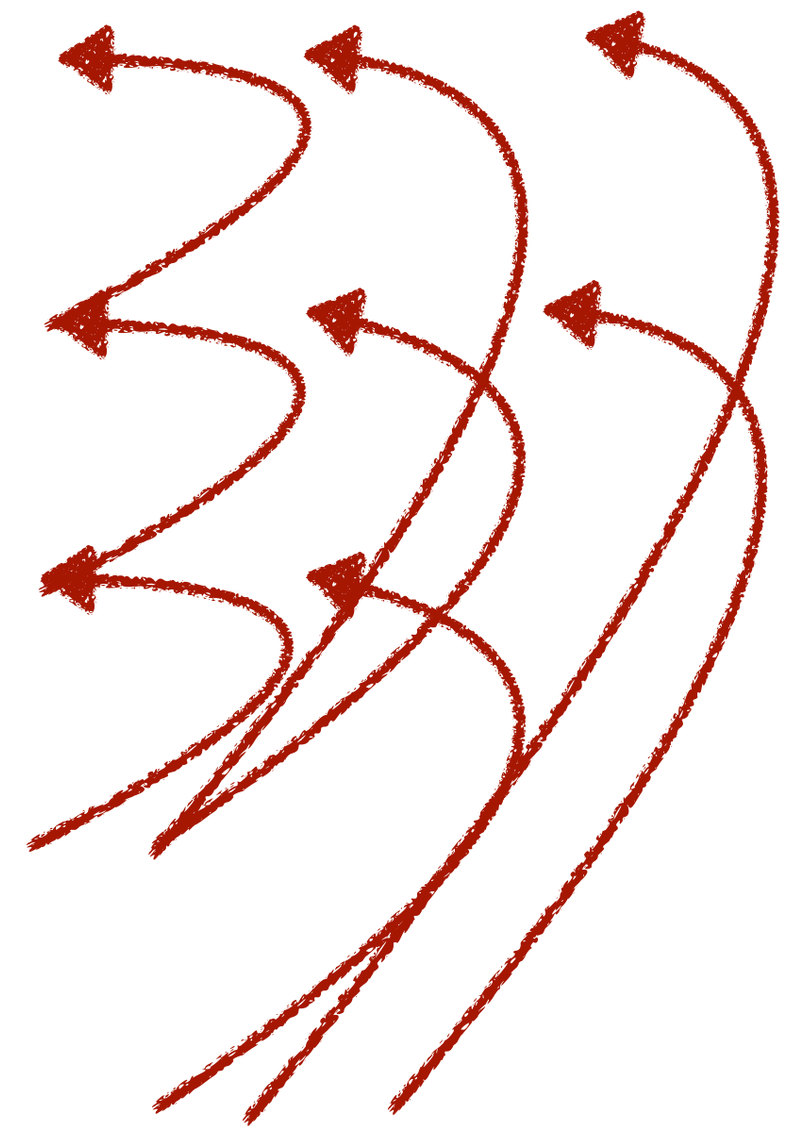
- Very often the only feasible way to address some research challenges
- Complement theory and experiments
- Emerging as a third pillar in research and development
- It is becoming more used and accepted as models, computational power, and simulation techniques advance

CONVENTIONAL COMPUTATIONAL WORKFLOW

1. write simulation code
2. run code
3. data analysis and visualisation
4. write paper
5. share & publish

CONVENTIONAL COMPUTATIONAL WORKFLOW

1. write simulation code
2. run code
3. data analysis and visualisation
4. write paper
5. share & publish



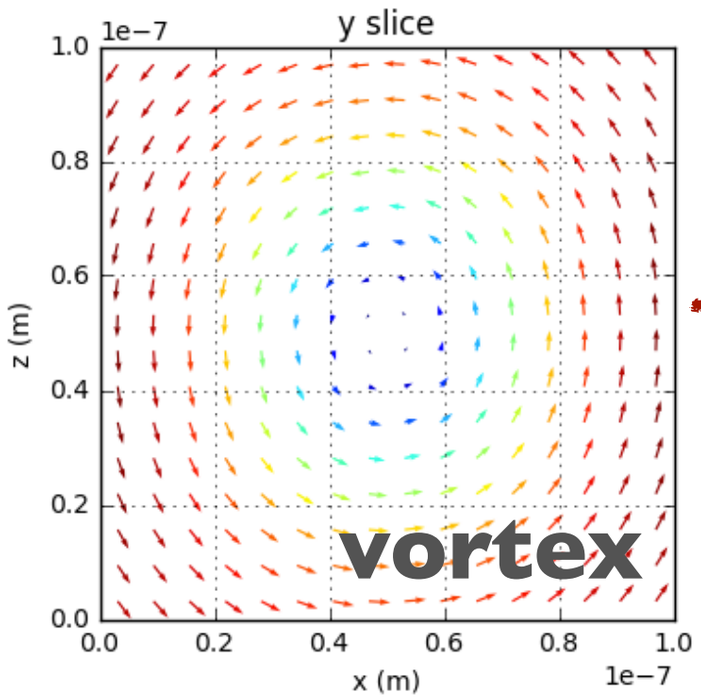
MICROMAGNETIC EXAMPLE

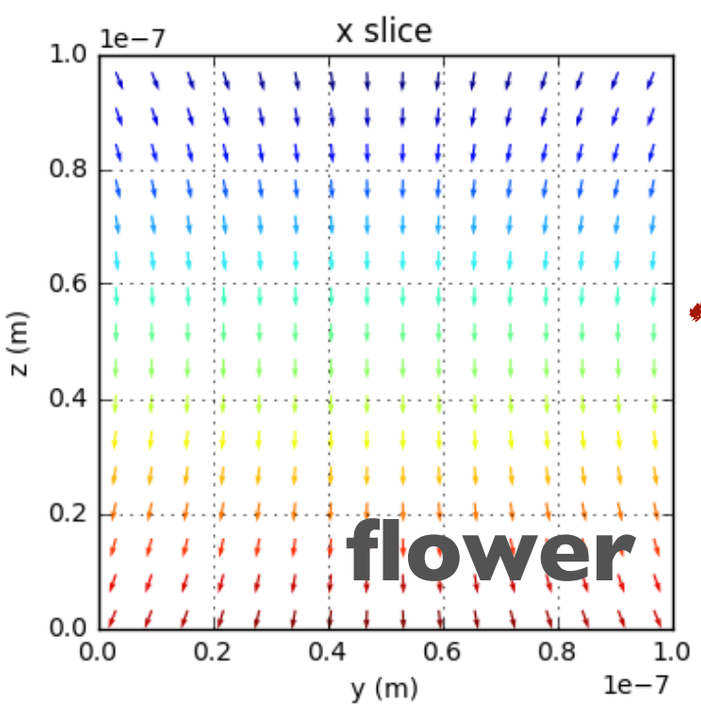
MICROMAGNETICS

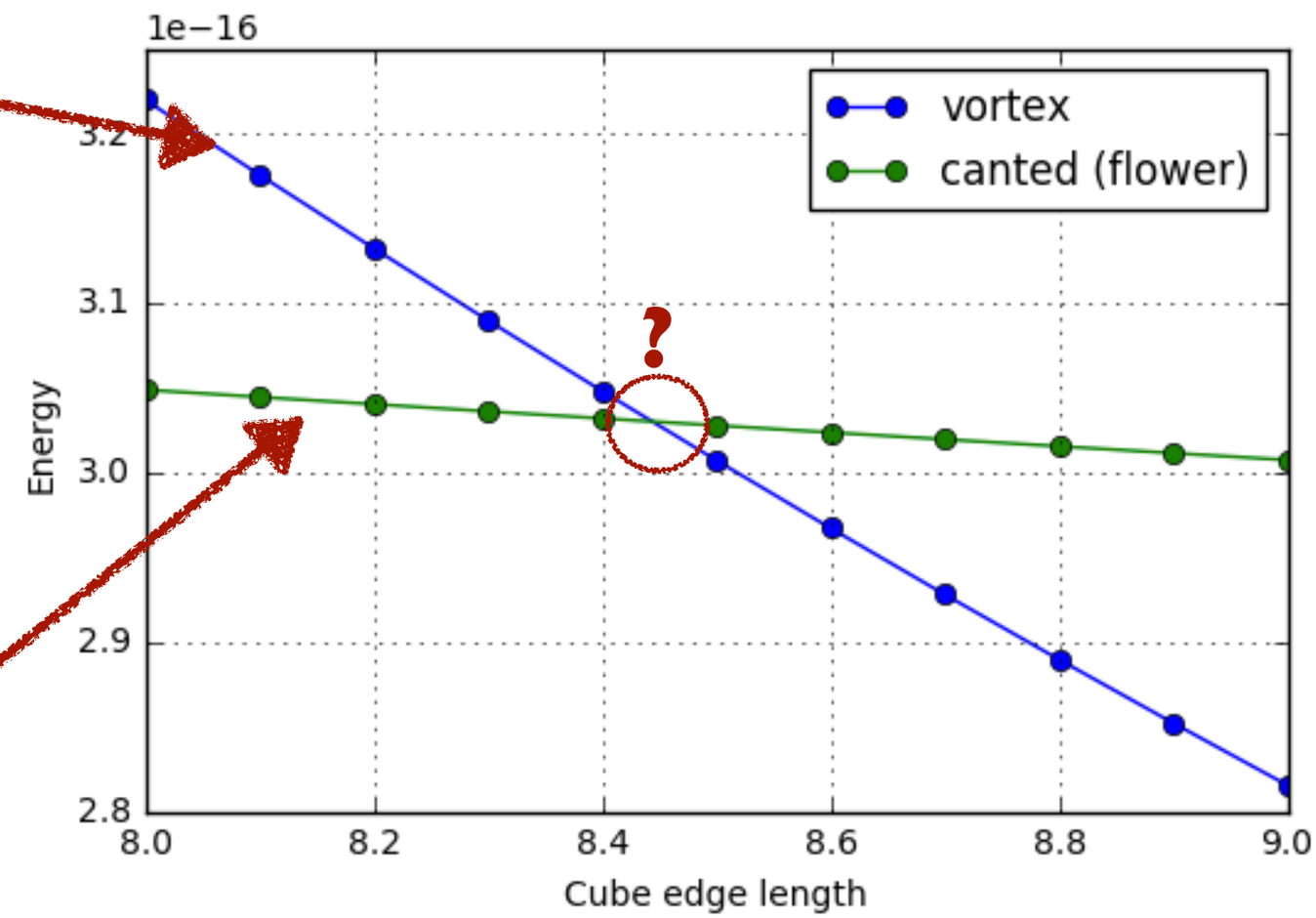
- Micromagnetics studies magnetic phenomena at micro- and nanoscale
- For instance, used in the design and engineering of magnetic storage devices (hard disks)
- Widely used in academic and industrial communities
- Very often the only possible way of addressing particular research challenges

RESEARCH QUESTION

For what edge length,
vortex and flower states have the same energy?



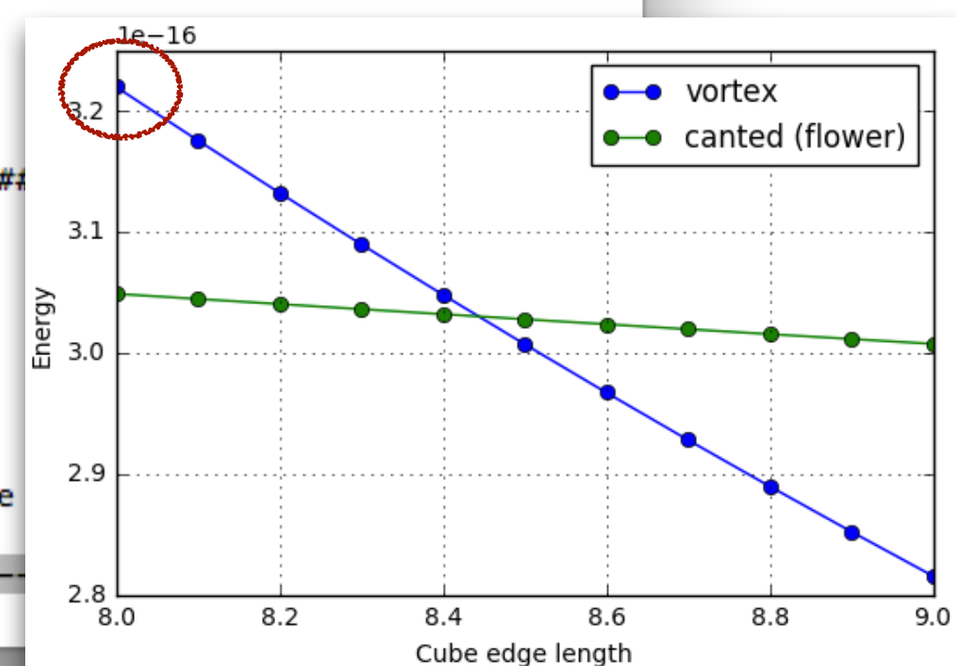




STEP 1: WRITE SIMULATION CONFIGURATION

my_project — IPython: Users/mb4e10 — emacs -nw stdprob3.mif — 95×37

```
# 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
# exchange length based on parameters L and N.
-uu-:---F1  stdprob3.mif    Top L1     (Fundamental)--------------
```



STEP 2: RUN SIMULATION

my_project — IPython: Users/mb4e10 — -bash ▸ python — 95×37

```
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$
```

STEP 3: READ DATA

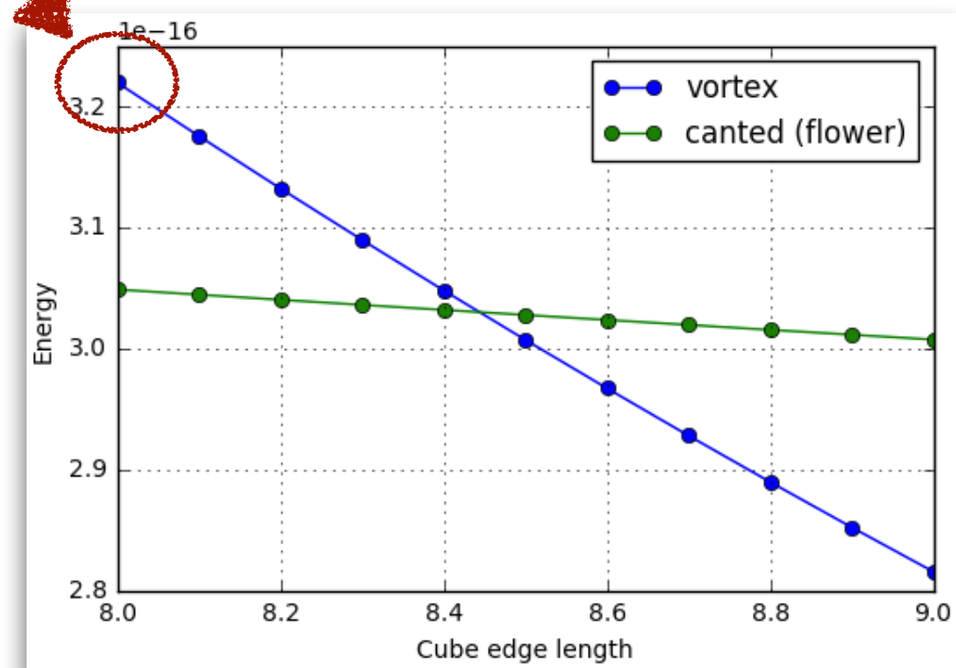
my_project — IPython: Users/mb4e10 — emacs -nw sp3-vortex-seed0000.odt — 95×37

```
# 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         \
           0.40912126717720015  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")
```

REPEAT SIMULATIONS…

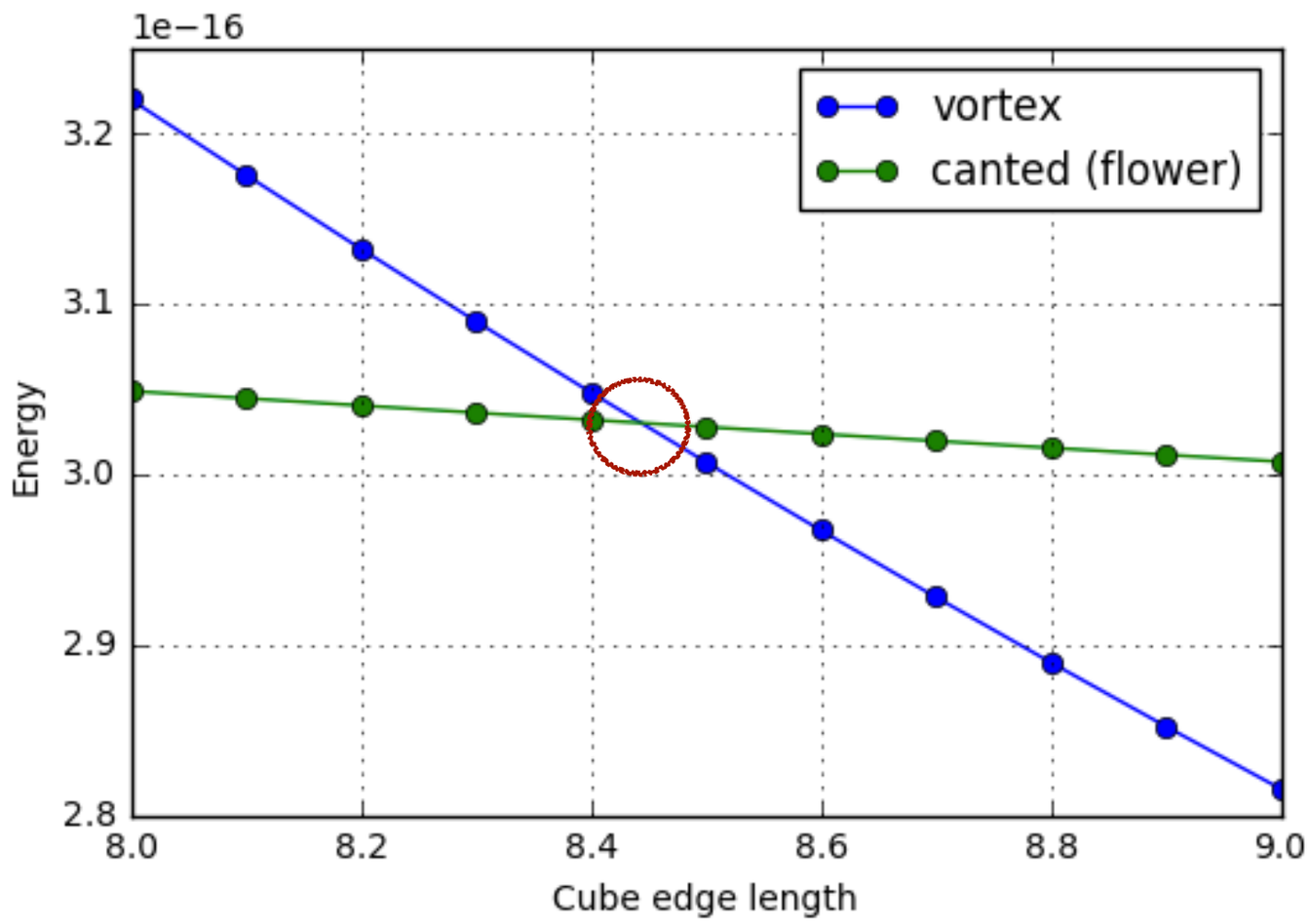
L	flower	vortex
8.0	?	3.23×10^{-16}
8.1	?	?
8.2	?	?
8.3	?	?
8.4	?	?
8.5	?	?
8.6	?	?
8.7	?	?
8.8	?	?
8.9	?	?
9.0	?	?



We have to repeat steps 1, 2, and 3 to obtain other 21 points

POSTPROCESSING

- We plot the data we obtained by running separate plotting scripts or by using some Graphical User Interfaces (Python, MATLAB, Excel, Origin…)



- Find crossing

PROBLEMS WITH THIS SIMPLE EXAMPLE WORKFLOW

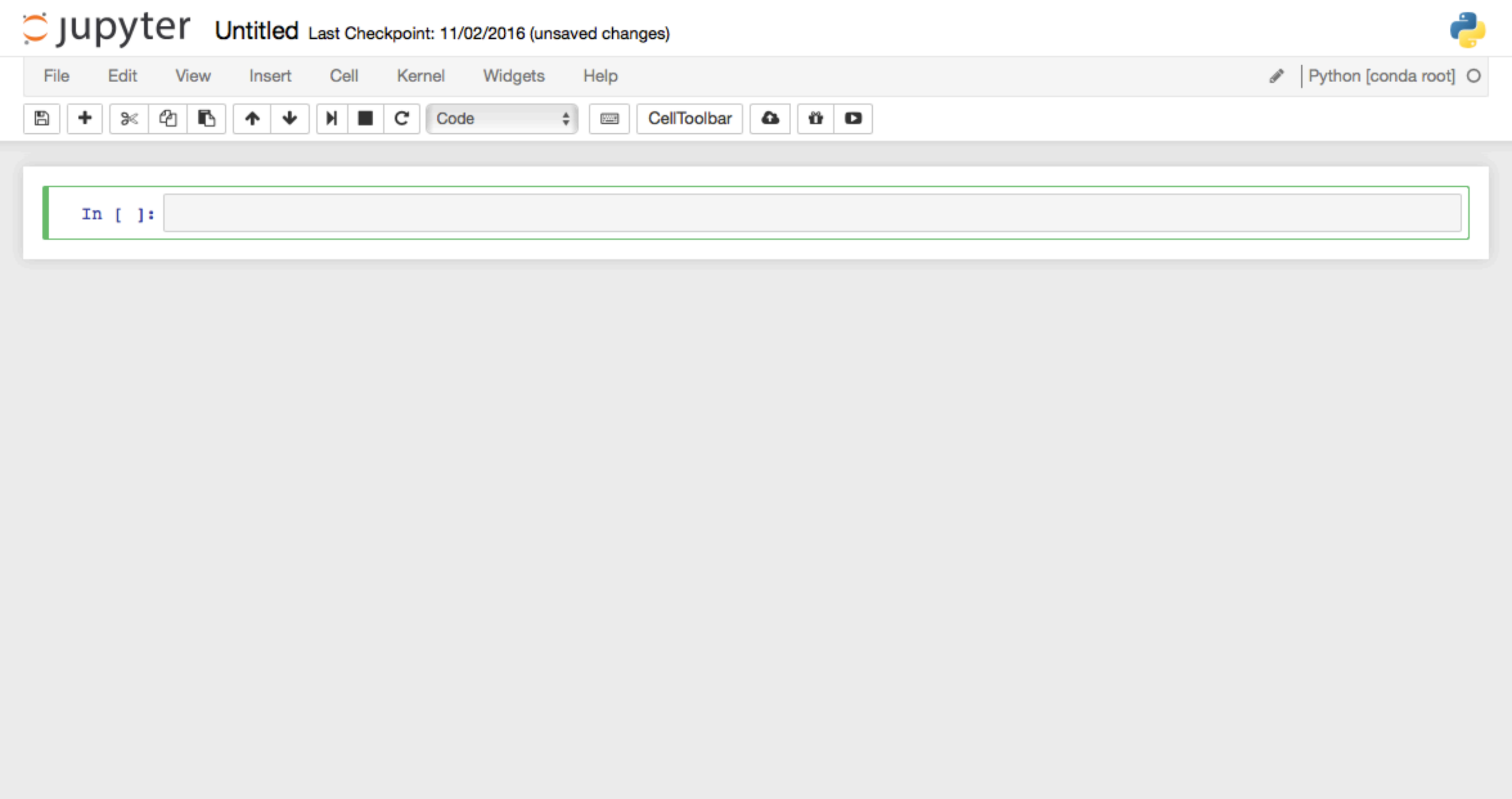
1. Time consuming
2. Keeping log of all steps that were run and in what order
3. Necessary to write postprocessing scripts (well tested?)
4. Collaboration?
5. Reproducibility?
6. Abusing instead of using simulation

VIRTUAL RESEARCH ENVIRONMENT WORKFLOW

JOOMMF

- Micromagnetic Virtual Research Environment
- Make running OOMMF simulations in Jupyter notebook possible
- Jupyter + OOMMF = JOOMMF
- Domain specific language embedded in general purpose language
- In the first stage, we developed a Python wrapper for OOMMF

JUPYTER NOTEBOOK



TEXT AND EQUATIONS

jupyter vre_workflow Last Checkpoint: 11/02/2016 (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Python [conda root]

Markdown CellToolbar

```
# JOOMMF workflow in Virtual Research Environment

## Problem specification

This problem is to calculate a single domain limit of a cubic magnetic particle. This is the size $L$ of equal energy
for the so-called flower state (which one may also call a splayed state or a modified single-domain state) on the one
hand, and the vortex or curling state on the other hand.

Geometry:

A cube with edge length, $L$, expressed in units of the intrinsic length scale, $l_\text{ex} = \sqrt{A/K_\text{m}}$,
where $K_\text{m}$ is a magnetostatic energy density, $K_\text{m} = \frac{1}{2}\mu_{0}M_\text{s}^{2}$.

Material parameters:

- uniaxial anisotropy $K_\text{u}$ with $K_\text{u} = 0.1 K_\text{m}$, and with the easy axis directed parallel to a
principal axis of the cube (0, 0, 1),
- exchange energy constant is $A = \frac{1}{2}\mu_{0}M_\text{s}^{2}l_\text{ex}^{2}$.

More details about the standard problem 3 can be found in Ref. 1.
```

TEXT AND EQUATIONS

jupyter vre_workflow Last Checkpoint: 11/02/2016 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Python [conda root]

Markdown CellToolbar

JOOMMF workflow in Virtual Research Environment

Problem specification

This problem is to calculate a single domain limit of a cubic magnetic particle. This is the size L of equal energy for the so-called flower state (which one may also call a splayed state or a modified single-domain state) on the one hand, and the vortex or curling state on the other hand.

Geometry:

A cube with edge length, L, expressed in units of the intrinsic length scale, $l_{\mathrm{ex}} = \sqrt{A/K_{\mathrm{m}}}$, where K_{m} is a magnetostatic energy density, $K_{\mathrm{m}} = \frac{1}{2}\mu_0 M_{\mathrm{s}}^2$.

Material parameters:

- uniaxial anisotropy K_{u} with $K_{\mathrm{u}} = 0.1K_{\mathrm{m}}$, and with the easy axis directed parallel to a principal axis of the cube (0, 0, 1),
- exchange energy constant is $A = \frac{1}{2}\mu_0 M_{\mathrm{s}}^2 l_{\mathrm{ex}}^2$.

More details about the standard problem 3 can be found in Ref. 1.

CODE

jupyter standard_problem3 Last Checkpoint: 11/02/2016 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Python [default]

Code | CellToolbar

- exchange energy constant is $A = \frac{1}{2}\mu_0 M_s^2 l_{ex}^2$.

More details about the standard problem 3 can be found in Ref. 1.

Simulation

Firstly, we import all necessary modules.

In [1]:
```
import discretisedfield as df
import oommfc as oc
```

The following two functions are used for initialising the system's magnetisation [1].

In [2]:
```
import numpy as np

# Function for initiaising the flower state.
def m_init_flower(pos):
    x, y, z = pos[0]/1e-9, pos[1]/1e-9, pos[2]/1e-9
    mx = 0
    my = 2*z - 1
    mz = -2*y + 1
    norm_squared = mx**2 + my**2 + mz**2
    if norm_squared <= 0.05:
        return (1, 0, 0)
    else:
```

VISUALISATION

jupyter standard_problem3 Last Checkpoint: 11/02/2016 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Python [default]

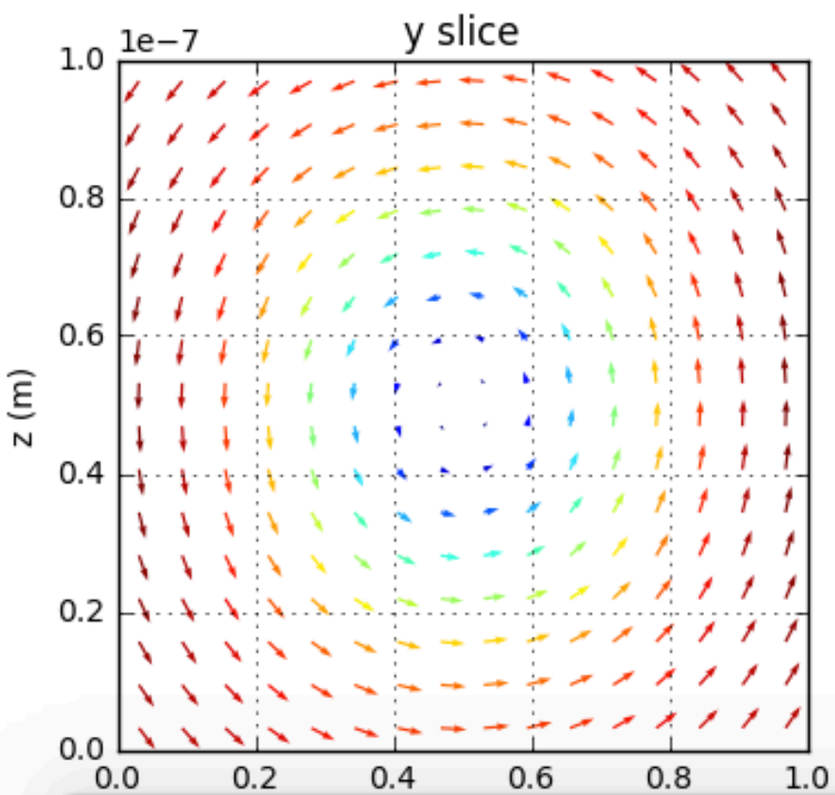
Code CellToolbar

Relaxed magnetisation states

Now, we show the magnetisation configurations of two relaxed states.

Vortex state:

In [11]:
```
%matplotlib inline
system = minimise_system_energy(8, m_init_vortex)
fig = system.m.plot_slice('y', 50e-9, xsize=4)
```



MULTIPLE SIMULATION RUNS

jupyter standard_problem3 Last Checkpoint: 11/02/2016 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Python [default]

Code CellToolbar

Energy crossing

Now, we can plot the energies of both vortex and flower states as a function of cube edge length. This will give us an idea where the state transition occurrs.

In [13]:
```
L_array = np.linspace(8, 9, 11)  # values of L for which the system is relaxed.

vortex_energies = []
flower_energies = []

for L in L_array:
    vortex = minimise_system_energy(L, m_init_vortex)
    flower = minimise_system_energy(L, m_init_flower)

    vortex_energies.append(vortex.total_energy())
    flower_energies.append(flower.total_energy())

# Plot the energy dependences.
import matplotlib.pyplot as plt
plt.plot(L_array, vortex_energies, 'o-', label='vortex')
plt.plot(L_array, flower_energies, 'o-', label='flower')
plt.xlabel('L (lex)')
plt.ylabel('E')
plt.xlim([8.0, 9.0])
plt.grid()
plt.legend()
```

PLOTTING

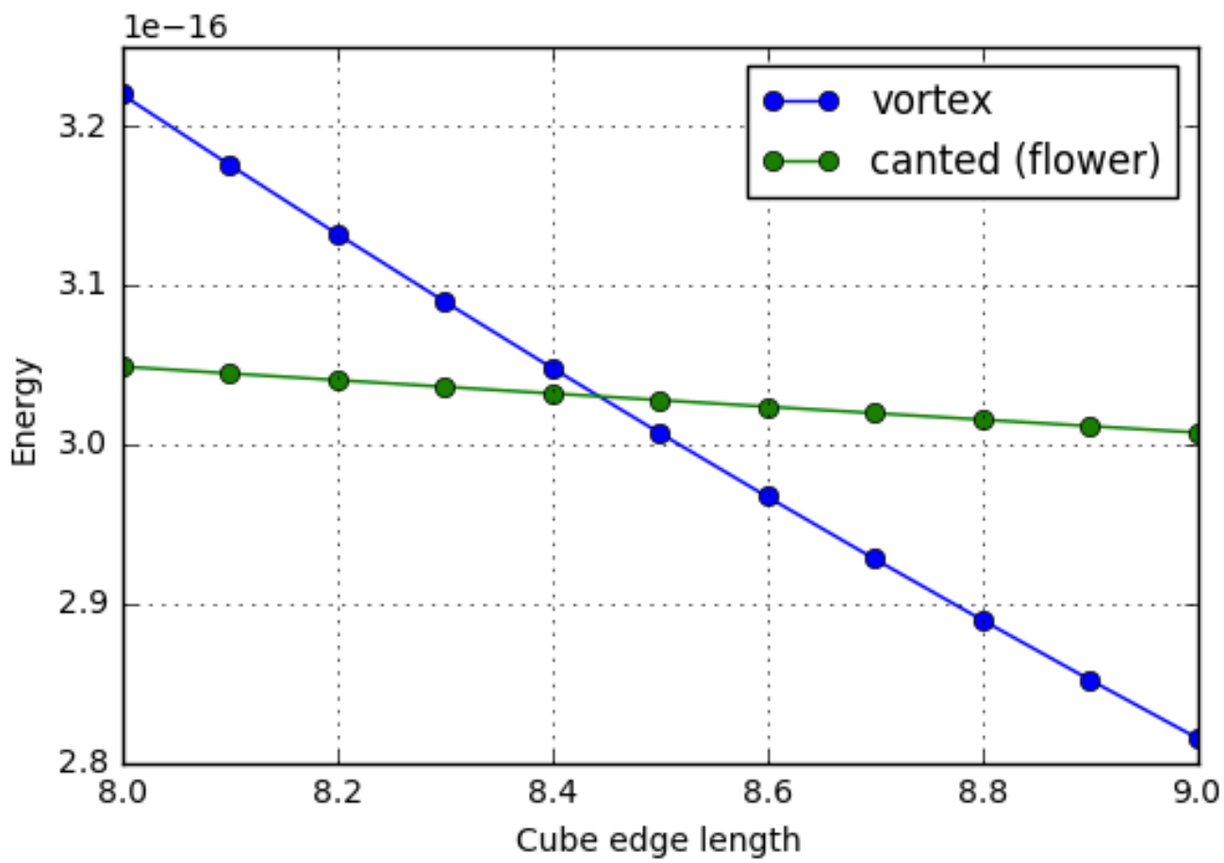
jupyter standard_problem3 Last Checkpoint: 11/02/2016 (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Python [default]

Markdown CellToolbar

```
plt.legend()
```

```
Out[8]: <matplotlib.legend.Legend at 0x112caab70>
```

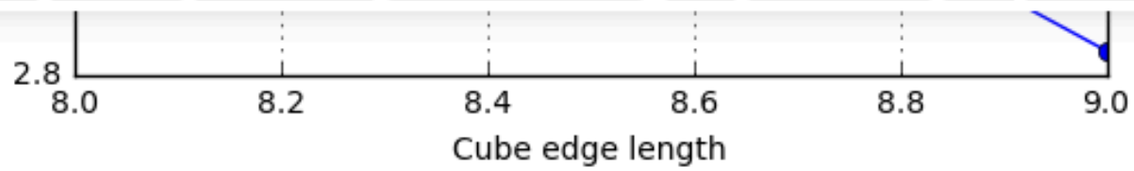


ENERGY CROSSING

jupyter standard_problem3 Last Checkpoint: 11/02/2016 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Python [default]

Markdown CellToolbar



We now know that the energy crossing occurrs between $8l_{\text{ex}}$ and $9l_{\text{ex}}$, so a bisection algorithm can be used to find the exact crossing.

In [9]:
```
from scipy.optimize import bisect

def energy_difference(L):
    vortex = minimise_system_energy(L, m_init_vortex)
    flower = minimise_system_energy(L, m_init_flower)

    return vortex.total_energy() - flower.total_energy()

cross_section = bisect(energy_difference, 8, 9, xtol=0.1)

print("The transition between vortex and flower states occurs at {}*lex".format(cross_section))
```

```
The transition between vortex and flower states occurs at 8.4375*lex
```

References

[1] μMAG Site Directory http://www.ctcms.nist.gov/~rdm/mumag.org.html

TABLES

jupyter standard_problem4 (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Python [default]

Markdown CellToolbar

Postprocessing

When we drove the system using the `TimeDriver`, we specified that we want to save the magnetisation configuration $n = 200$ times. A detailed table of all computed parameters from the last simulation can be shown from the datatable (`system.dt`), which is a `pandas` dataframe [2].

For instance, if we want to show the last 10 rows in the table, we run:

In [16]:
```
system.dt.tail(5)
```

Out[16]:

	E	Ecount	max_dm/dt	dE/dt	deltaE	Eex	max_spin_angle	stage_max_spin_angle	run_max_spin_angle	Ed
195	-2.676805e-18	3802.0	1168.558002	-1.701626e-09	-2.292409e-21	9.509917e-20	3.988227	4.335480	29.984064	8.53911 19
196	-2.685941e-18	3821.0	1264.690715	-1.936455e-09	-2.633442e-21	8.603004e-20	4.234452	4.234452	29.984064	8.85139 19
197	-2.695973e-18	3840.0	1385.550409	-2.055475e-09	-2.820054e-21	8.010709e-20	4.782664	4.782664	29.984064	9.07919 19
198	-2.706283e-18	3859.0	1350.603218	-2.048108e-09	-2.831441e-21	7.558538e-20	4.688752	4.815510	29.984064	9.22294 19
199	-2.716260e-18	3878.0	1189.283501	-1.925038e-09	-2.680093e-21	7.113525e-20	4.377352	4.688752	29.984064	9.29227 19

BENEFITS

- Documentation, computation, visualisation in the same notebook
- The entire workflow is contained in a single documents
- Self documenting
- Easy to share, publish, and collaborate
- Reproducible

SUMMARY

- Current workflows in computational science have many flaws.
- Probably the most important one is the reproducibility.
- Virtual Research Environment allows us to have documentation (text and equations), code, code outputs (text, figures, tables) in a single file
- We have the benefit of using already existing libraries for data analysis and visualisation because we use general purpose language
- Much easier publishing, sharing, collaboration
- Important for reproducibility
- URL: joommf.github.io
- Beg *et al. arXiv* 1609.07432 (2016)
- email: m.beg@soton.ac.uk
- Acknowledge contributions from Hans Fangohr, Ryan A. Pepper, Thomas Kluyver, and Min Ragan-Kelley. Financially supported by OpenDreamKit project