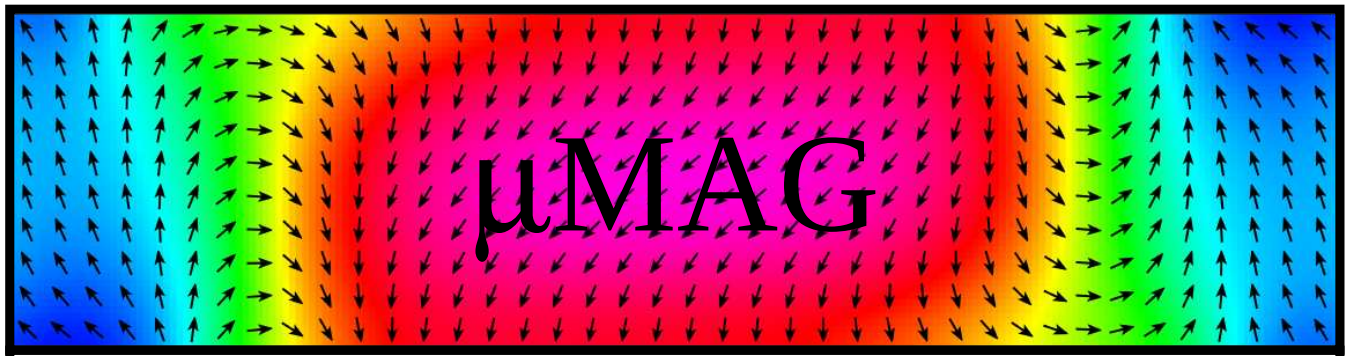


Micromagnetic Modeling Activity Group

http://www.ctcms.nist.gov/~rdm/mumag.html

Computation of magnetic material & device properties

Domain structure - Shape effects - Dynamics

Standard Problems for Micromagnetics

Web hosting for

- Problem Specifications
- Submitted results commentary & comparison

Mailing list for discussion & announcements

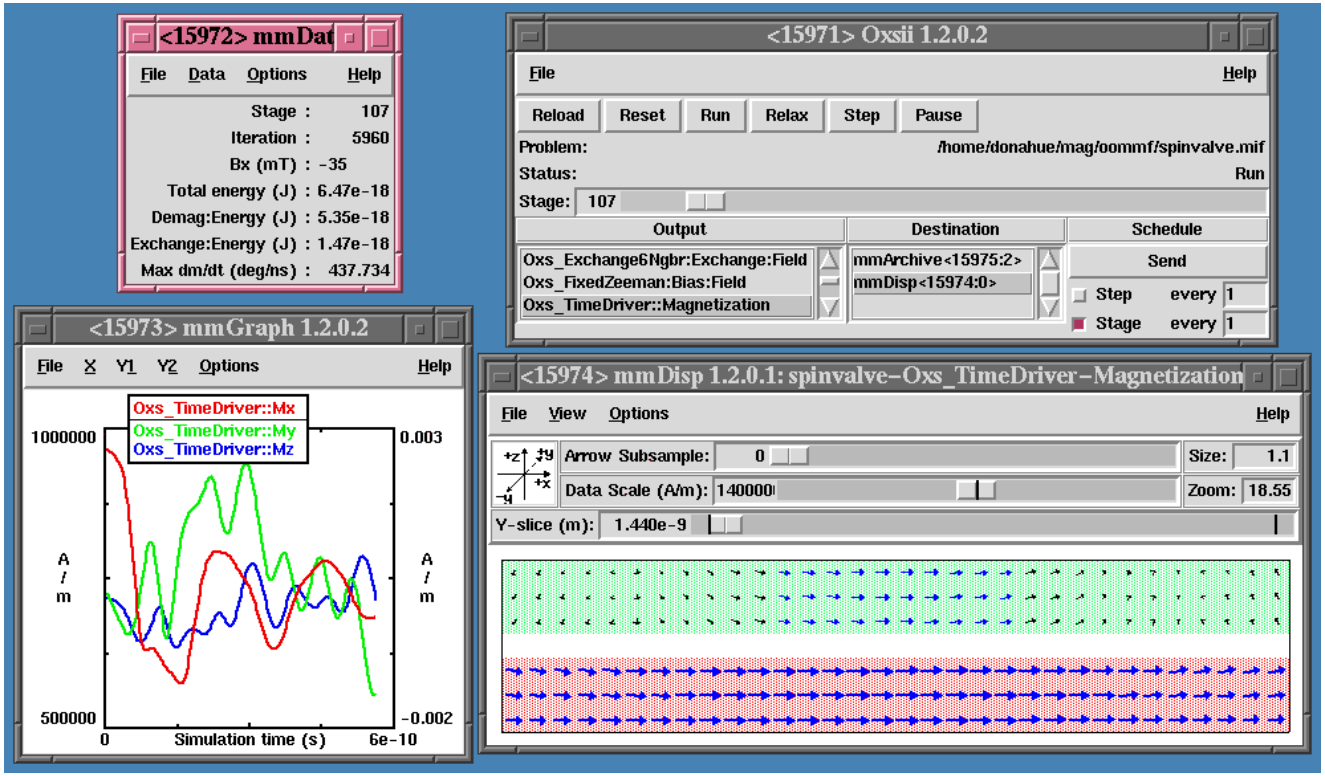
Contacts: Bob McMichael and Mike Donahue (NIST)

Portable, extensible, public domain programs & tools for micromagnetics

OOMMF

http://math.nist.gov/oommf

- Fully 3D
- FFT-base demag
- Landau-Lifshitz & energy minimization solvers
- Bilinear, biquadratic & long range (RKKY) exchange
- Time varying applied fields
- All parameters ptwise adj.



Contacts: Mike Donahue, Don Porter (ITL/NIST)

Requirements

- Unix/X, Windows 95+, Mac OS X
- Tcl/Tk 7.6/4.2+ (8.0+ recommended)
  Tcl Developer Xchange: http://www.tcl.tk
- TCP/IP networking on localhost
- RAM: 32 MB (minimum)
- Disk space: 25/80 MB
- Modern C++ compiler to build from source

OOMMF Testbeds

Platform	Compilers
Alpha/Linux	Compaq C++, Gnu gcc
Alpha/Tru64	Compaq C++ (cxx)
Alpha/Windows NT	Microsoft Visual C++
HP-UX	aCC
Intel/Linux	Gnu gcc
Intel/Windows	Microsoft Visual C++, Intel C++, Cygwin gcc, Borland C++
MIPS/IRIX 6 (SGI)	MIPSpro C++, Gnu gcc
SPARC/Solaris	Sun Workshop C++, Gnu gcc

3D Solver, Oxsii

- 3D spins on 3D grid of rectangular elts
- LLG and energy minimization solvers
- Demag calculated via FFT
- All material parameters ptwise selectable (shaped elts)
- Applied fields varied spatially and temporally
- Bilinear, biquadratic, and long range exchange
- Fixed spins
- Interactive and batch modes

OOMMF 1.2a4 features

- All features of 1.1b2
- External library support
  - FFTW
  - CVODE
- Runge-Kutta-Fehlberg solver
- New initialization routines
- Restart feature in Oxs

Release date: 13-February-2004

Extending OOMMF: Input scripts

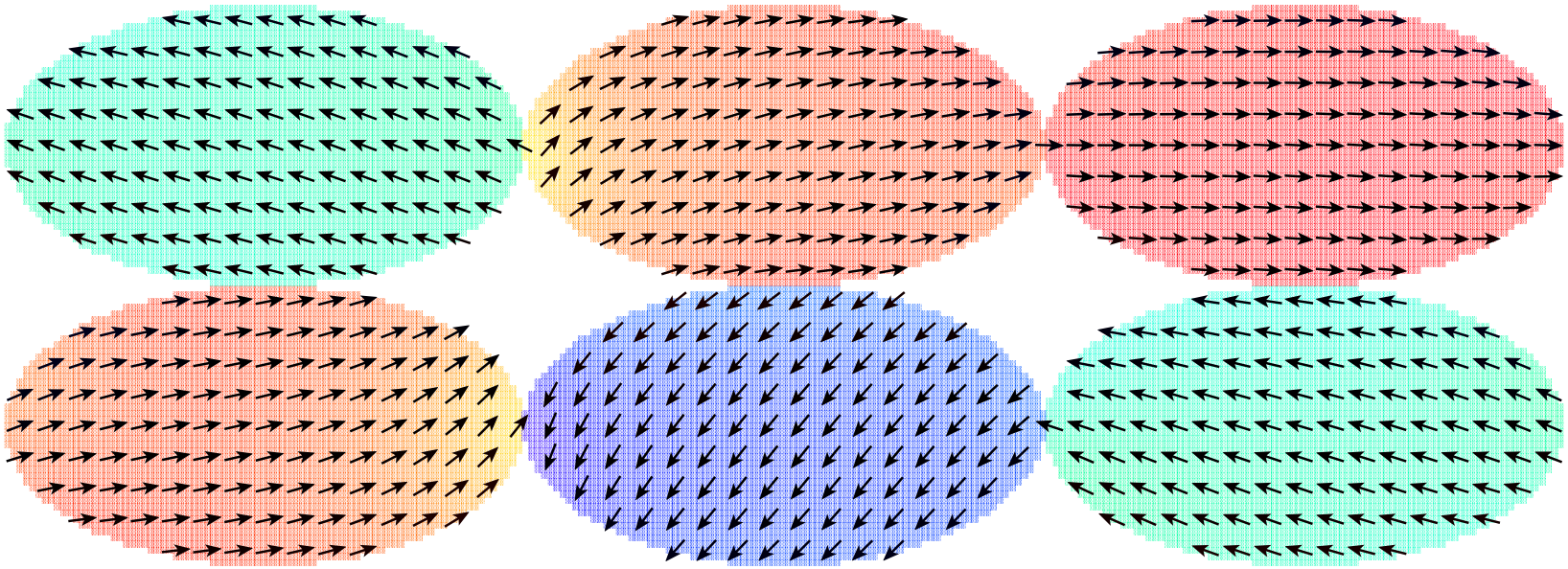
- Input scripts (MIF) are Tcl code
- Adjust:
  - material parameters
  - part geometry
  - initial magnetization
  - applied fields (time & space)

Sample MIF file

```
# MIF 2.1
Specify Oxs_BoxAtlas:atlas {
  xrange {0 300e-9}  yrange {0 200e-9}  zrange {0  16e-9}
}
Specify Oxs_RectangularMesh:mesh {
  cellsize {5e-9 5e-9 4e-9}
  atlas :atlas
}
Specify Oxs_UniformExchange { A  8E-12 }
proc Vortex { x y z } {
    set xrad [expr $x-0.5]
    set yrad [expr $y-0.5]
    set normsq [expr $xrad*$xrad+$yrad*$yrad]
    if {$normsq <= 0.0125} {return "0 0 1"}
    return [list [expr -1*$yrad] $xrad 0]
}
...
```

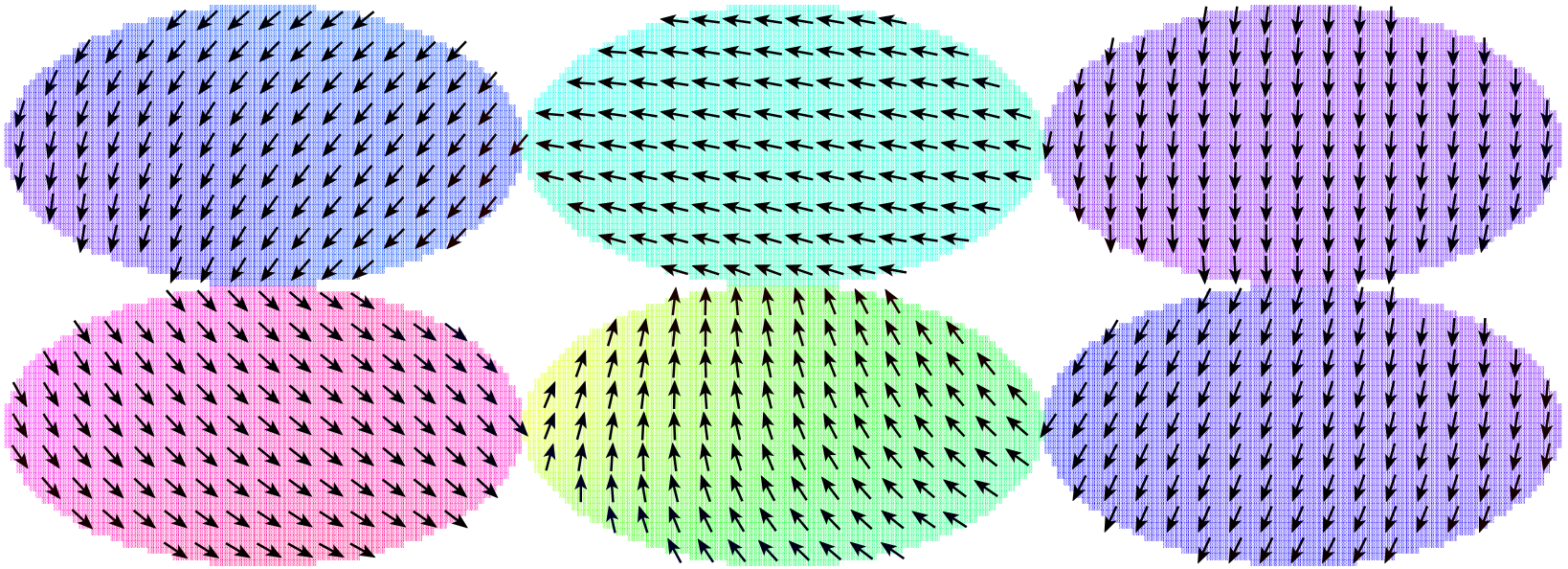
Geometry script

Layer 1:

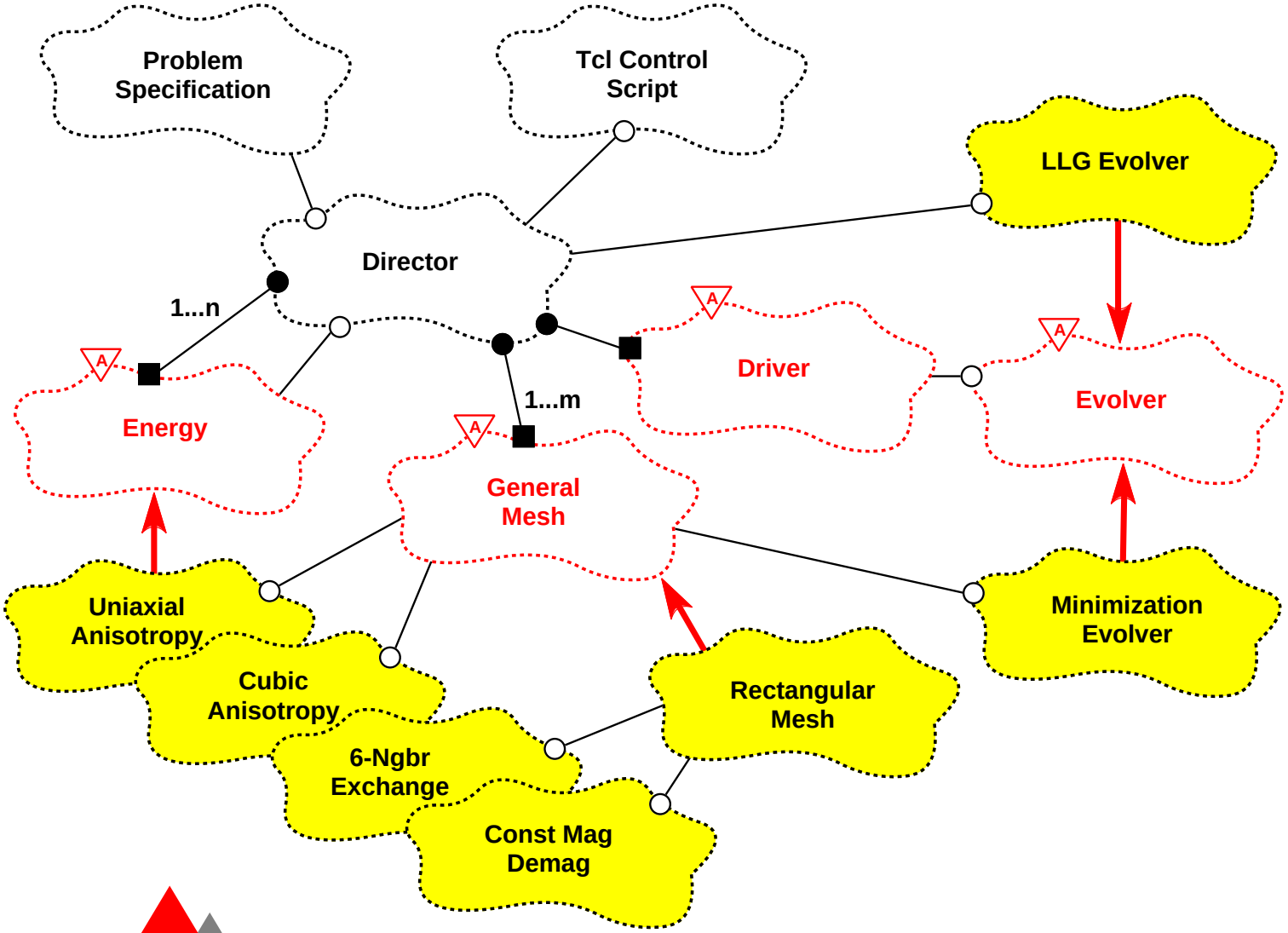


Geometry script

Layer 2:



Oxs framework



Extending OOMMF: New Ext classes

1. Create Oxs_Ext class file
   - Initializer
   - Energy & field routine
2. Run pimake
3. Create input file

Oxs_Ext initializer

```
// Constructor
Oxs_HexAnisotropy::Oxs_HexAnisotropy(
  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("K3",Oxs_ScalarField,K3_init);
  OXS_GET_INIT_EXT_OBJECT("K4",Oxs_ScalarField,K4_init);
  OXS_GET_INIT_EXT_OBJECT("axis1",Oxs_VectorField,axis1_init);
  OXS_GET_INIT_EXT_OBJECT("axis2",Oxs_VectorField,axis2_init);
  VerifyAllInitArgsUsed();
}
```

Oxs_Ext energy

```
void Oxs_HexAnisotropy::GetEnergy
(const Oxs_SimState& state,Oxs_EnergyData& oed) const {
<SNIP: Boilerplate initialization>
  for(UINT4m i=0;i<size;++i) {
    const ThreeVector&  m = spin[i];
    const REAL8m k1 = K1[i];    const REAL8m k2 = K2[i];
    const REAL8m k3 = K3[i];    const REAL8m k4 = K4[i];
    const ThreeVector& u1 = axis1[i];
    const ThreeVector& u2 = axis2[i];
    ThreeVector mxu1 = m;    mxu1 ^= u1;
    REAL8m sintheta2 = mxu1.MagSq();
    REAL8m sintheta6 = sintheta2*sintheta2*sintheta2;
    REAL8m costheta = m*u1;
    REAL8m cosphi = m*u2;
    REAL8m cosphi2 = cosphi*cosphi;
    REAL8m cos6phi = ((32*cosphi2-48)*cosphi2+18)*cosphi2-1;
```

Oxs_Ext (cont.)

```
    // Compute energy at cell i
    energy[i] = (((k4*cos6phi+k3)*sintheta2+k2)*sintheta2+k1)
      *sintheta2;

    // Compute field at cell i
    REAL8m Hcoef1 = (((k4*cos6phi+k3)*3*sintheta2+2*k2)*sintheta2+k1)
      *2*costheta;
    REAL8m Hcoef2 = -12*k4*cosphi*sintheta6
      *((cosphi2-1)*16*cosphi2+3);
    field[i] = Hcoef1*u1;
    field[i] += Hcoef2*u2;
    field[i] *= field_mult;
  }
}
```

Sample input file

```
...
Specify Oxs_HexAnisotropy {
  K1  -4.5e3
  K2   100
  K3   100
  K4   1e4
  axis1 {0 0 1}
  axis2 {1 0 0}
}
...
```

Future OOMMF

- Programming manual
- Contrib site
- Modeling of thermal effects
  - Langevin dynamics
  - Energy barriers

Future OOMMF?

- Oxs problem editor
- Periodic boundaries
- Mini test problems for validation
- Parallel processing
- ???