OOMMF Tutorial
Part III: Advanced Simulations

Michael J. Donahue

Applied and Computational Mathematics Division
National Institute of Standards and Technology
Gaithersburg, Maryland

2-June-2020

Special Thanks to

Online Spintronics Seminar

Professor Xin Fan (Univ. Denver)

Professor Kirill Belashchenko (Univ. Nebraska-Lincoln)

nanoHUB

Tanya Faltens (Purdue University)

IEEE Magnetics Society

DISCLAIMER: The mention of specific products, trademarks, or brand names is for purposes of identification only. Such mention is not to be interpreted in any way as an endorsement or certification of such products or brands by the National Institute of Standards and Technology. All trademarks mentioned herein belong to their respective owners.

OOMMF Tutorial
Part III: Advanced Simulations

Session schedule

- Thur, 21-May-2020: Intro to Micromagnetics
- Tues, 26-May-2020: OOMMF Basics
- Tues, 2-June-2020: Pitfalls, advanced MIF, writing an extension
- Tues, 9-June-2020: Data analysis, pics, movies, dispersion curves, . . .

All sessions start at 12:00 noon EDT.

OOMMF Tutorial

Part III: Advanced Simulations

Homework: Skyrmion initialization

```
Specify Oxs_CGEvolve {}

Specify Oxs_MinDriver [subst {
 evolver Oxs_CGEvolve
 stopping_mxHxm 1e-5
 mesh :mesh
 Ms $Ms
 m0 { Oxs_ScriptVectorField {
   script Skyrmion
   atlas :atlas
   script_args rawpt
 }}
}]
```

Skyrmion initialization

```
proc Skyrmion { x y z } {
 global skyrmion_x skyrmion_y
 global skyrmion_rsq_inner skyrmion_rsq_outer
 set xoff [expr {$skyrmion_x-$x}]
 set yoff [expr {$skyrmion_y-$y}]
 set rsq [expr {$xoff*$xoff+$yoff*$yoff}]
 if {$rsq<$skyrmion_rsq_inner} {return {0. 0.  1.}}
 if {$rsq>$skyrmion_rsq_outer} {return {0. 0. -1.}}
 return [list $xoff $yoff 0]
}
```

Exchange and DMI

```
Specify Oxs_UniformExchange:HeisenbergEx {
  A 1.6e-11
}

#uniform DMI is used here
Specify Oxs_DMExchange6Ngbr:DMEx [subst {
  default_D $DD
  atlas :atlas
  D {
    world world $DD
  }
}]
```

No demag! (Sorry. . .)

Anisotropy and pinning

```
set divot_r [expr {4*$xcell}]

Specify Oxs_MultiAtlas:atlas [subst {
 atlas { Oxs_BoxAtlas:divot {
 xrange { [expr {$skyrmion_r-$divot_r}]
          [expr {$skyrmion_r+$divot_r}] }
 yrange { [expr {$skyrmion_r-$divot_r}]
          [expr {$skyrmion_r+$divot_r}] }
  zrange { 0 $film_thickness }
 }}

 atlas { Oxs_BoxAtlas:world {
  xrange { 0 $xmax }
  yrange { 0 $ymax }
  zrange { 0 $film_thickness }
 }}
}]
```

Anisotropy and pinning

```
set K1 0.51e6
set K1_divot [expr {1.03*$K1}]

Specify Oxs_UniaxialAnisotropy [subst {
   axis {0 0 1}
   K1 { Oxs_AtlasScalarField {
      atlas :atlas
      default_value $K1
      values {
         divot $K1_divot
      }
   }}
}]
```

Homework 2

Using the equilibrium state from Homework 1 as the initial state, run a STT simulation using the **Anv_SpinTEvolve** extension with these parameters:

- u=100 m/s
- alpha=0.1
- beta=0.04

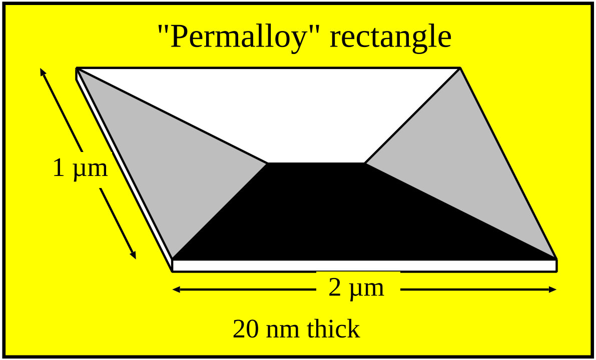
See the **Anv_SpinTEvolve** web page and sample problem to get started.

The skyrmion should move to the right, and slightly upward. Determine the speed of the skyrmion and the drift angle. Try varying alpha. For alpha = beta there should be no up or down drift. For alpha<beta the drift should be downward. For that condition flip the initial state using **Oxs_AffineOrientVectorField** and **Oxs_AffineTransformVectorField**.

OOMMF Tutorial

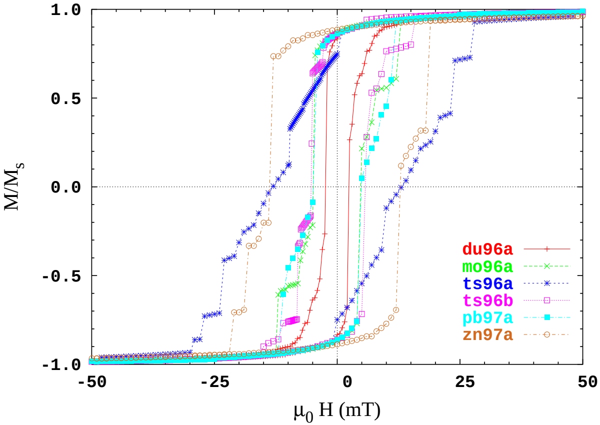
Part III: Advanced Simulations

μMAG standard problem 1



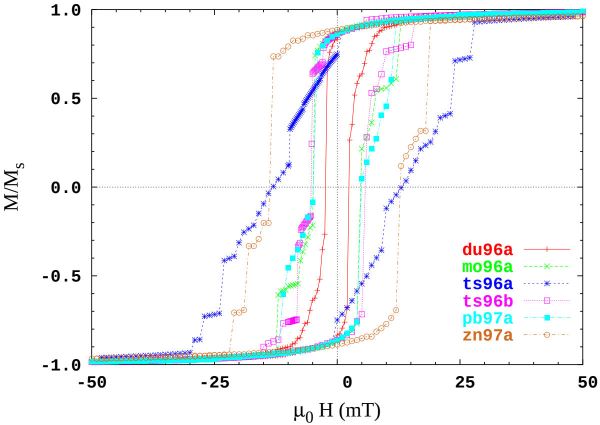
Task: Run this through a hysteresis loop.

μMAG standard problem 1: results



Problem: ???

μMAG standard problem 1: results



Problem: Meshes were too coarse!

Exchange lengths

Magnetocrystalline exchange length (for hard materials):

$$\ell_{\text{ex,K}} = \sqrt{\frac{A}{K_u}}$$

Magnetostatic exchange length (for soft materials):

$$\ell_{\text{ex,Ms}} = \sqrt{\frac{2A}{\mu_0 M_s^2}}$$

- Don't mesh any coarser than smaller of these two values!
- Don't confuse the latter with the "characteristic length"

$$R_0 = \sqrt{2\pi}\sqrt{\frac{2A}{\mu_0 M_s^2}} \approx 2.5\ell_{\text{ex,Ms}}$$

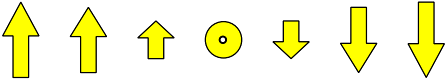
G.S. Abo, Y.-K. Hong et al., *IEEE Trans. Magn.*, **49**, 4937 (2013).

Example ℓ_{ex} values

Material	M_s (kA/m)	K (kJ/m^3)	A (pJ/m)	$\ell_{ex,K}$ (nm)	$\ell_{ex,Ms}$ (nm)
Fe	1700	48	21	21	3.4
Co	1400	520	30	7.6	4.9
Ni	490	-5.7	9	40	7.7
Permalloy	800	0	13	-	5.7
$Nd_2Fe_{14}B$	1280	4500	13	1.7	3.6

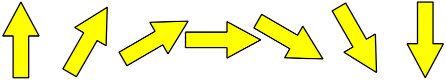
1D wall types

Bloch wall



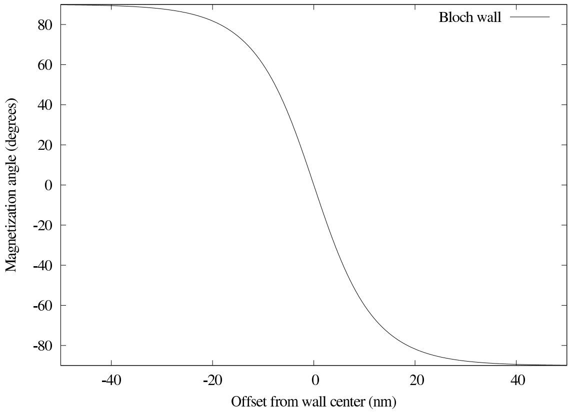
$$\nabla \cdot \boldsymbol{M} = 0 \quad \Rightarrow \quad \boldsymbol{H}_{\text{demag}} = 0$$

Néel wall

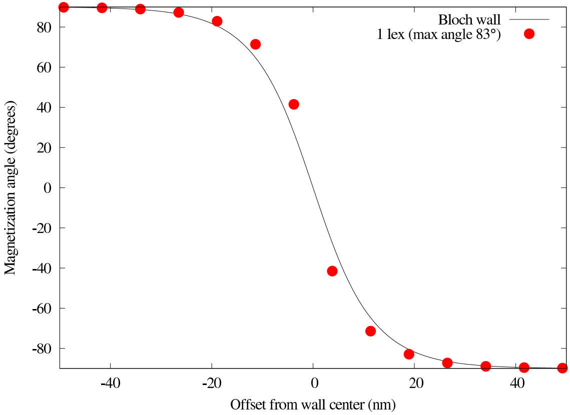


$$\nabla \cdot \boldsymbol{M} \neq 0 \quad \Rightarrow \quad \boldsymbol{H}_{\text{demag}} \neq 0$$

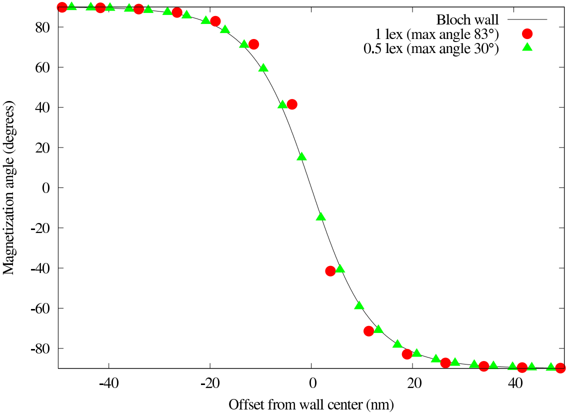
Bloch wall discretization



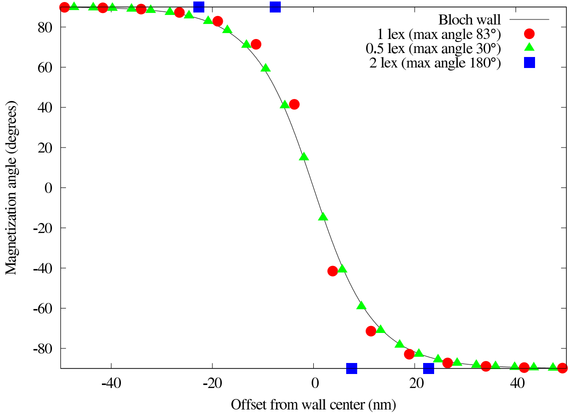
Bloch wall discretization



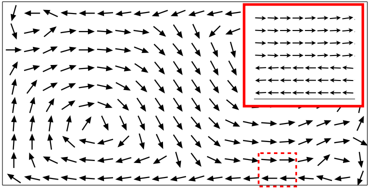
Bloch wall discretization



Bloch wall discretization



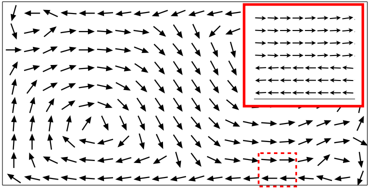
Néel wall collapse



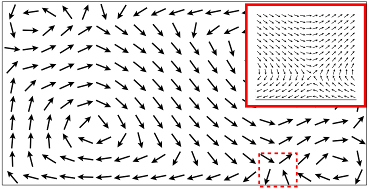
25 nm cells

MJ Donahue, “A variational approach to exchange energy calculations in micromagnetics,” *JAP*, **83**, 6491 (1998).

Néel wall collapse



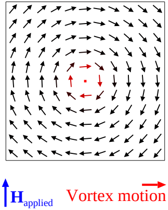
25 nm cells

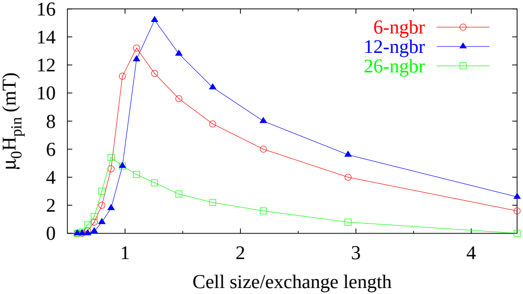


12.5 nm cells

MJ Donahue, "A variational approach to exchange energy calculations in micromagnetics," *JAP*, **83**, 6491 (1998).

Vortex mobility





MJ Donahue & RD McMichael, *Physica B*, **233**, 272 (1997).
MJ Donahue & DG Porter, *Physica B*, **343**, 177 (2004).

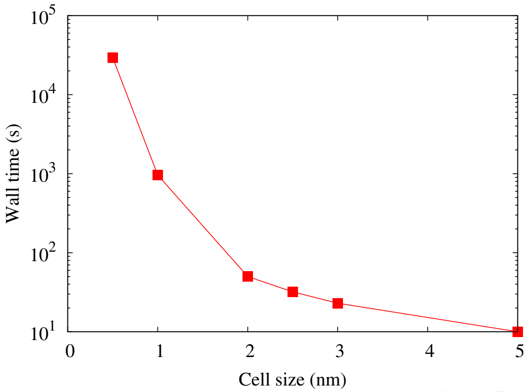
Cell size recommendations

- Don't mesh coarser than ℓ_{ex}
- Check max neighbor angle: under 30°is usually reliable, over 90°is questionable, 180°is bogus.
- Run at multiple discretizations and check for convergence (if possible!)

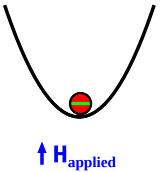
Over-mesh = too stiff

Standard Problem 4: Run time vs. cell size

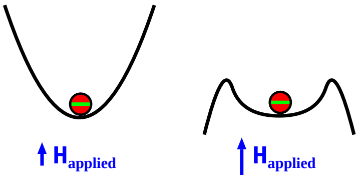
Cellsize (nm)	Cell count	Iterations	Wall time (s)	Max angle (deg)
5.0	2500	583	10	108.1
3.0	7014	1521	23	68.0
2.5	10000	2165	32	52.2
2.0	15750	3405	50	37.4
1.0	187500	18565	961	17.7
0.5	1500000	79191	29469	8.7



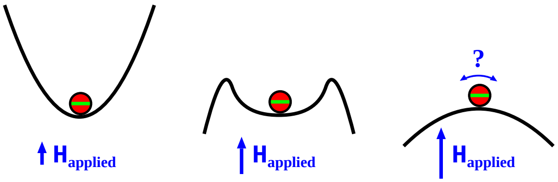
Symmetry breaking



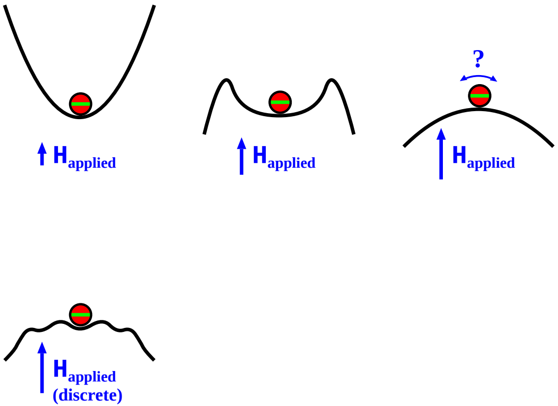
Symmetry breaking



Symmetry breaking



Symmetry breaking

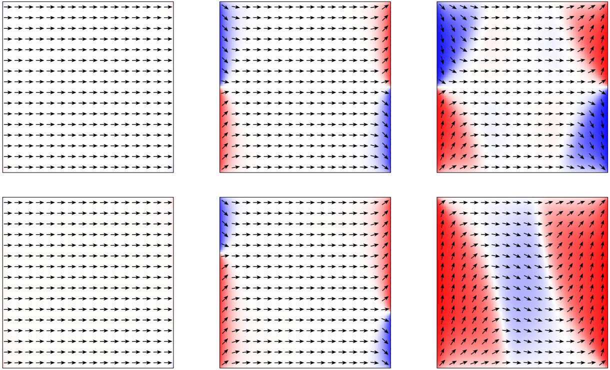


- Discretization introduces (false) divots
- Maximum replaced by saddle in higher dimensions

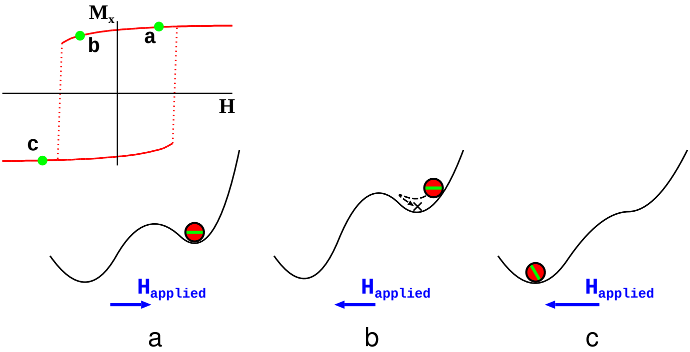
Symmetry breaking

H_{applied} on axis

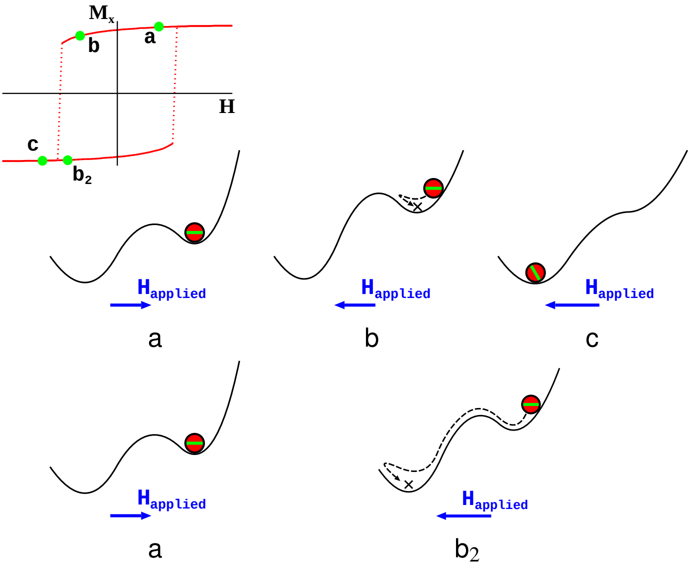
H_{applied} 1°off-axis



Big field steps ⇒ premature switching



Big field steps ⇒ premature switching

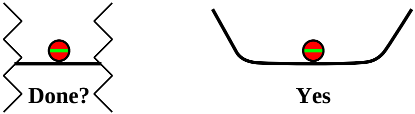


Stopping too soon

Stopping criteria: $\boldsymbol{M} \times \boldsymbol{H}_{\text{eff}} <$ stoptorque

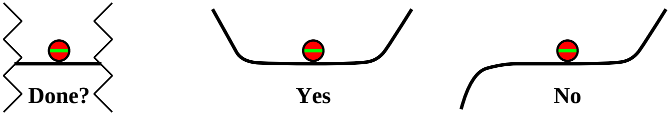
Stopping too soon

Stopping criteria: $\boldsymbol{M} \times \boldsymbol{H}_{\text{eff}} < \text{stoptorque}$



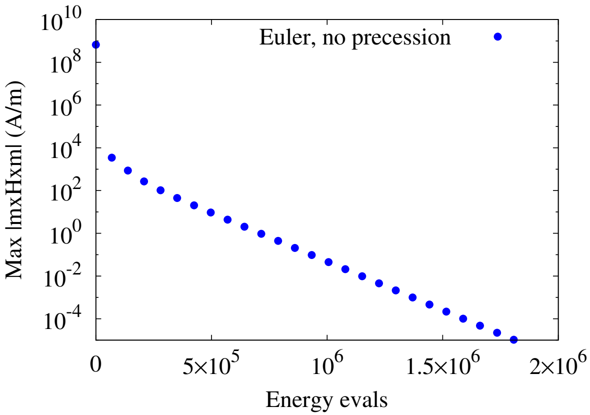
Stopping too soon

Stopping criteria: $\boldsymbol{M} \times \boldsymbol{H}_{\text{eff}} <$ stoptorque



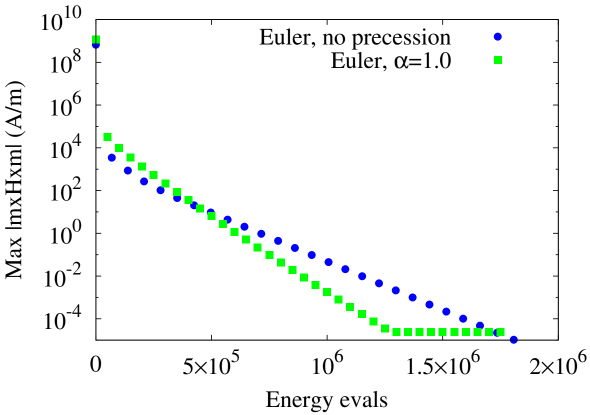
LLG vs. Conjugate Gradient

Standard problem 3 (energy minimization):



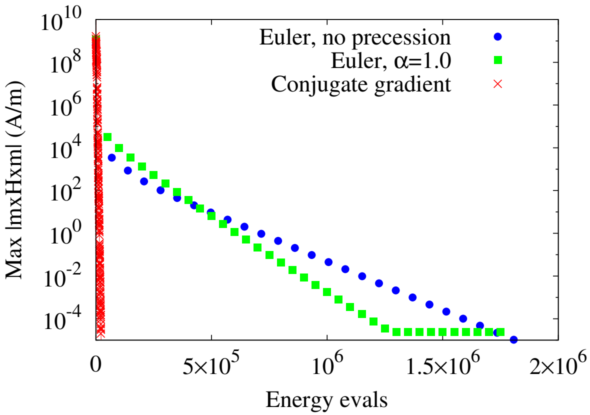
LLG vs. Conjugate Gradient

Standard problem 3 (energy minimization):



LLG vs. Conjugate Gradient

Standard problem 3 (energy minimization):



- Use the right tool for the job!

OOMMF Tutorial
Part III: Advanced Simulations

Editing MIF files

- MIF files can be edited in any plain-text editor.
- Source code editors (e.g. Notepad++, Geany, Emacs, vi) can ease the task with syntax highlighting. Use Tcl mode for MIF files.
- Select Tcl mode in Notepad++ via
  `Language | T | Tcl`
- Select Tcl mode in Geany via
  `Document | Set Filetype | Scripting Languages | Tcl source file`
- It is usually possible to configure your editor to automatically recognize .mif files as Tcl source code files.

OOMMF User's Guide (OUG)

The OUG provides information on:

- Graphical applications
  Controls, keyboard shortcuts, configuration files
- Command line applications
  Command line options, outputs
- File formats
  MIF, ODT, OVF
- Oxs_Ext child classes
  Atlases, energies, evolvers, . . .
  Specify options, examples

MIF 2.1 vs. 2.2

- MIF 2.1 files start with the line *# MIF 2.1*
- MIF 2.1 commands include

  Specify Parameter Ignore comment ReadFile Report Random RandomSeed Destination Schedule

- MIF 2.1 files are processed in two passes. Specify blocks are evaluated in during pass.

- MIF 2.2 files start with the line *# MIF 2.2*
- MIF 2.2 introduces additional commands, including

  SetOptions GetMifParameters EvalScalarField EvalVectorField GetAtlasRegions GetAtlasRegionByPosition

- MIF 2.2 files are processed in one pass $\Rightarrow$ **proc** definitions must precede use.

Log files

- Each run of Oxsii logs output to oommf/oxsii.errors
- Each run of Boxsi logs output to oommf/boxsi.errors
- Each log entry lists PID, machine, user, timestamp and message
- OOMMF never trims these files.
- The MIF **Report** command writes into log files.

Starting a simulation from a previous state

Say the name of the initial magnetization state file is m0_init.ovf. Load that as the initial state in the Driver block like so:

```
Specify Oxs_TimeDriver [subst {
  evolver :evolver
  stopping_dm_dt 1e-3
  mesh :mesh
  Ms $Ms
  m0 { Oxs_FileVectorField {
      file m0_init.ovf
      atlas Oxs_MultiAtlas:atlas
  } }
}]
```

Scheduled outputs

Outputs can be set up inside MIF files using the Destination and Schedule commands:

```
# Default outputs
Destination hystgraph mmGraph
Destination monitor   mmGraph   new
Destination archive   mmArchive

Schedule DataTable hystgraph Stage 1
Schedule DataTable monitor   Step 5
Schedule Oxs_TimeDriver::Magnetization archive Stage 1
```

These are documented in the MIF 2.1 chapter of the OUG.

Output basename (MIF 2.1)

The output basename determines the prefix for files saved via mmArchive. For MIF 2.1 use the driver Specify block:

```
set basename [format {multilayer-thick%04.1f} \
    [expr {$thickness*1e9}]]

Specify Oxs_TimeDriver [subst {
 basename $basename
 evolver Oxs_RungeKuttaEvolve
 stopping_dm_dt 0.01
 mesh :mesh
 Ms 8e5  comment {implicit Oxs_UniformScalarField}
 m0 {1 0 0} comment {implicit Oxs_UniformVectorField}
}]
```

Output basename (MIF 2.2)

In MIF 2.2 use the SetOptions command to set basename:

```
SetOptions [subst {
 basename $basename
 scalar_output_format %.12g
 scalar_field_output_format {text %.4g}
 scalar_field_output_meshtype irregular
 vector_field_output_format {binary 4}
}]
```

Stopping criteria

- Stage stopping options for **Oxs_MinDriver**:

```
stopping_mxHxm
stage_iteration_limit
total_iteration_limit
```

- Stage stopping options for **Oxs_TimeDriver**:

```
stopping_dm_dt
stopping_time
stage_iteration_limit
total_iteration_limit
```

Fixed spins

Use `fixed_spins` in evolver classes to mark pinned regions.

```
set frame_width 10e-9
Specify Oxs_MultiAtlas:atlas [subst {
 atlas { Oxs_BoxAtlas:interior {
  xrange { $frame_width [expr {$xmax-$frame_width}] }
  yrange { $frame_width [expr {$ymax-$frame_width}] }
  zrange { 0 $film_thickness }
 }}
 atlas { Oxs_BoxAtlas:frame {
  xrange { 0 $xmax }
  yrange { 0 $ymax }
  zrange { 0 $film_thickness }
 }}
}]
Specify Oxs_CGEvolve {
 fixed_spins {:atlas frame}
}
```

Periodic boundaries

Periodic boundaries are specified with the Oxs_PeriodicRectangularMesh class. That is, replace

```
Specify Oxs_RectangularMesh {
   cellsize { 5e-9 5e-9 4e-9 }
   atlas :atlas
}
```

with

```
Specify Oxs_PeriodicRectangularMesh {
   cellsize { 5e-9 5e-9 4e-9 }
   atlas :atlas
   periodic x
}
```

where the allowed values for periodic are x, y, or z.

OOMMF Tutorial

Part III: Advanced Simulations

Introduction to the OOMMF command line

- oommf.tcl help
- pidinfo, killoommf
- oxspkg
- pimake
- launchhost

Reference: “Command Line Utilities” section of the OUG.

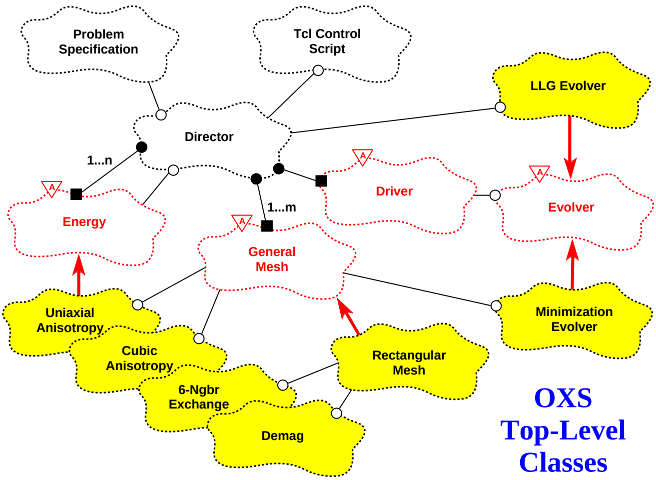
Batch OOMMF on clusters

- Use launchhost to protect against interference between sessions:

```
#!/bin/sh
OOMMF_HOSTPORT=`tclsh oommf.tcl launchhost 0`
export OOMMF_HOSTPORT
tclsh oommf.tcl mmArchive
tclsh oommf.tcl boxsi sample.mif
tclsh oommf.tcl killoommf all
```

OOMMF Tutorial
Part III: Advanced Simulations

OOMMF C++ class structure



Adding a new energy term to OOMMF

1. Copy sample `.h` and `.cc` files to `oommf/app/oxs/local`.
2. Change names.
3. Add new code.
4. Run `pimake`.
5. Add new term to `MIF` input file.

NB: Modify no files in OOMMF distribution!

See Oxs Extension Modules page

`https://math.nist.gov/oommf/contrib/oxsext/`

for examples.

Example extension: uniaxial anisotropy

Simple form: $E_{\text{anis}} = K_1 \sin^2 \phi$

Extended form: $E_{\text{anis}} = K_1 \sin^2 \phi + K_2 \sin^4 \phi$

where ϕ is angle between $\boldsymbol{m}$ and $\boldsymbol{u}$.

Sample anisotropy header file

```
-#ifndef _OXS_SIMPLEANISOTROPY)
-#define _OXS_SIMPLEANISOTROPY
+#ifndef _MY_ANISOTROPY
+#define _MY_ANISOTROPY

 #include "nb.h"
 ...
-class Oxs_SimpleAnisotropy:public Oxs_Energy {
+class My_ExtendedAnisotropy:public Oxs_Energy {
 private:
   Oxs_OwnedPointer<Oxs_ScalarField> K1_init;
+  Oxs_OwnedPointer<Oxs_ScalarField> K2_init;
   Oxs_OwnedPointer<Oxs_VectorField> axis_init;
   mutable OC_UINT4m mesh_id;
   mutable Oxs_MeshValue<OC_REAL8m> K1;
+  mutable Oxs_MeshValue<OC_REAL8m> K2;
   mutable Oxs_MeshValue<ThreeVector> axis;
 protected:
   virtual void GetEnergy(const Oxs_SimState& state,
                          Oxs_EnergyData& oed) const;
 public:
   virtual const char* ClassName() const; // ClassName() is
   /// automatically generated by the OXS_EXT_REGISTER macro.
   Oxs_SimpleAnisotropy(const char* name,  // Child instance id
-  Oxs_SimpleAnisotropy(const char* name,  // Child instance id
+  My_Anisotropy(const char* name,  // Child instance id
                        Oxs_Director* newdtr, // App director
                        const char* argstr);  // MIF input block parameters

   virtual ~Oxs_SimpleAnisotropy() {}
-  virtual ~Oxs_SimpleAnisotropy() {}
+  virtual ~My_Anisotropy() {}
   virtual OC_BOOL Init();
 };
-#endif // _OXS_SIMPLEANISOTROPY
+#endif // _MY_ANISOTROPY
```

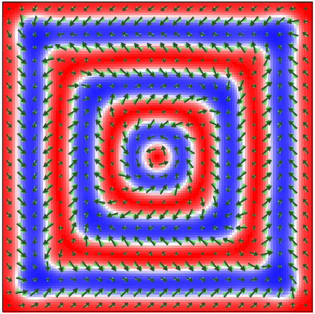
Sample anisotropy source file (part 1/2)

```
 ...
 #include "meshvalue.h"
-#include "simpleanisotropy.h"
+#include "myanisotropy.h"
 #include "energy.h"              // Needed to make MSVC++ 5 happy
 OC_USE_STRING;
 // Oxs_Ext registration support
-OXS_EXT_REGISTER(Oxs_SimpleAnisotropy);
+OXS_EXT_REGISTER(My_Anisotropy);

 /* End includes */

 // Constructor
-Oxs_SimpleAnisotropy::Oxs_SimpleAnisotropy(
+My_Anisotropy::My_Anisotropy(
   const char* name,     // Child instance id
   Oxs_Director* newdtr, // App director
   const char* argstr)   // MIF input block parameters
   : Oxs_Energy(name,newdtr,argstr), mesh_id(0)
 {
   // Process arguments
   OXS_GET_INIT_EXT_OBJECT("K1",Oxs_ScalarField,K1_init);
+  OXS_GET_INIT_EXT_OBJECT("K2",Oxs_ScalarField,K2_init);
   OXS_GET_INIT_EXT_OBJECT("axis",Oxs_VectorField,axis_init);
   VerifyAllInitArgsUsed();
 }

-OC_BOOL Oxs_SimpleAnisotropy::Init()
+OC_BOOL My_Anisotropy::Init()
 {
   mesh_id = 0;
   K1.Release();
+  K2.Release();
   axis.Release();
   return Oxs_Energy::Init();
 }
```

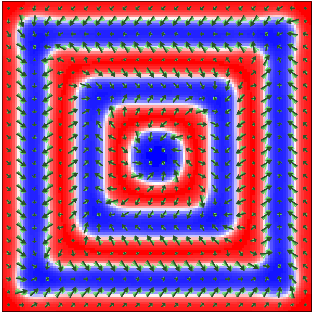
Sample anisotropy source file (part 2/2)

```
-void Oxs_SimpleAnisotropy::GetEnergy
+void My_Anisotropy::GetEnergy
 (const Oxs_SimState& state,
  Oxs_EnergyData& oed
  ) const
 {
   const Oxs_MeshValue<OC_REAL8m>& Ms_inverse = *(state.Ms_inverse);
   const Oxs_MeshValue<ThreeVector>& spin = state.spin;
 ...
   OC_INDEX size = state.mesh->Size();
   if(mesh_id != state.mesh->Id()) {
     // This is either the first pass through, or else mesh
     // has changed.
     mesh_id=0;
     K1_init->FillMeshValue(state.mesh,K1);
+    K2_init->FillMeshValue(state.mesh,K2);
     axis_init->FillMeshValue(state.mesh,axis);
     mesh_id=state.mesh->Id();
   }

   for(OC_INDEX i=0;i<size;++i) {
     OC_REAL8m field_mult = Ms_inverse[i]/MU0;
     if(field_mult==0.0) {
       energy[i]=0.0; field[i].Set(0.,0.,0.); continue;
     }
     const ThreeVector& u1 = axis[i];
     const ThreeVector&  m = spin[i];
     const OC_REAL8m k1 = K1[i];
+    const OC_REAL8m k2 = K2[i];
     OC_REAL8m dot = m*u1;
-    energy[i] = -k1*dot*dot;
-    field[i] = 2*k1*dot*field_mult*u1;
+    OC_REAL8m dotsq = dot*dot;
+    energy[i] = ((dotsq-2)*k2-k1)*dotsq;
+    field[i] = (2*field_mult*(k1+2*(1-dotsq)*k2)*dot)*u1;
   }
 }
```

Remanent magnetization configuration



Simple Anisotropy



Extended Anistropy

OOMMF Tutorial
Part III: Advanced Simulations

Spatially varying properties

Atlases can be used for discontinuous variation.

```
Specify Oxs_MultiAtlas:atlas {
 atlas { Oxs_BoxAtlas:left {
  xrange {0 500e-9}
  yrange {0 200e-9} zrange {0 20e-9}
 }}
 atlas { Oxs_BoxAtlas:right {
  xrange {500e-9 1000e-9}
  yrange {0 200e-9} zrange {0 20e-9 }
 }}
}
Specify Oxs_UniaxialAnisotropy {
 axis {0 0 1}
 K1 { Oxs_AtlasScalarField {
  atlas :atlas
  values {left 500e3 right 50e3}
 }}
}
```

Spatially varying properties

Use script fields for continuous variation.

```
set bfr 20e-9
proc Absorb { valM valB x y z } {
  global xmax ymax bfr
  set xoff $bfr
  if {$x<$bfr} {
    set xoff $x
  } elseif {$x>$xmax - $bfr} {
    set xoff [expr {$xmax - $x}]
  }
  set yoff $bfr
  if {$y<$bfr} {
    set yoff $y
  } elseif {$y>$ymax - $bfr} {
    set yoff [expr {$ymax - $y}]
  }
  set scale [expr {($xoff*$yoff)/($bfr*$bfr)}]
  return [expr {$valM*$scale+$valB*(1-$scale)}]
}
```

Spatially varying properties (edgedamp.mif)

```
proc Absorb { valI valB x y z } {
   ...
  return [expr {$valI*$scale+$valB*(1-$scale)}]
}

Specify Anv_SpinTEvolve:evolver [subst {
  alpha {Oxs_ScriptScalarField {
     script {Absorb $alpha 100}
     script_args rawpt
     atlas :atlas
  }}
  u $u
  beta $beta
  method rkf54s
}]
```

Patterned structures (dots.mif)

```
Parameter xcount  3 ;# Dots in x direction
Parameter ycount  2 ;# Dots in y direction
Parameter dotrad 40 ;# Dot radius, in nm
set xblocksize [expr {$xrange/$xcount}]
set yblocksize [expr {$yrange/$ycount}]
set dotrad [expr {$dotrad*1e-9}]

proc Dots { x y z } {
 global xblocksize yblocksize dotrad
 # Determine position relative to corresponding
 # dot center.
 set x [expr {fmod($x,$xblocksize)-0.5*$xblocksize}]
 set y [expr {fmod($y,$yblocksize)-0.5*$yblocksize}]
 if {$x*$x + $y*$y < $dotrad*$dotrad} {
  return 1  ;# Inside a dot
 }
 return 2   ;# Outside any dot
}
```

Patterned structures (dots.mif, cont.)

```
Specify Oxs_ScriptAtlas:atlas [subst {
 xrange {0 $xrange} yrange {0 $yrange}
 zrange {0 $zrange}
 regions { dot antidot }
 script Dots
 script_args { rawpt }
}]
...
Specify Oxs_MinDriver [subst {
 evolver :evolver mesh :mesh
 stopping_mxHxm 0.01
 Ms {Oxs_AtlasScalarField {
  atlas :atlas
  values { dot 8e5 antidot 0.0 }
 }}
 m0 {Oxs_RandomVectorField {
  min_norm 1.0 max_norm 1.0
 }}
}]
```

Layered structures (layers.mif)

```
Parameter layer1 6 ;# Layer thicknesses, in nm
Parameter layer2 4
Parameter layer3 2
set grpsize [expr {$layer1+$layer2+$layer3}]

proc Layers { x y z } {
  global layer1 layer2 layer3 grpsize
  # Determine layer in group
  set z [expr {$z*1e9}] ;# Convert to nm
  set zr [expr {fmod($z,$grpsize)}]
  if {$zr<$layer1} {
   return 1
  } elseif {$zr<$layer1+$layer2} {
   return 2
  }
  return 3
}
```

Layered structures (layers.mif, cont.)

```
Specify Oxs_ScriptAtlas:atlas [subst {
 xrange {0 $xrange}
 yrange {0 $yrange}
 zrange {0 $zrange}
 regions {layer1 layer2 layer3}
 script Layers
 script_args { rawpt }
}]

Specify Oxs_Exchange6Ngbr {
 default_A  0.0
 atlas :atlas
 A {
  layer1 layer1 20e-12
  layer2 layer2 12e-12
  layer1 layer2 16e-12
 }
}
```

Layered structures (layers.mif, tricky)

```
proc zHeight { x y z } {
 # Returns z height in nanometers
 return [expr {$z*1e9}]
}

Specify Oxs_ScriptScalarField:zHeight {
 atlas :atlas
 script zHeight
 script_args rawpt
}
```

Layered structures (layers.mif, tricky, cont.)

```
set zmax [expr {$zrange*1e9}]
for {set i 1} {$i*$grpsize<$zmax} {incr i} {
 set zhi  [expr {$grpsize*$i}]
 set zlow [expr {$zhi-$layer3}]
 Specify Oxs_TwoSurfaceExchange:set${i} [subst {
  sigma -1e-4
  surface1 {
   atlas :atlas  region layer2
   scalarfield :zHeight
   scalarvalue $zlow  scalarside -
  }
  surface2 {
   atlas :atlas  region layer1
   scalarfield :zHeight
   scalarvalue $zhi  scalarside +
  }
 }]
}
```

Time varying applied fields (Hping.mif)

```
proc Sinc { t } {
 if {abs($t)<1e-6} {
  set v [expr {1-$t*$t/6.}]
  set dv [expr {$t*$t*$t/-3.}]
 } else {
  set v [expr {sin($t)/$t}]
  set dv [expr {($t*cos($t)-sin($t))/($t*$t)}]
 }
 return [list $v $dv]
}
proc SincPulse { total_time } {
 global amp scale offset
 set st [expr {$scale*($total_time - $offset)}]
 set vals [Sinc $st]
 set Hy [expr {$amp*[lindex $vals 0]}]
 set dHy [expr {$amp*$scale*[lindex $vals 1]}]
 return [list 0 $Hy 0 0 $dHy 0]
}
```

Time varying applied fields (Hping.mif, cont.)

```
Specify Oxs_ScriptUZeeman [subst {
  multiplier [expr {0.001/$mu0}]
  script SincPulse
  script_args total_time
}]

Specify Oxs_TimeDriver [subst {
  evolver :evolver
  mesh :mesh
  stopping_time $stage_time
  stage_count $number_of_stages
  Ms {Oxs_AtlasScalarField {
  atlas :atlas
  default_value 0.0
   values { ellipsoid 8e5 }
  }}
  m0 {1 0 0}
}]
```

Current pulse (spinping.mif)

```
proc Sinc { t } {
 if {abs($t)<1e-6} {
  set v [expr {1-$t*$t/6.}]
 } else {
  set v [expr {sin($t)/$t}]
 }
 return $v
}

proc SincPulse { total_time } {
 global pulse_scale pulse_offset
 set t [expr {$total_time - $pulse_offset}]
 set st [expr {$t*$pulse_scale}]
 return [Sinc $st]
}
```

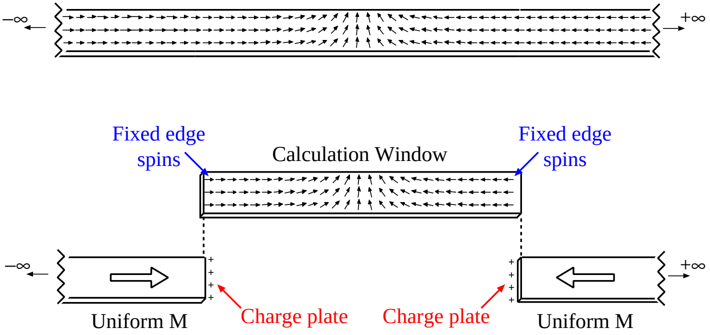
Current pulse (spinping.mif, cont.)

```
Specify Anv_SpinTEvolve [subst {
 do_precess 1
 gamma_LL 2.21e5
 method rkf54s
 alpha 0.005
 fixed_spins { atlas fixed }
 u $u_max
 u_profile SincPulse
 u_profile_args total_time
 beta 0.04
}]
```

⇒ Current density at point (x,y,z) is proportional to

$$u_{\text{profile}}(t) \cdot u(x,y,z).$$

Infinite strips



See spinping.mif for an example.

R.D. McMichael & M.J. Donahue, *IEEE Trans. Magn.*, **33**, 4167 (1997).