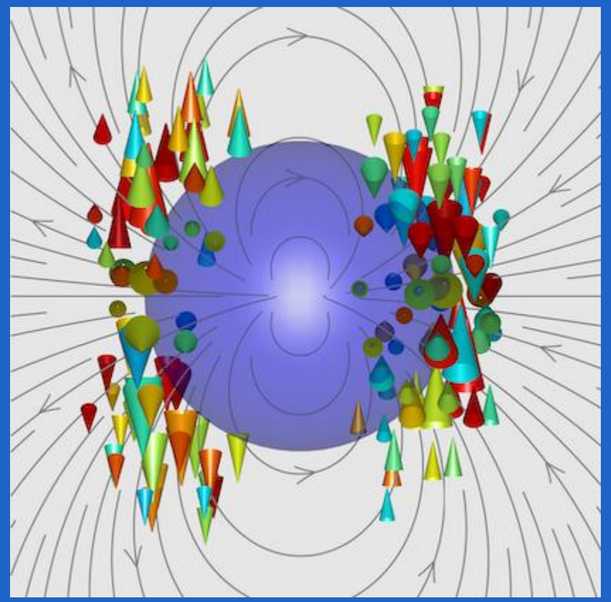
FACULTY OF SCIENCES

DEPARTMENT OF SOLID STATE SCIENCES
RESEARCH GROUP DYNAMAT

Getting started with mumax^3



Dr. Jonathan Leliaert

GHENT UNIVERSITY

Jonathan.Leliaert@ugent.be

mumax^3

GPU-accelerated micromagnetism

CONTENTS

- What is mumax3?
- A few examples
- Hardware & Extra Resources

SOURCE: https://mumax.ugent.be/mumax3-workshop/

The workshop is created by the researchers of the DyNaMat group at Ghent University.

Dr. Jonathan Leliaert

Dr. Jeroen Mulkers

Prof. Dr. Bartel Van Waeyenberge

Drs. Pieter Gypens

GHENT UNIVERSITY

mumax^3

GPU-accelerated micromagnetism

Jonathan.Leliaert@ugent.be

NVIDIA

FACULTY
OF SCIENCES

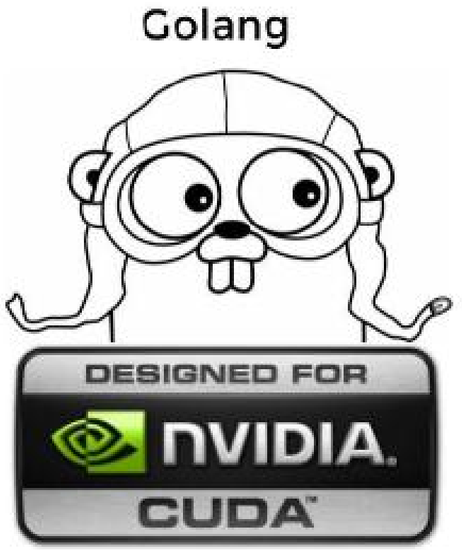
WHAT IS MUMAX3?

GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

WHAT IS MUMAX3?

- Free finite-difference based micromagnetic simulation package
- GPU-accelerated *nvidia GPU required*
- Developed at DyNaMat (Ugent) by Arne Vansteenkiste
- Latest official release mumax3.10 *(Aug 13, 2020)*
- Active community *groups.google.com/forum/#!forum/mumax2*
- Documented API *mumax.github.io*
- Open source (GPLv3) *github.com/mumax/3*
- Mainly written in Go
- CUDA C kernels for heavy lifting
- Scripting language + Web GUI
- Well tested (unit tests + NIST standard problems)

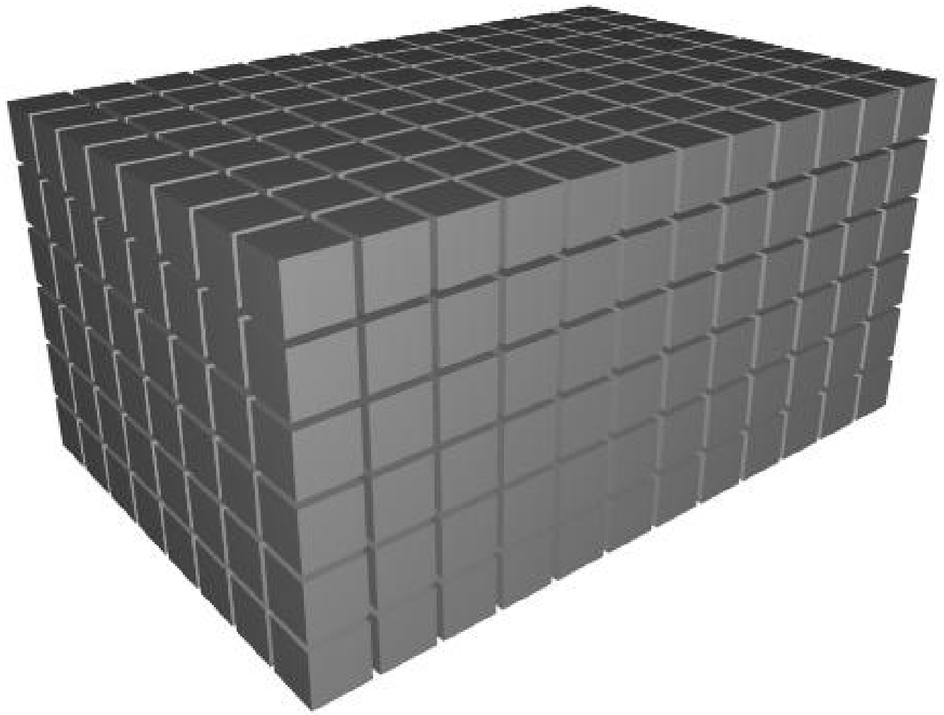


SCRIPTING LANGUAGE

Mumax3's scripting language is a subset of golang

```
// saturation magnetization
Msat = 5e6

// declare new variable
Freq := 1e9

for i:=0; i<10; i++ {
        print(i)
}

if 1+8 == 9 {
        print("Of course 1+8=9")
}
```

DISCRETIZATION



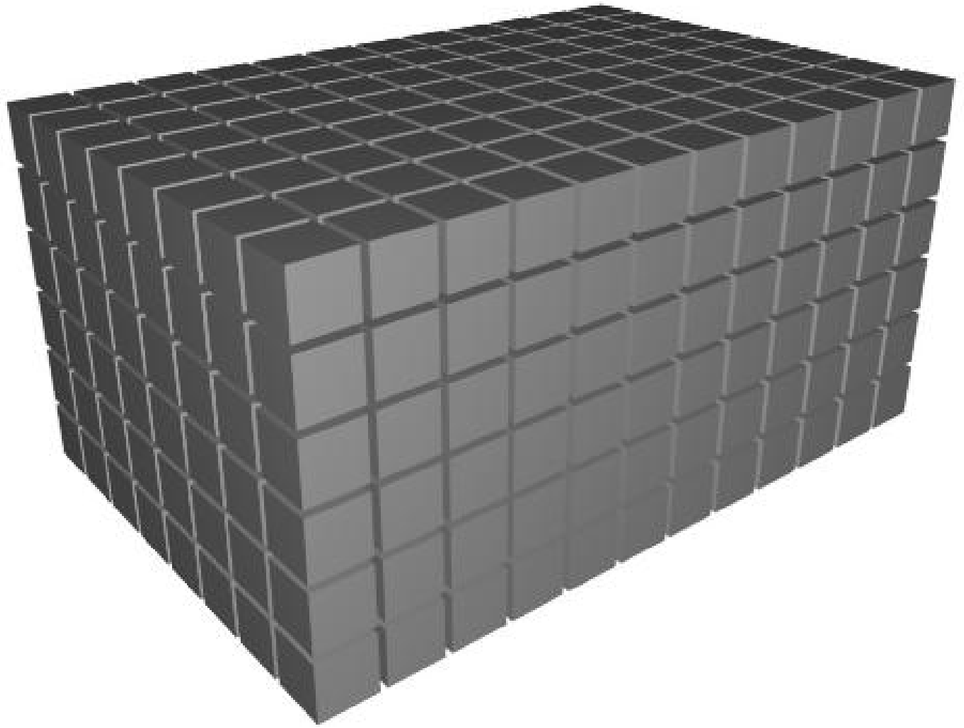
- Rectangular simulation box (origin in the center)
- Single regular rectangular grid
- Uniform magnetization inside cell $\boldsymbol{m}_{ijk} = \boldsymbol{m}(x_i, y_j, z_k)$
- Cell size < exchange length

```
setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
```

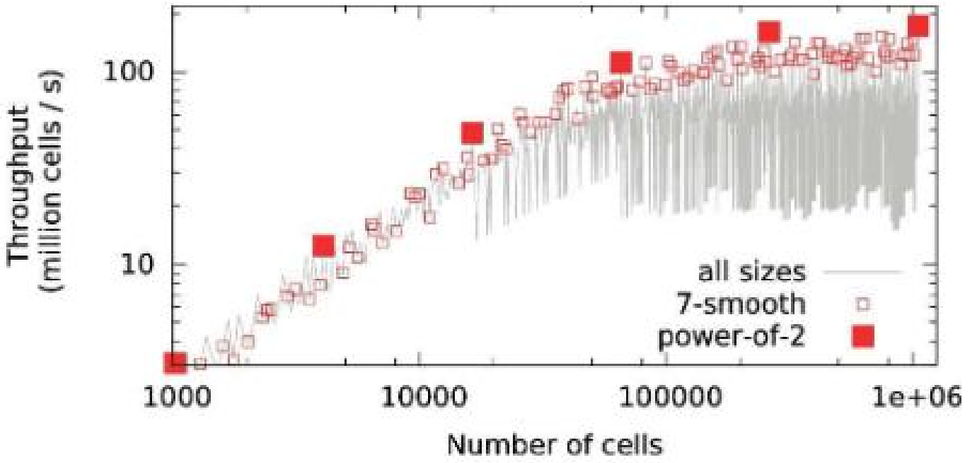
- PBC values are number of virtual repetitions of the simulation box used to calculate dipolar interactions

```
setpbc(4,0,0)
```

DISCRETIZATION



The cuda fft library (used for the computation of the demag field) is highly optimized for grid size dimensions with small prime factors.



Tip: try to use grid size dimensions which are '7-smooth'

Example of good and bad grid size dimensions:

$190 = 2 \cdot 5 \cdot 19$ $191 = 191$ $192 = 2^6 \cdot 3$



SHAPES

- A shape can be considered as a function $f: \mathbb{R}^3 \rightarrow \{true, false\}$ where

$$f(x,y,z) = \begin{cases} true \text{ if } (x,y,z) \text{ in shape} \\ false \text{ otherwise} \end{cases}$$

- Most shapes do not depend on the grid
- Default location: center of universe (0,0,0)
- Large set of predefined basic shapes
- Define new shapes by combining and modifying the basic shapes
- Shapes are useful for different tasks:
  - The geometry
  - Defining regions
  - To set locally an initial magnetization

SHAPES

Shapes

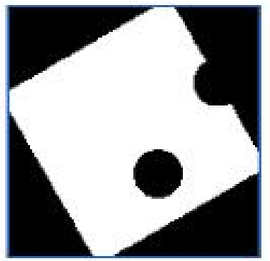
Cell(j,k,l)
Circle(diameter)
Cone(diameter,height)
Cuboid(Lx,Ly,Lz)
Cylinder(diameter,height)
Ellipse(a,b)
Ellipsoid(a,b,c)
ImageShape(filename)
Layer(i)
Layers(i1,i2)
Rect(Lx,Ly)
Square(L)
Xrange(xmin,xmax)
Yrange(ymin,ymax)
Zrange(zmin,zmax)

Shape methods

Transl(dx,dy,dz)
Scale(sx,sy,sz)
RotX(angle)
RotY(angle)
RotZ(angle)
Repeat(dx,dy,dz)

Add(shape)
Sub(shape)
Inverse()
Intersect(shape)
Xor(shape)

```
// Rotated cheese example

d  := 200e-9
sq := square(d)

h      := 50e-9
hole   := cylinder(h, h)
hole1 := hole.transl(100e-9, 0, 0)
hole2 := hole.transl(0, -50e-9, 0)

cheese := sq.sub(hole1).sub(hole2)
cheese = cheese.rotz(pi/6)
```



rotated cheese

GEOMETRY

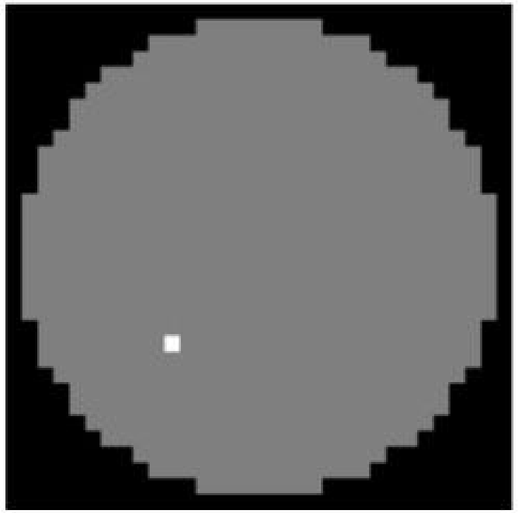
Optionally a magnet Shape other than the full simulation box can be specified

```
// Ring geometry example

setGridsize(100, 100, 10)
setCellsize(1e-9,1e-9,1e-9)

ring := circle(100e-9).sub(circle(50e-9))

setgeom(ring)

save(geom)
```

REGIONS

- 256 regions in total (index 0→ 255)
- Each cell is assigned to a single region (default region id is 0)
- Each region has its own set of material parameters
- Two ways to set the region id in cells:
  1. Set region id of a single cell
  2. Set region id of all cells in a shape

```
SetGridSize(32,32,1)
SetCellSize(1,1,1)

// Set region id of cells in a circle to 1
DefRegion(1, circle(30))

// Set region id of cell (10,10,0) to 2
DefRegionCell(2, 10, 10, 0)

Save(regions)
```



GHENT UNIVERSITY

Jonathan.Leliaert@ugent.be

MATERIAL PARAMETERS

- Material parameters are assigned to the 256 regions.
- Material parameters can be functions of time
- There are vector and scalar material parameters
- Material parameters are predefined, they can not be created

```
// Assigning to a material parameter sets a value in all regions:
Msat  = 800e3
AnisU = vector(1, 0, 0)

// When regions are defined, they can also be set region-wise:
Msat.SetRegion(0, 800e3)
Msat.SetRegion(1, 540e3)

// Material parameters can be functions of time as well:
f := 500e6
Ku1 = 500 * sin(2*pi*f*t)
```

MATERIAL PARAMETERS: EXCITATIONS

- An excitation is a regional material parameter
- Additionally, one can add an arbitrary number of time- and space-dependent vector fields of the form :

$$g(x_i, y_j, z_k) * f(t)$$

```
B_ext = vector(0,0,1)

B_ext.Add(LoadFile("antenna.ovf"), sin(2*pi*f*t))

B_ext.removeExtraTerms()
```

INITIAL MAGNETIZATION

Different ways to set the magnetization:

```
m = config
m.LoadFile(filename)
m.SetRegion(regionId, config)
m.SetInShape(shape, config)
m.SetCell(j,k,l, vector)
```

Here, a ‘config’ is an object which represents a magnetization configuration

INITIAL MAGNETIZATION

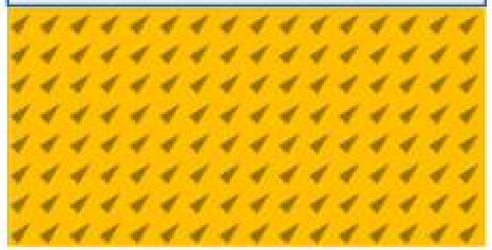
Config

Uniform(mx, my, mz)
RandomMag()
RandomMagSeed(seed)
TwoDomain(
  mx1, my1, mz1,
  my2, my2, mz2,
  mx3, my3, mz3)
Vortex(circ, pol)
AntiVortex(circ, pol)
VortexWall(mxLeft, mxRight, circ, pol)
NeelSkyrmion(charge, pol)
BlochSkyrmion(charge, pol)
Conical(kVec, coneDir, coneAngle)
Helical(kVec)

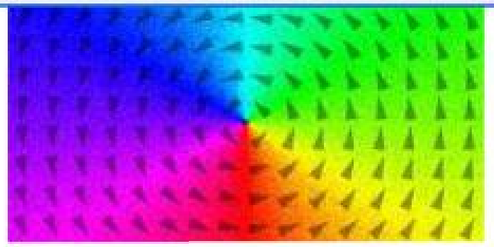
Config methods

Transl(dx,dy,dz)
Scale(sx,sy,sz)
Add(ratio, config)
RotZ(angle)

```
m = uniform(1,1,0)
```

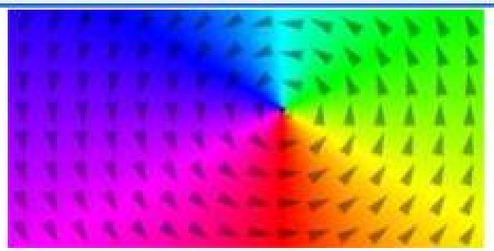


```
m=Vortex(1,-1).Add(0.1,randomMag())
```

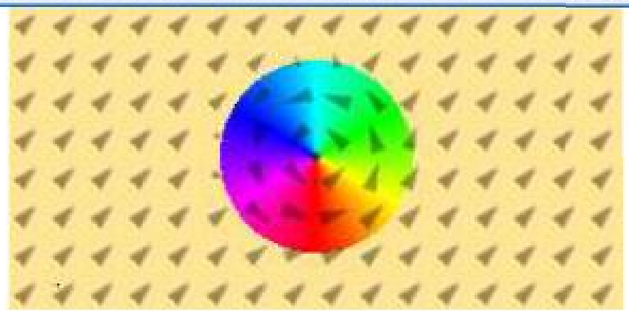


```
M = BlochSkyrmion(1, 1)
```

```
m=Vortex(1,-1).transl(100e-9,50e-9,0)
```



```
m = uniform(1, 1, 1)
m.setInShape(cylinder(400e-9,100e-9), vortex(1,-1))
```



GHENT UNIVERSITY

Jonathan.Leliaert@ugent.be

OUTPUT

3 output media:

- log file for input, logging and printing
- table.txt (t, mx, my, mz, …)

```
tableadd(E_total)
tableaddvar(myVar,”myVar”,”unit”)
tablesave() // write single line
tableautosave(1e-12) // write periodically
```

- .ovf files for scalar and vector fields

```
save(Edens_total)
saveas(Edens_total,”edens.ovf”)
autosave(Edens_total, 1e-10) // write perdiodically
```

RUN/RELAX/MINIMIZE

- Solving the LLG equation (time integration)

```
run(timeperiod)
steps(100)
runWhile(condition)
```

- Minimizing the energy

```
relax()    // LLG without precession
Minimize() // steepest descent[1]
```

THE INTERACTIONS

Effective field terms

- Demagnetization
- Exchange
- Anisotropy
- Dzyaloshinskii-Moriya
- External field
- Thermal field
- Custom field

Spin transfer torques

- Zhang-Li STT
- Slonczewski STT

DEMAGNETIZATION FIELD TERM

Demagnetization energy density

$$\varepsilon = -\frac{\mu_0}{2} M_s \boldsymbol{m} \cdot \boldsymbol{H}_{demag}$$

Regional Material Parameters	
`Msat`	Saturation magnetization (A/m)
`NoDemagSpins`	Disable magnetostatic interaction per region (default=0, set to 1 to disable).

Output Quantities	
`B_demag`	Magnetostatic field (T)
`Edens_demag`	Magnetostatic energy density (J/m3)
`E_demag`	Magnetostatic energy (J)

Other functionalities	
`EnableDemag`	Enables/disables demag (default=true)
`SetPBC`	Sets the number of repetitions in X,Y,Z to create periodic boundary conditions. The number of repetitions determines the cutoff range for the demagnetization.
`DemagAccuracy`	Controls accuracy of demag kernel

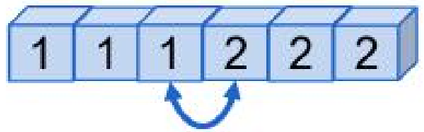
EXCHANGE FIELD TERM

Exchange energy density

$$\varepsilon = \mathrm{A}\,(\nabla \boldsymbol{m})^2$$

Harmonic mean for inter-region exchange coupling (default behavior)

$$\frac{A}{M_s} = 2\frac{\frac{A_1}{M_{s_1}}\frac{A_2}{M_{s_2}}}{\frac{A_1}{M_{s_1}} + \frac{A_2}{M_{s_2}}}$$



Regional Material Parameters	
Aex	Exchange stiffness (J/m)
Msat	Saturation magnetization (A/m)

Output Quantities	
B_exch	Exchange field (T)
Edens_exch	Total exchange energy density (including DMI) (J/m3)
E_exch	Total exchange energy (including DMI) (J)
MaxAngle	Maximum angle between exchanged coupled spins (rad)

Other functionalities	
Ext_InterExchange	Sets exchange coupling between two regions
Ext_ScaleExchange	Re-scales exchange coupling between two regions

ANISOTROPY FIELD TERM

Uniaxial anisotropy energy density

$$\varepsilon = -K_1 (\hat{u} \cdot \boldsymbol{m})^2 - K_2 (\hat{u} \cdot \boldsymbol{m})^4$$

Similar expression for cubic anisotropy energy density [1]

Regional Material Parameters	
`anisU`	Uniaxial anisotropy direction
`Ku1, Ku2`	Uniaxial anisotropy constants (J/m3)
`AnisC1, AnisC2`	Cubic anisotropy directions
`Kc1, Kc2, Kc3`	Cubic anisotropy constants (J/m3)
`Msat`	Saturation magnetization (A/m)

Output Quantities	
`B_anis`	Anisotropy field (T)
`Edens_anis`	Total anisotropy energy density (including DMI) (J/m3)
`E_anis`	Total anisotropy energy (including DMI) (J)

DZYALOSHINSKII-MORIYA FIELD TERM

Interfacially-induced DMI energy density

$$\varepsilon = D\,[m_z(\nabla \cdot \boldsymbol{m}) - (\boldsymbol{m} \cdot \nabla)m_z]$$

Bulk DMI energy density

$$\varepsilon = D\,\boldsymbol{m} \cdot (\nabla \times \boldsymbol{m})$$

Note:

Only single DMI type allowed at once

Regional Material Parameters	
`Dind`	Interfacially-induced DMI strength (J/m2)
`Dbulk`	Bulk DMI strength (J/m2)
`Msat`	Saturation magnetization (A/m)

Output Quantities	
`B_exch`	Exchange field (including DMI) (T)
`Edens_exch`	Total exchange energy density (including DMI) (J/m3)
`E_exch`	Total exchange energy (including DMI) (J)

Other functionalities	
`Ext_InterDind`	Sets Dind coupling between two regions
`Ext_ScaleDind`	Re-scales Dind coupling between two regions
`OpenBC`	Use open boundary conditions (default=false, use Neumann BC). This setting is only relevant for DMI.

EXTERNAL FIELD TERM

Zeeman energy density

$$\varepsilon = -M_s \boldsymbol{m} \cdot \boldsymbol{B}_{ext}$$

Regional Material Parameters	
`Msat`	Saturation magnetization (A/m)
Excitation	
`B_ext`	Externally applied field(T)

Output Quantities	
`B_ext`	Externally applied field(T)
`Edens_zeeman`	Zeeman energy density (J/m3)
`E_zeeman`	Zeeman energy (J)

THERMAL FIELD TERM

$$\langle \boldsymbol{H}_{th}(\boldsymbol{r},t) \rangle = 0$$

$$\langle \boldsymbol{H}_{th}(\boldsymbol{r},t), \boldsymbol{H}_{th}(\boldsymbol{r}',t') \rangle = \frac{2k_B T\alpha}{M_s \gamma} \delta(\boldsymbol{r}-\boldsymbol{r}')\delta(t-t')$$

Regional Material Parameters	
`Temp`	Temperature (K)
`alpha`	Damping parameter
`Msat`	Saturation magnetization (A/m)

Output Quantities	
`B_therm`	Thermal field (T)
`Edens_therm`	Thermal energy density (J/m3)
`E_therm`	Thermal energy (J)

Other functionalities	
`ThermSeed`	Sets random seed for thermal noise

FACULTY
OF SCIENCES

A FIRST SIMULATION

GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

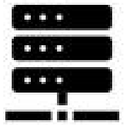
INTRODUCTION

HOW TO RUN MUMAX3

Windows/linux PC with NVIDIA GPU

Preferred option for small simulations and to learn about mumax3

Web GUI should work out of the box



Windows/linux server with NVIDIA GPUs

Preferred option for big simulations, or large batches of simulations

Setting up the Web GUI might be possible but requires some knowledge on network ports

Google collaboratory session

To fiddle around with mumax3 if you do not have access to an NVIDIA GPU

Free for gmail users, no Web GUI

https://colab.research.google.com/github/JeroenMulkers/mumax3-tutorial/blob/master/mumax3.ipynb

STANDARD PROBLEM 4

- https://www.ctcms.nist.gov/~rdm/mumag.org.html

  - Geometry

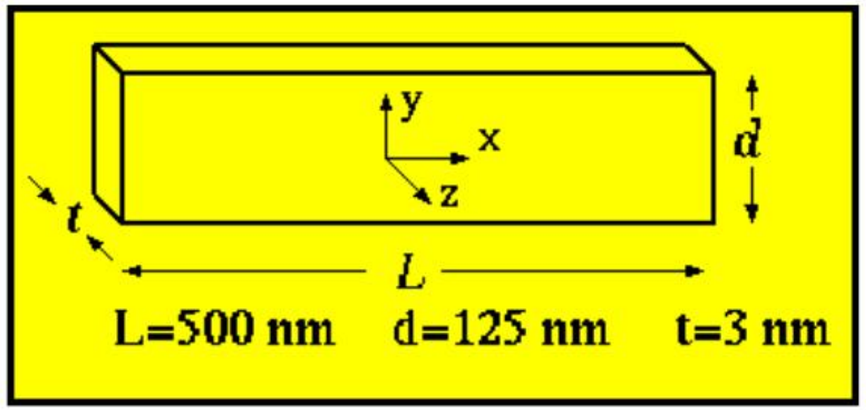
    A film of thickness, t=3 nm, length, L=500 nm and width, d=125 nm will be used.

  - Material parameters

    A = 1.3e-11 J/m
    M_s = 8.0e5 A/m
    } exchange length: 5.68 nm

    K = 0.0

    α = 0.02



  - Field

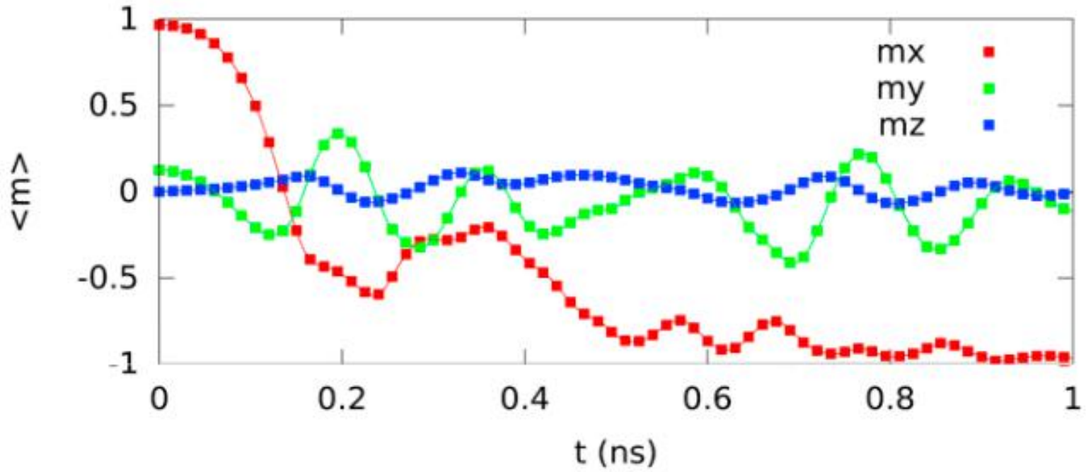
    $\mu_0 H_x$=-24.6 mT, $\mu_0 H_y$= 4.3 mT, $\mu_0 H_z$= 0.0 mT

STANDARD PROBLEM 4

- Input file

```
SetGridsize(128, 32, 1)
SetCellsize(500e-9/128, 125e-9/32, 3e-9)

Msat  = 800e3
Aex   = 13e-12
alpha = 0.02

m = uniform(1, .1, 0)
relax()
save(m)    // relaxed state

autosave(m, 200e-12)
tableautosave(10e-12)

B_ext = vector(-24.6E-3, 4.3E-3, 0)
run(1e-9)
```

- Output table

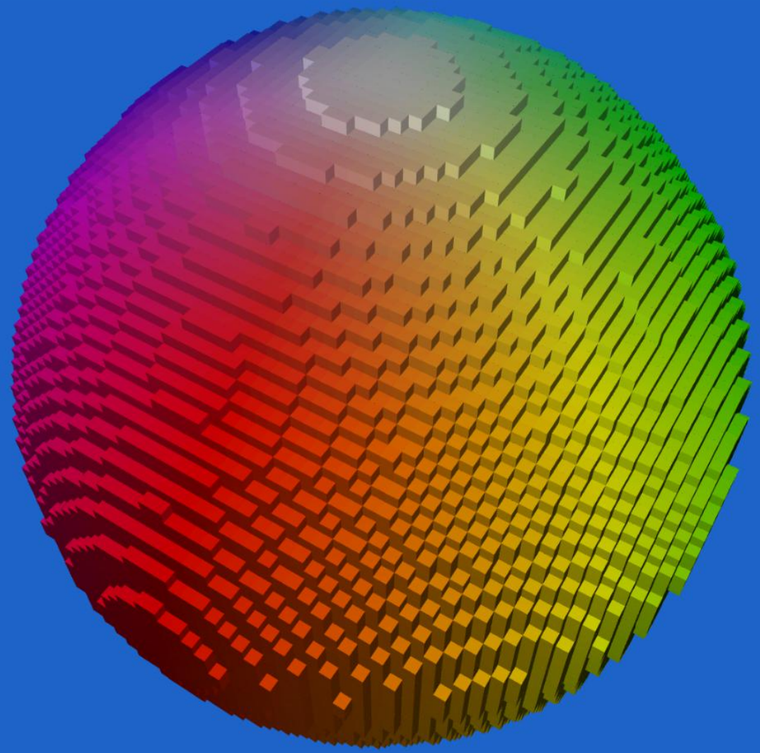


GOOGLE COLLABORATORY

- If you do not have access to a machine which has a CUDA enabled graphics card, you can still try out the examples for yourselves, by running mumax3 in a google collaboratory session for free
- The only thing you will need to get this working is a gmail account
- https://colab.research.google.com/github/JeroenMulkers/mumax3-tutorial/blob/master/mumax3.ipynb

FACULTY
OF SCIENCES

GROUND STATE OF IRON-OXIDE NANOPARTICLE



GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= vortex(1,1)

tableadd(E_total)
tableaddvar(D,”D”,”nm”)

for D=Dmax; D>=Dmin; D-=Dstep{
       setcellsize(D/N,D/N,D/N)
       minimize()
       tablesave()
}
```

Setting the grid size

```
N := 64
setgridsize(N,N,N)
```

GHENT UNIVERSITY

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= uniform(0,0,1)

tableadd(E_total)
tableaddvar(D,”D”,”nm”)

for D=Dmin; D<=Dmax; D+=Dstep{
      setcellsize(D/N,D/N,D/N)
      minimize()
      tablesave()
}
```

Typical material parameters for iron-oxide

```
Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)
```

Exchange length ~ 10 nm

GHENT UNIVERSITY

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= vortex(1,1)

tableadd(E_total)
tableaddvar(D,"D","nm")

for D=Dmax; D>=Dmin; D-=Dstep{
      setcellsize(D/N,D/N,D/N)
      minimize()
      tablesave()
}
```

Defining the particle diameter range

```
Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)
```

Our cell size ranges from 0.0625 to 0.25 l_{ex}

GHENT UNIVERSITY

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= vortex(1,1)

tableadd(E_total)
tableaddvar(D,"D","nm")

for D=Dmax; D>=Dmin; D-=Dstep{
      setcellsize(D/N,D/N,D/N)
      minimize()
      tablesave()
}
```

Initialize with a vortex state

```
m= vortex(1,1)
```

Alternatively, we could have used a uniform state

```
m= uniform(0,0,1)
```

GHENT UNIVERSITY

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= vortex(1,1)

tableadd(E_total)
tableaddvar(D,”D”,”nm”)

for D=Dmax; D>=Dmin; D-=Dstep{
      setcellsize(D/N,D/N,D/N)
      minimize()
      tablesave()
}
```

Scheduling the output

```
tableadd(E_total)
tableaddvar(D,”D”,”nm”)
```

GHENT UNIVERSITY

PARTICLE GROUND STATE

```
N := 64
setgridsize(N,N,N)

Msat = 400e3
Aex = 10e-12
Ku1 = 1e4
anisU = vector(0,0,1)

Dmin := 40e-9
Dmax := 160e-9
Dstep := 5e-9

D := Dmax
setcellsize(D/N,D/N,D/N)

m= vortex(1,1)

tableadd(E_total)
tableaddvar(D,”D”,”nm”)

for D=Dmax; D>=Dmin; D-=Dstep{
      setcellsize(D/N,D/N,D/N)
      minimize()
      tablesave()
}
```

sweeping the size

```
for D=Dmax; D>=Dmin; D-=Dstep{
     setcellsize(D/N,D/N,D/N)
     minimize()
     tablesave()
}
```

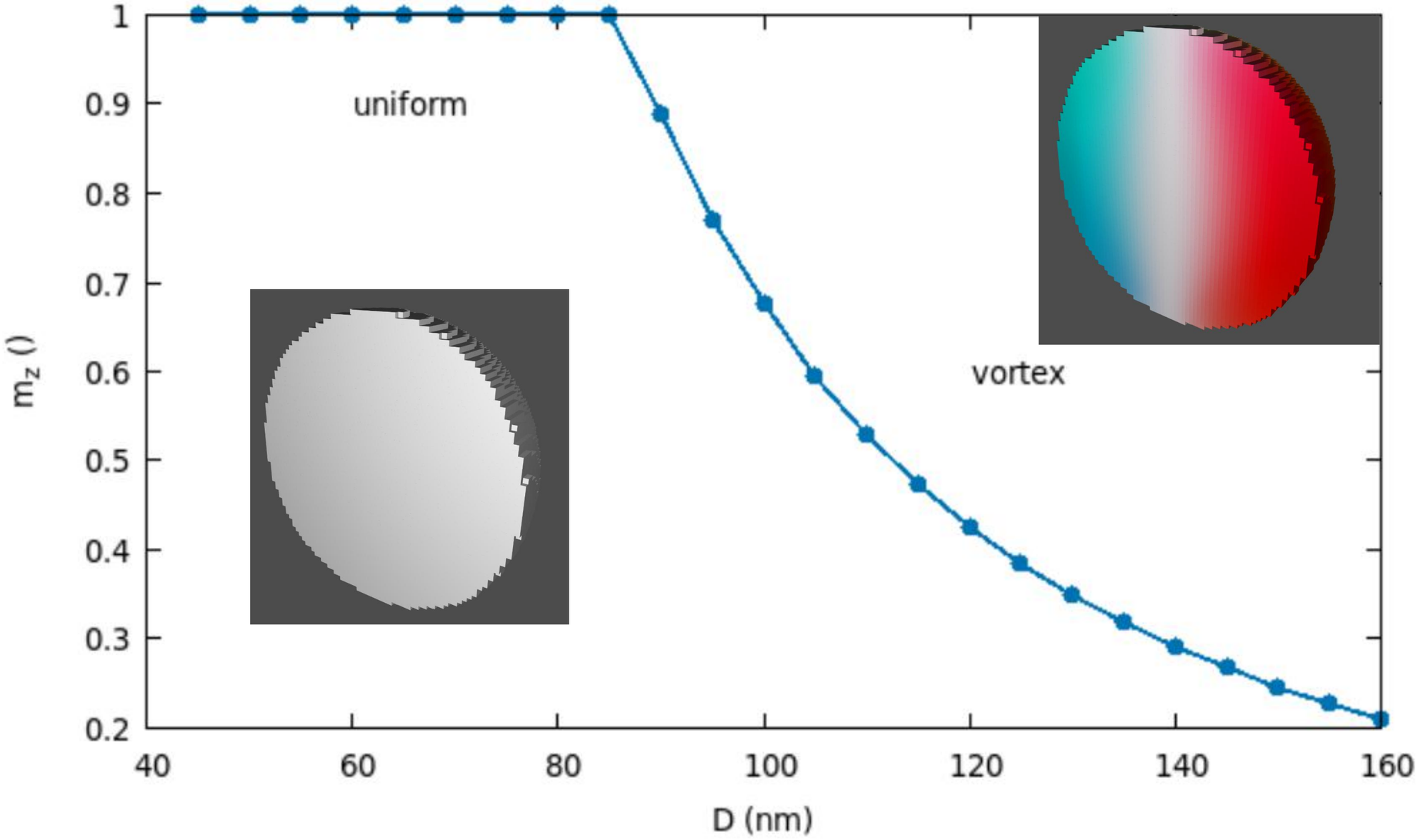
GHENT UNIVERSITY

PARTICLE GROUND STATE

# t (s)	mx ()	my ()	mz ()	E_total (J)	D (m)
0	2,37E-04	9,08E-05	0.99999726	7,88E-12	4,00E-08
0	-0.001503764	0.0004155231	0.99999434	1,12E-11	4.5e-08
0	-0.00025198163	0.00027258066	0.9999936	1,54E-11	5,00E-08
0	-0.00032723788	0.00035623793	0.9999911	2,05E-12	5.5e-08
0	0.0001287238	-0.00036605453	0.99998826	2,66E-11	6,00E-08
0	-1,71E+02	-0.0002854268	0.99998486	3,38E-11	6.5e-08
0	3,32E+01	0.00019878031	0.9999812	4,22E-12	7,00E-08
0	3,22E+01	1,03E+01	0.99997705	5,19E-12	7.5e-08
0	1,94E-03	4,04E-02	0.9999721	6,30E-12	8,00E-08
0	1,51E+02	-3,66E+01	0.9999653	7,56E-12	8.5e-08
0	-3,32E+01	5,31E+00	0.88785946	8,89E-12	9,00E-08
0	-3,84E+02	-0.0008873366	0.7715161	1,01E-12	9.5e-08
0	-0.0017875922	-0.003912248	0.6751741	1,13E-11	1,00E-07
0	-0.000107000684	-0.3572311	0.0006567935	1,16E-10	1.05e-07

GHENT UNIVERSITY

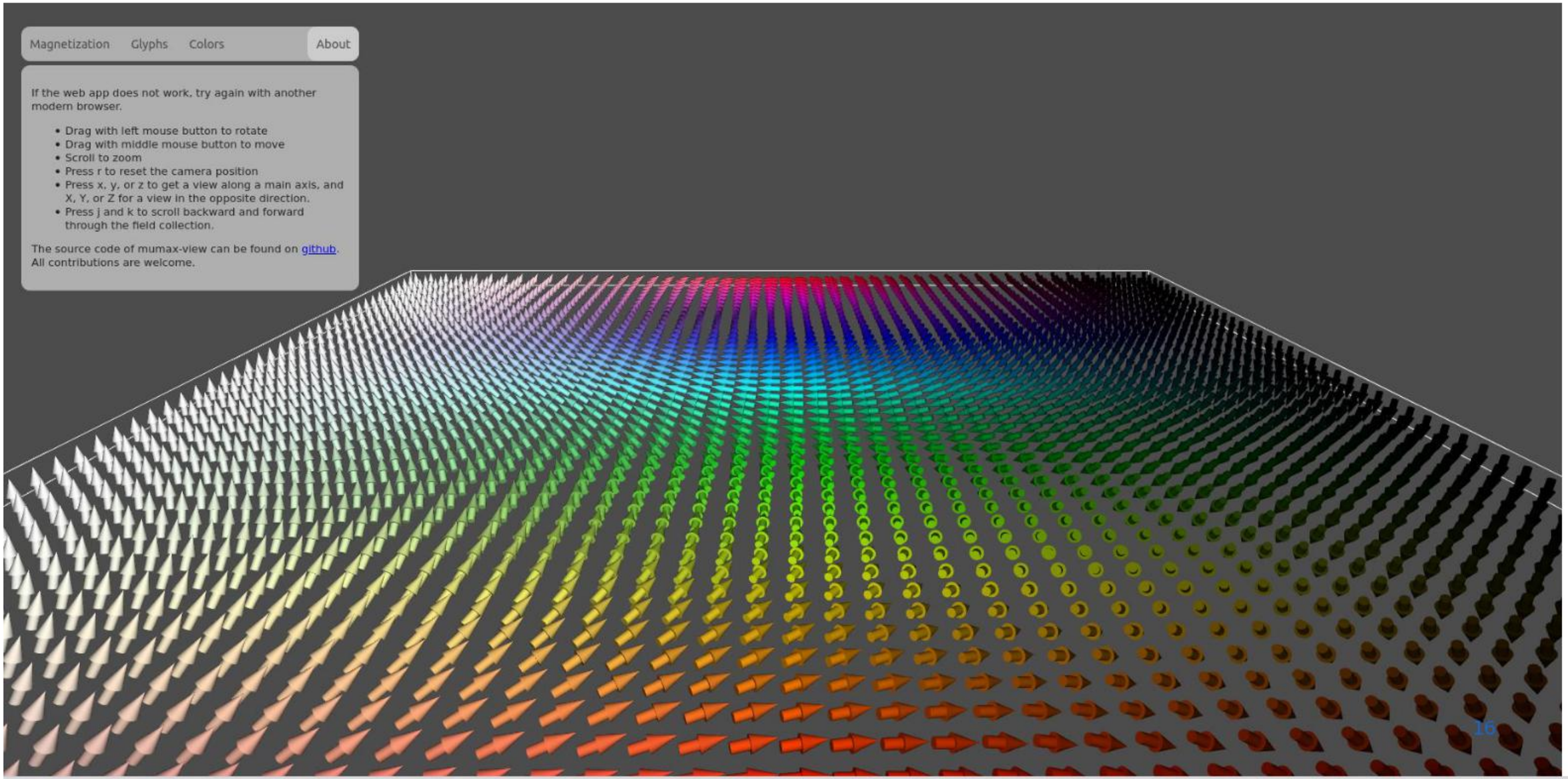
PARTICLE GROUND STATE



GHENT UNIVERSITY

MUMAX-VIEW

www.mumax.ugent.be/mumax-view

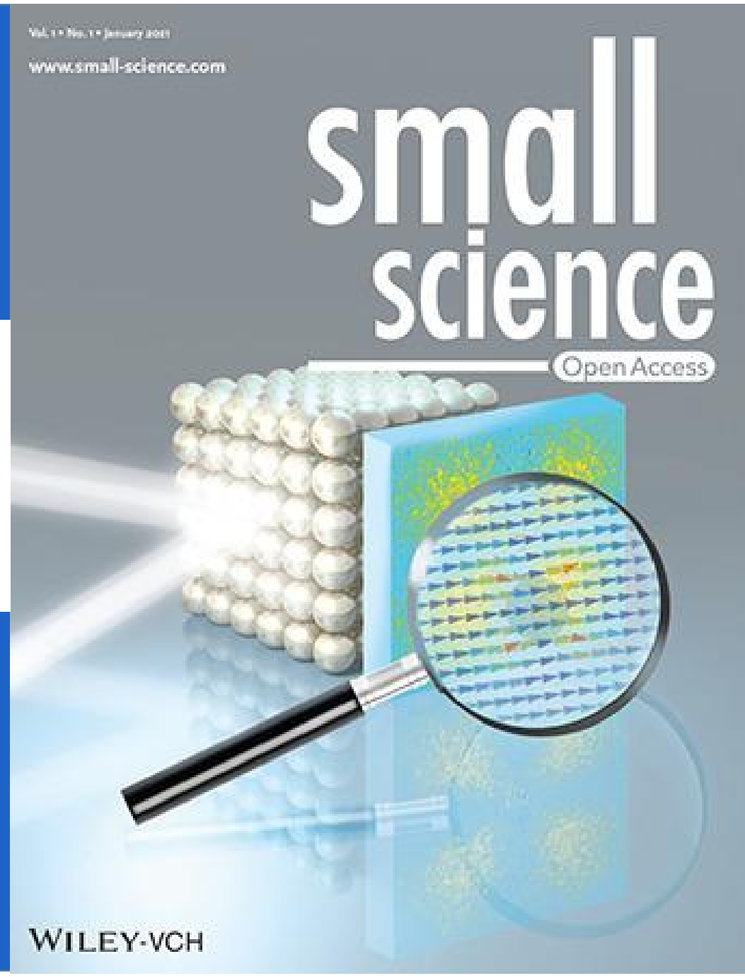


FACULTY
OF SCIENCES

Unraveling Nanostructured Spin Textures in Bulk Magnets

Philipp Bender ✉, Jonathan Leliaert, Mathias Bersweiler, Dirk Honecker, Andreas Michels

First published: 13 August 2020 | https://doi.org/10.1002/smsc.202000003



GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

SKYRMION RACETRACK

Skyrmion driven by spin transfer torques (STT) in a chiral ferromagnetic strip [1]

System: Co/Pt strip

- Ferromagnetic
- Perpendicular Magnetic Anisotropy (PMA)
- Interfacially-induced Dzyaloshinskii-Moriya interaction (iDMI)
- Uniform spin-polarized current along the length of the strip

[1] **Nucleation, stability and current-induced motion of isolated magnetic skyrmions in nanostructures**
J. Sampaio, V. Cros, S. Rohart, A. Thiaville, A. Fert
Nature Nanotechnology **volume 8**, pages 839–844 (2013)

GHENT UNIVERSITY

Jonathan.Leliaert@ugent.be

SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0) // A/m2

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

Setting the mesh

```
setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)
```

Note on the cell size:

In chiral ferromagnets, the magnetization might vary on a much smaller length scale than the exchange length!

When choosing the cell size make sure that maxAngle is small enough during the simulation (let's say <0.35rad)

In this example `print(maxAngle)` returns 0.26rad → cell size OK

GHENT UNIVERSITY

SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0) // A/m2

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

The material parameters[1]

```
Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1
```

GHENT UNIVERSITY

[1] **Nucleation, stability and current-induced motion of isolated magnetic skyrmions in nanostructures**
J. Sampaio, V. Cros, S. Rohart, A. Thiaville, A. Fert
Nature Nanotechnology **volume 8**, pages 839–844 (2013)

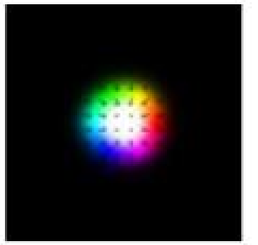
SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0) // A/m2

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

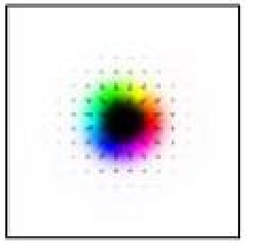
The initial state:

```
m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()
```

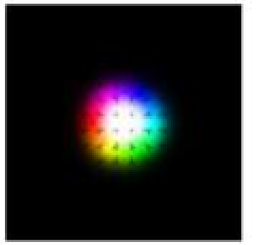
Neelskyrmion(1,1)



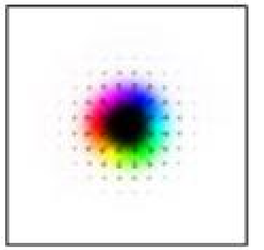
Neelskyrmion(1,-1)



Neelskyrmion(-1,1)



Neelskyrmion(-1,-1)



GHENT UNIVERSITY

SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0)

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

Applying a spin polarized current:

```
Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0)
```

GHENT UNIVERSITY

SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0)

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

Output scheduling:

```
autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)
```

GHENT
UNIVERSITY

SKYRMION RACETRACK

```
// Skyrmion racetrack

setgridsize(256,64,1)
setcellsize(1e-9,1e-9,1e-9)
setpbc(4,0,0)

Msat = 580e3
Aex = 15e-12
Dind = 3.0e-3
Ku1 = 0.8e6
AnisU = vector(0,0,1)
alpha = 0.1

m = neelskyrmion(-1, 1).transl(-40e-9,0,0)
minimize()

Pol = 0.4
xi = 0.2
j = vector(-1e12,0,0)

autosave(m,4e-10)
tableAutosave(1e-11)
tableAdd(ext_bubblepos)

run(10e-9)
```

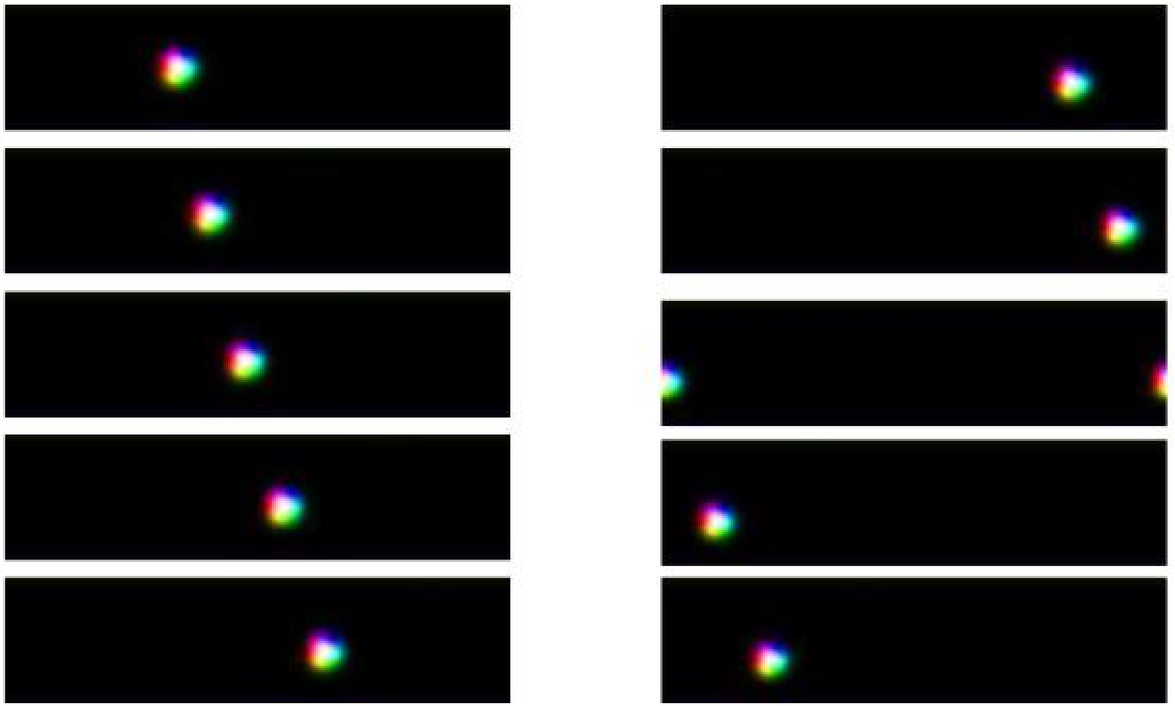
Solving the LLG equation

```
Run(10e-9)
```

GHENT
UNIVERSITY

SKYRMION RACETRACK

```
mumax3-convert -png skyrmionracetrack.out/*.ovf
```

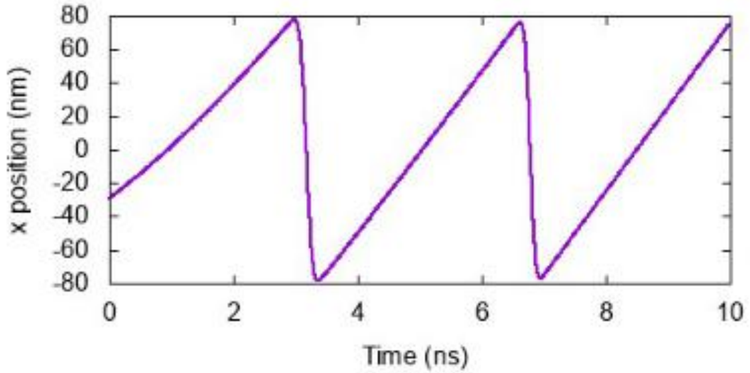


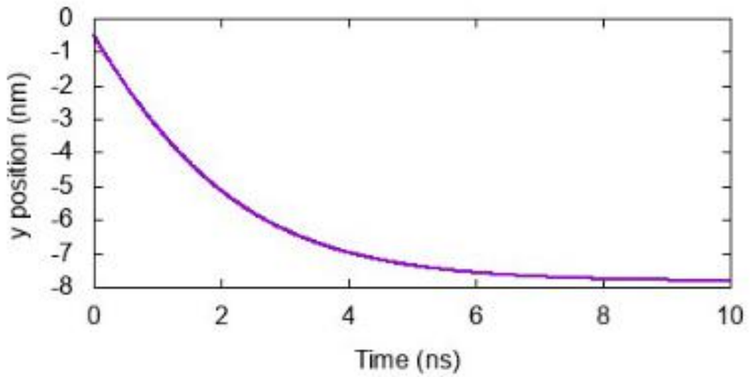
GHENT
UNIVERSITY

SKYRMION RACETRACK

table.txt

# t (s)	mx ()	my ()	mz ()	ext_bubbleposx (m)	ext_bubbleposy (m)	ext_bubbleposz (m)
0	1.46E-11	1.86E-09	-0.96588	-2.90E-08	-4.99E-10	0
1.00E-11	-0.00011191	-2.78E-05	-0.96588	-2.88E-08	-5.31E-10	0
2.00E-11	-1.95E-05	5.34E-06	-0.96587	-2.85E-08	-5.61E-10	0
3.00E-11	-9.36E-05	-1.93E-05	-0.96587	-2.82E-08	-5.93E-10	0
4.00E-11	-3.82E-05	-7.81E-06	-0.96587	-2.79E-08	-6.25E-10	0
5.00E-11	-7.69E-05	-1.55E-05	-0.96587	-2.76E-08	-6.56E-10	0
6.00E-11	-4.99E-05	-1.33E-05	-0.96587	-2.74E-08	-6.88E-10	0
7.00E-11	-6.81E-05	-1.61E-05	-0.96588	-2.71E-08	-7.19E-10	0
8.00E-11	-5.55E-05	-1.68E-05	-0.96588	-2.68E-08	-7.50E-10	0
9.00E-11	-6.35E-05	-1.79E-05	-0.96588	-2.65E-08	-7.82E-10	0
1.00E-10	-5.81E-05	-1.91E-05	-0.96588	-2.62E-08	-8.13E-10	0
1.10E-10	-6.13E-05	-2.03E-05	-0.96588	-2.60E-08	-8.44E-10	0
1.20E-10	-5.89E-05	-2.13E-05	-0.96588	-2.57E-08	-8.74E-10	0
1.30E-10	-6.03E-05	-2.26E-05	-0.96588	-2.54E-08	-9.05E-10	0
1.40E-10	-5.90E-05	-2.36E-05	-0.96589	-2.51E-08	-9.36E-10	0
1.50E-10	-5.96E-05	-2.49E-05	-0.96589	-2.48E-08	-9.67E-10	0
1.60E-10	-5.89E-05	-2.59E-05	-0.96589	-2.45E-08	-9.97E-10	0
1.70E-10	-5.91E-05	-2.71E-05	-0.96589	-2.42E-08	-1.03E-09	0
1.80E-10	-5.87E-05	-2.82E-05	-0.96589	-2.39E-08	-1.06E-09	0
1.90E-10	-5.87E-05	-2.94E-05	-0.9659	-2.37E-08	-1.09E-09	0
2.00E-10	-5.84E-05	-3.05E-05	-0.9659	-2.34E-08	-1.12E-09	0





GHENT UNIVERSITY

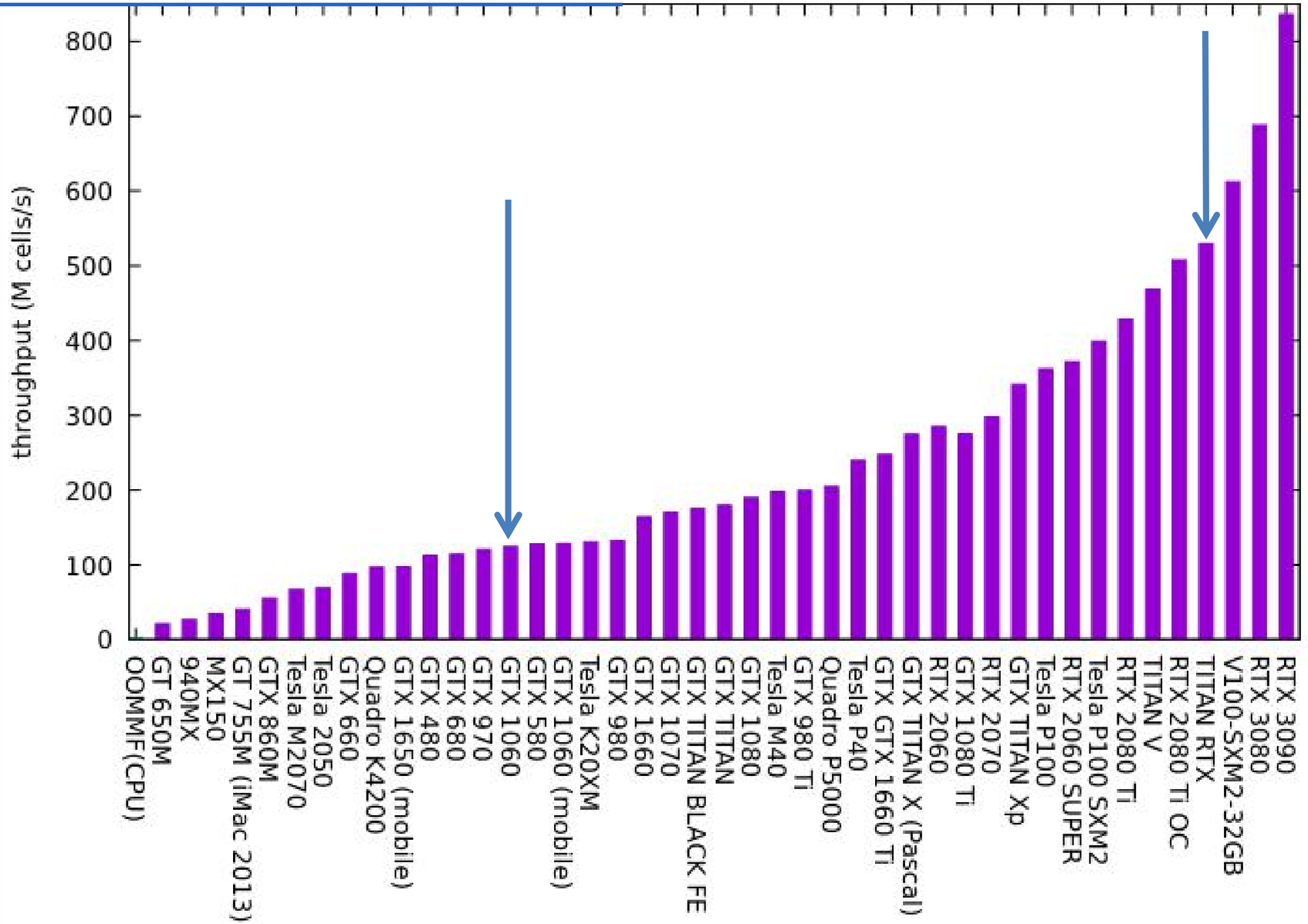
FACULTY
OF SCIENCES

HARDWARE

GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

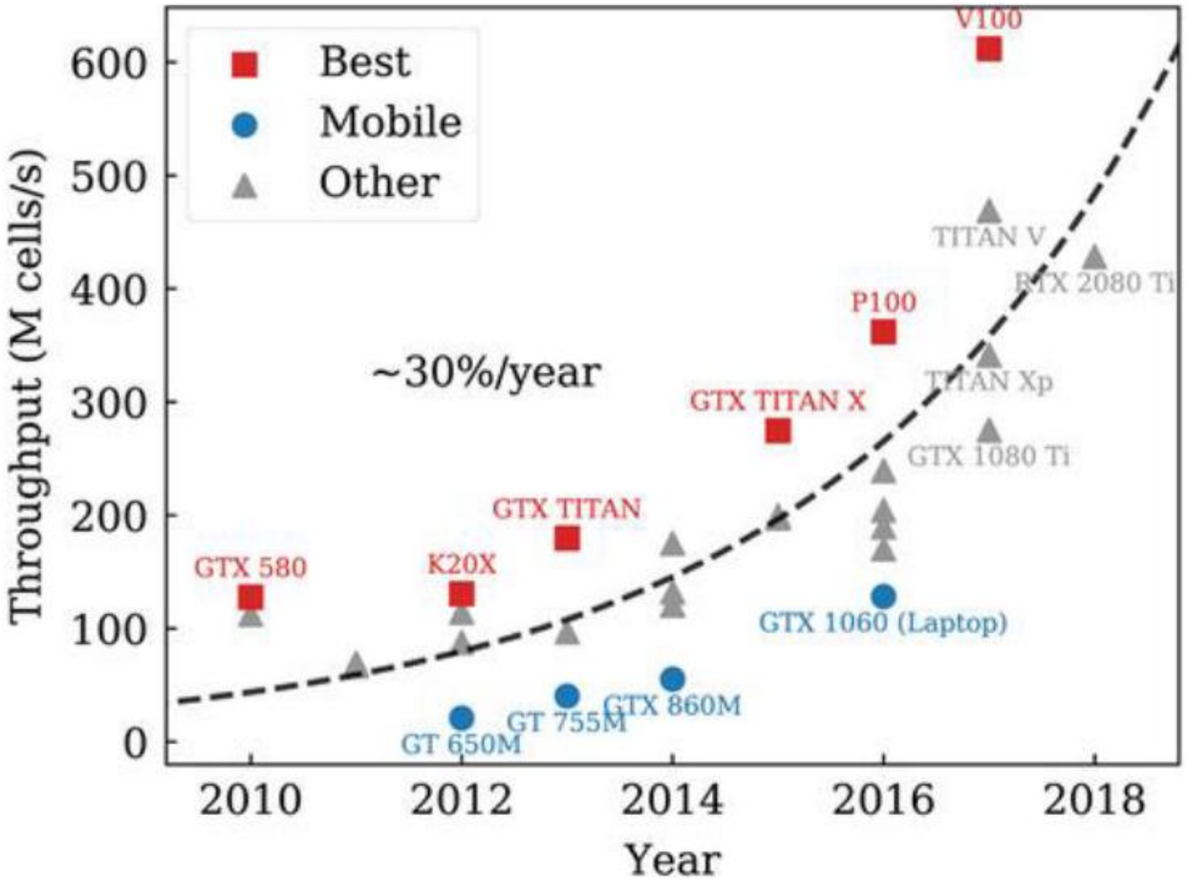
MUMAX3 BENCHMARKS



GHENT UNIVERSITY

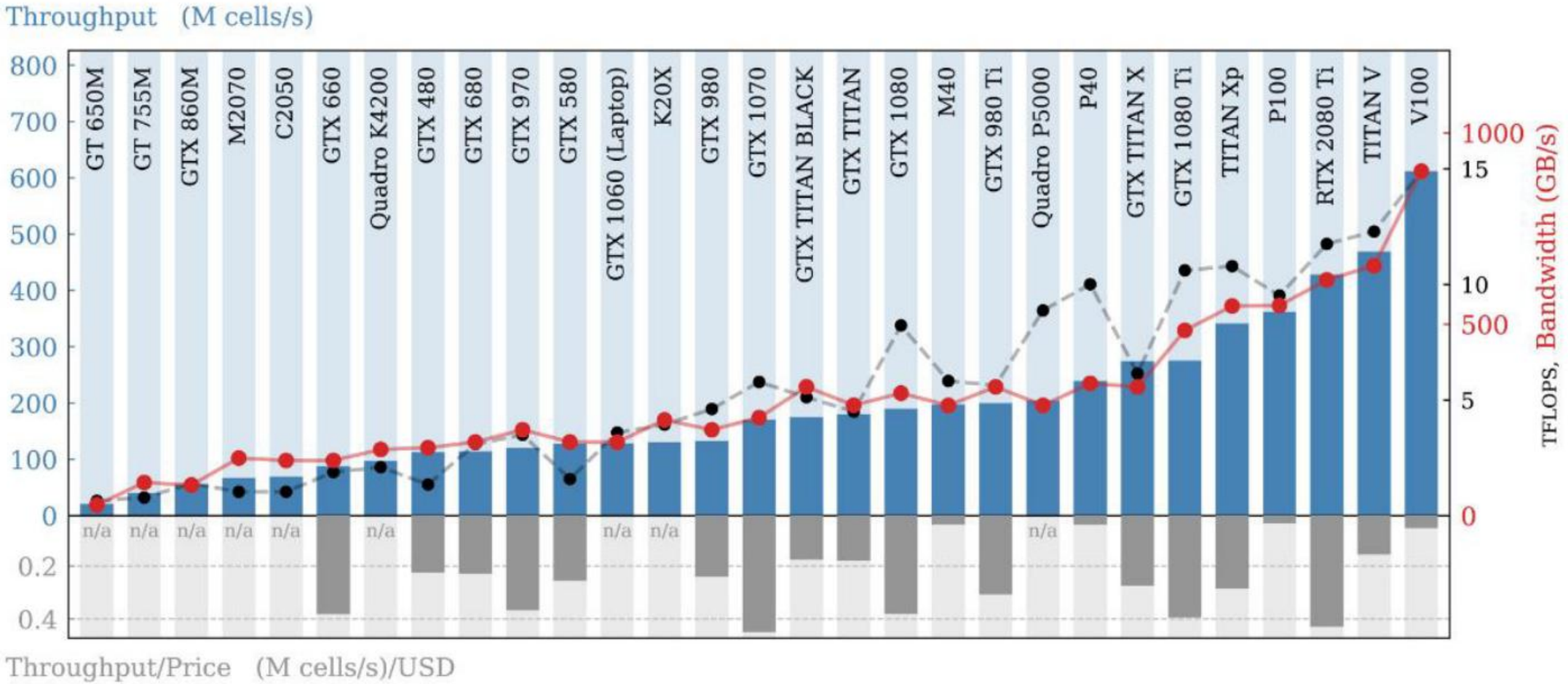
Jonathan.Leliaert@ugent.be

MUMAX3 BENCHMARKS



Tomorrow's micromagnetic simulations, Journal of Applied Physics 125, 180901 (2019)
https://doi.org/10.1063/1.5093730

MUMAX3 BENCHMARKS



Tomorrow's micromagnetic simulations, Journal of Applied Physics 125, 180901 (2019)
https://doi.org/10.1063/1.5093730

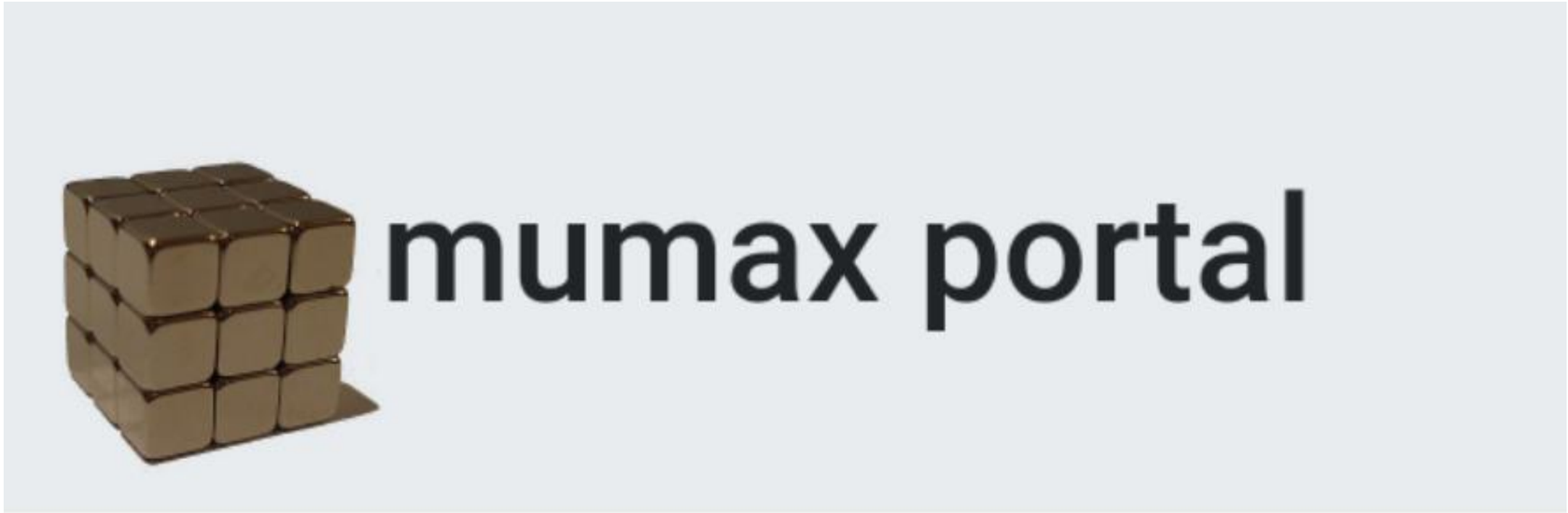
FACULTY
OF SCIENCES

EXTRA RESOURCES

GHENT
UNIVERSITY

Jonathan.Leliaert@ugent.be

EXTRA RESOURCES https://mumax.ugent.be



- mumax3 website
- mumax3 workshop
- mumax-view
- mumax mailing list
- dynamat website

EXTRA RESOURCES

Publications:

- The design and verification of mumax3
  https://doi.org/10.1063/1.4899186
- Fast micromagnetic simulations on GPU - recent advances made with mumax3
  https://doi.org/10.1088/1361-6463/aaab1c
- Tomorrow's micromagnetic simulations
  https://doi.org/10.1063/1.5093730
- Adaptively time stepping the stochastic Landau-Lifshitz-Gilbert equation at nonzero temperature: Implementation and validation in MuMax3
  https://doi.org/10.1063/1.5003957
- A numerical approach to incorporate intrinsic material defects in micromagnetic simulations
  https://doi.org/10.1063/1.4854956

FACULTY OF SCIENCES

Thank you for your interest!

GHENT UNIVERSITY

GHENT UNIVERSITY

Jonathan.Leliaert@ugent.be