Readme version 1.2, 02-07-2019. Kristof Lebecki, lebecki@fuw.edu.pl

Disclaimer

This software was developed during my stay at the University of Konstanz (Germany), in the group of Prof. Nowak. Another person that helped me strongly was Prof. Chubykalo-Fesenko. Of course, I would not be able to accomplish this work without help of Michael Donahue.
You can use it in any way you wish (there is no special copyright). If you want to improve it however and modify it, please contact the authors – certainly they will appreciate your help.
I would appreciate acknowledgement if the software is used. When referencing, I recommend citing following paper:

"Key role of temperature in ferromagnetic Bloch point simulations"
KM Lebecki, D Hinzke, U Nowak, O Chubykalo-Fesenko
Physical Review B **86** (9), 094409

1 Overview

The aim of this software is to support the Landau-Lifshitz-Bloch equation (LLB) [Garanin
Phys. Rev. B 55, 3050, 1997] in the OOMMF project.
To accomplish it, a few new Oxs_Energy classes have been created for realization of LLB. Also, the evolver and driver had to be adapted.

One thing the user should be aware is extension of the magnetization description. Usually, in micromagnetism you describe the state of the system with

- the magnetization saturation, M_s,
- the direction of the magnetization, $\mathbf{m}$.

Because is M_s constant you often focus on the variable $\mathbf{m}$ – a *normalized* vector.
In temperature-aware simulations the situation is more complex. Beside the magnetization saturation (measured at zero temperature, $M_s(T=0)$) you have:

- equilibrium magnitude of the magnetization at the given temperature, $M_e(T)$,
- the actual magnetization having magnitude that possibly differs from M_e. Thus instead of unit-less field $\mathbf{m}(\mathbf{r},t)$ we describe it with a vector-field having units of magnetization $\mathbf{M}(\mathbf{r},t)$.

This approach required changes in the original OOMMF files. There, the state of the system is described with more variables now.

Another point worth to mention is the noise. In the LLB equation you have a stochastic term describing thermal fluctuations. You do not have it in my implementation, as I wanted to use the fast Runge-Kutta evolver. So, use this program in cases where you can neglect thermal fluctuations.

2 Installation

First of all, you must have the OOMMF 2.0 alpha 1 release (snapshot 2018.09.30). My program might work with different version of OOMMF, but I have not tested it.

Second, you must have a working compilation environment. I.e. you must be able to

remove all OOMMF binaries (if they exist) and create them "from scratch", by help of the compiler of course. The way to check the status of your compiler is described in the installation section of the OOMMF user's guide (chapter "Check Your Platform Configuration"), instructions how to compile OOMMF are in the next chapter ("Compiling and Linking").
If you have any error messages stop proceeding and contact the OOMMF authors.

Then, remove my old files from the app/oxs/local directory (if present):

```
        kl_changes.txt
        kl_demag.cc
        kl_demag.h
        kl_help.doc
        kl_infinite_prism.mif
        kl_pbc_util.cc
        kl_pbc_util.h
        kl_progress.tcl
        kl_simpledemag.cc
        kl_simpledemag.h
        kl_uniformexchange.cc
        kl_uniformexchange.h
```

They are obsolete, I think. Their functionality—periodic boundary conditions—is now available through standard OOMMF package.

Copy the following files to the OOMMF's app/oxs/local directory:

```
        kl_llb_util.cc
        kl_llb_util.h
        kl_llbrungekuttaevolve.cc
        kl_llbrungekuttaevolve.h
        kl_llbterm.cc
        kl_llbterm.h
        kl_timedriver.cc
        kl_timedriver.h
        kl_timeevolvervarms.cc
        kl_timeevolvervarms.h
        kl_uniformexchange.cc
        kl_uniformexchange.h
```

Copy **and overwrite** the following files in the OOMMF's app/oxs/base directory. For your information, you have original version of these files in my base/ORG directory.

```
        chunkenergy.cc
        driver.cc
        driver.h
        simstate.cc
        simstate.h
```

(Alternatively, you can apply the differences to the existing files. You have to do it, however, on your own.)

Run the compilation script **pimake, see the** OOMMF user's guide:

```
        tclsh oommf.tcl pimake upgrade
        tclsh oommf.tcl pimake distclean
        tclsh oommf.tcl pimake
```

You should see a message about successful compilation of the files, then about the update of the OXS binary. I.e. something like this (I was compiling with g++ under Windows):

```
C:\>tclsh oommf.tcl pimake
Updating C:/oommf/oommf20a0_20170929.fresh.app.01/pkg/oc/tclIndex ...
Making directory 'C:/oommf/oommf20a0_20170929.fresh.app.01/app/mmdisp/windows-x8
6_64' ...
(...)
b -L../../pkg/oc/windows-x86_64 -loc C:/programs/ActiveTcl64/lib/tk85.lib C:/pro
grams/ActiveTcl64/lib/tcl85.lib -Wl,--subsystem,console -o windows-x86_64/oxs.exe
Updating windows-x86_64/appindex.tcl
<9504> pimake 2.0a0  info:
```

```
Target '' up to date.
Built Thu Jan 04 16:33:49 CET 2018
Current time: Thu Jan 04 16:33:49 CET 2018
```

In case of any strange messages I recommend contacting me.

3 Usage

Following are described the changes to your "usual" MIF specification.
Only two parameters here are optional (`normalize_aveM_output`, `stopping_dM_dt`), the rest must be set.

3.1 Klm_UniformExchange

Specify this **instead** of the Oxs_UniformExchange term.
Parameters

- `kernel`
  Value (fixed): `6ngbrLLB`
- `A`
  Value: exchange constant for given temperature (Joule/meter).
  Site note: $A(T=0)$ is the usual used value of exchange constants. For T approaching T_C, the exchange constant drops to zero.

3.2 Klm_LLB_Term

Specify this **additionally** to other energy terms.
Parameters

- `chi_parallel`
  Value: parallel susceptibility for given temperature (unitless).
  Site note: $chi_{||}(T=0)$ is equal to zero. For T approaching T_C, it diverges positively.

3.3 Klm_LLB_RKEvolve

Specify this **instead** of the Oxs_RungeKuttaEvolve evolver.
Parameters (I list here only LLB-relevant)

- `relative_temperature`
  Value: given temperature divided by the Curie temperature (unitless).

3.4 Klm_TimeDriver

Specify this **instead** of the Oxs_TimeDriver driver.
Parameters (I list here only LLB-relevant)

- `normalize_aveM_output`
  Value: 0 to avoid normalization of the magnetization in the output. Or 1 to have it normalized.
  Optional.
  Default: 1.
- `stopping_dM_dt`
  Use this parameter instead of OOMMF's `stopping_dm_dt`.
  Value: Ampere/(meter*second).
  Optional.
  I suggest something close to $1.5*10^{11}$ (I work mostly with permalloy). Of

course, you can use alternative stopping criteria.

- `Ms_T0`
  Value: magnetization saturation at zero temperature (Ampere /meter).
- `Ms`
  Value: equilibrium magnetization at given temperature (Ampere /meter).
- `Ms_initial`
  Value: initial magnetization length (Ampere /meter).
  You will probably set it often equal to `Ms`, but this is not necessary. You can choose any positive value, just avoid zero: this might lead to numerical troubles.

3.5 Sample MIF file

Attached sample MIF file shows typical (for me…) usage of the parameters.

4 Restrictions, limitations, bugs

- Setting T=0 leads to much slower simulation.
  Plus, there might be some loss in the energy precision. I have not investigated it deeper, but comparisons with standard LLG simulations showed no remarkable difference/deviation.
  Also, in this case setting parallel susceptibility to zero might lead to numerical issues, I am afraid.
  To make long story short: My advice is setting T=1 K, or something similar.
- No support for the `-restart` parameter.
  So far…

5 FAQ, frequently asked questions

Q: Will you include the noise?
A: No, for that use the code of Yu Yahagi, see https://github.com/yuyahagi/oommf-llb.

6 License

Public Domain.