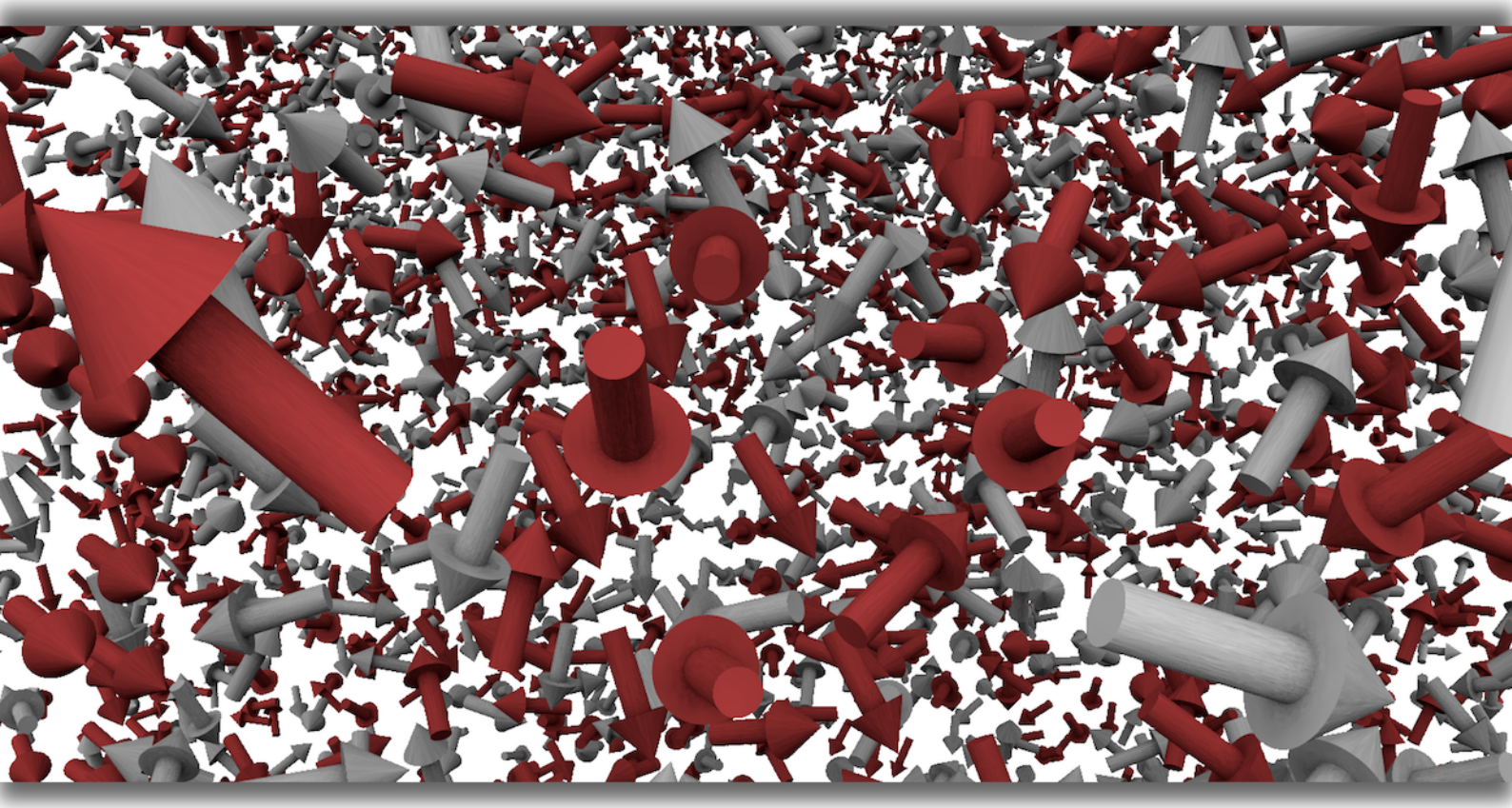


Uppsala Atomistic Spin Dynamics User Guide

J. Hellsvik, L. Bergqvist, A. Bergman, J. Chico *et al.*

HTTP://PHYSICS.UU.SE/UPPASD

June 2019

Contents

1 Introduction 5

1.1 Theoretical overview 5
1.2 License 6
1.3 Installation 6
1.4 Principles of the Code 7

2 Input files 9

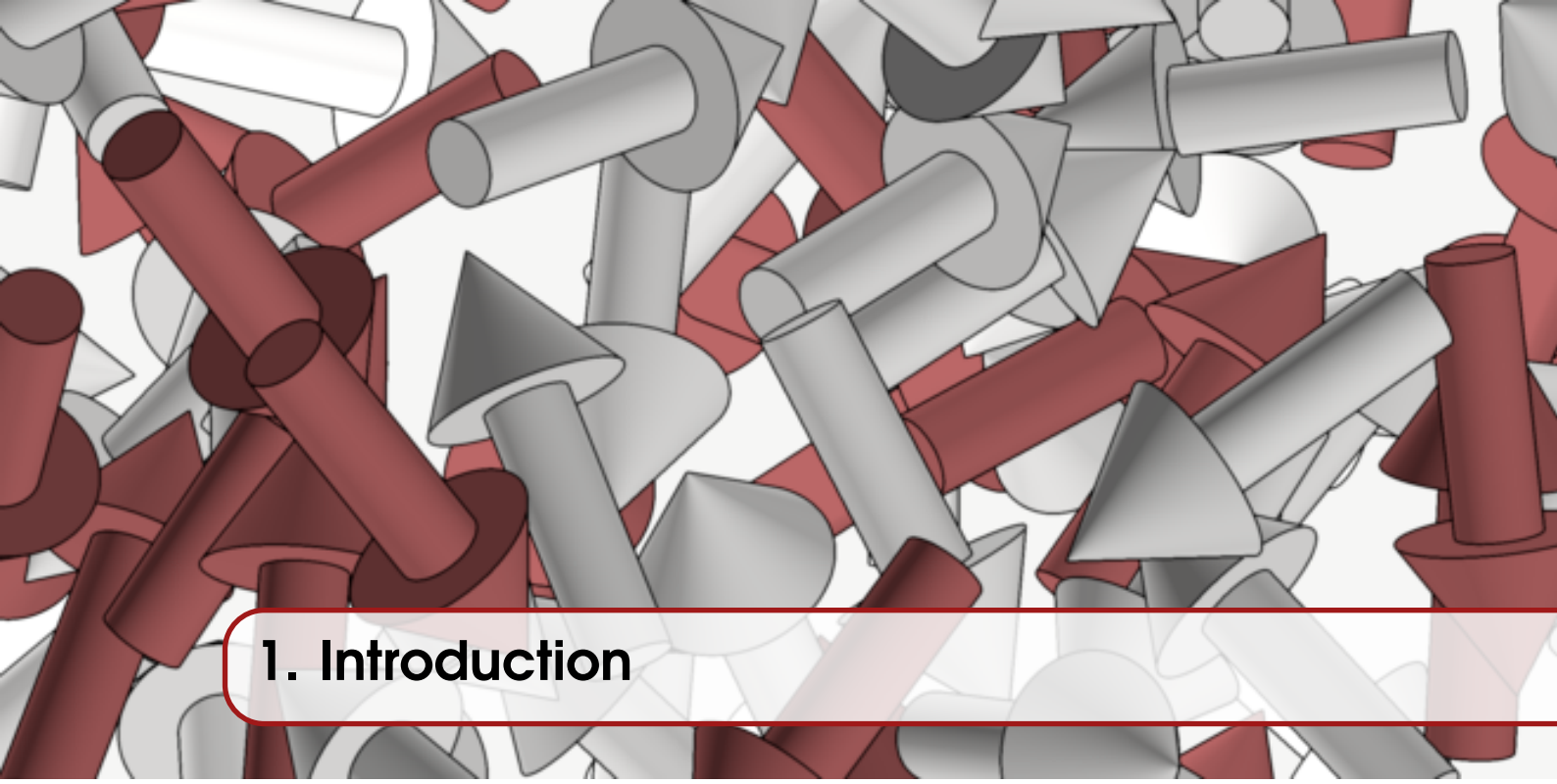
2.1 inpsd.dat 9
2.1.1 Random alloys and more than one atom in the cell 11
2.2 Input Entries 13
2.2.1 System parameters 14
2.2.2 Hamiltonian parameters 14
2.2.3 General simulation parameters 16
2.2.4 Initialization parameters 17
2.2.5 Initial phase parameters 17
2.2.6 Measurement phase parameters 18
2.2.7 Parameters for measuring of observables 18
2.3 Output files 20
2.3.1 Simulation and Hamiltonian output 20
2.3.2 Measured observables 21

3 Tutorial and walkthrough 25

3.1 Dynamics of a single spin 25
3.1.1 Switching under external magnetic field 27
3.1.2 Thermal effects 28

3.2 Determination of T_c of a ferromagnetic material 30
3.3 Dynamical correlations and magnon spectra 32
3.3.1 Ferromagnetic magnons . . . 32
3.3.2 Magnons in antiferromagnets and spin spirals . . . 34

4 Examples . . . 37
4.0.1 bcc Fe . . . 37
4.0.2 Heisenberg Spin Chain . . . 37
4.0.3 Two-dimensional Systems . . . 37
4.0.4 fcc Co . . . 38
4.0.5 GaMnAs . . . 38
4.0.6 Random Alloy . . . 38
4.0.7 SKKR Input (test) . . . 38



1. Introduction

1.1 Theoretical overview

This user guide describes the essential features of Uppsala Atomistic Spin Dynamics program (UppASD). The emphasis is on the input files necessary to run calculations using UppASD and the output files it generates. Some information concerning the analysis of data generated by the code is also given. The ASD method and our implementation of it is described in the article by Skubic *et al.* [1]. The underlying theory is described in the articles by Antropov *et al.* [2] and García-Palacios and Lázaro [3]. An old, yet remarkably lucid overview is also given by Watson *et al.* [4]. A thorough review of the method and relevant applications is given in the book by Eriksson *et al.* [5].

The program evolves the equations of motion for atomic magnetic moments in a solid. These take the form of the Landau-Lifshitz-Gilbert (LLG) equation,

Equation 1 — The Landau-Lifshitz-Gilbert (LLG) equation.

$$\frac{d\mathbf{m}_i}{dt} = -\frac{\gamma}{1+\alpha^2}\mathbf{m}_i \times [\mathbf{B}_i + \mathbf{b}_i(t)] - \frac{\gamma}{m_i}\frac{\alpha}{1+\alpha^2}\mathbf{m}_i \times (\mathbf{m}_i \times [\mathbf{B}_i + \mathbf{b}_i(t)])$$

In this expression, γ is the gyromagnetic ratio and $\mathbf{b}_i(t)$ is a stochastic magnetic field with a Gaussian distribution, the magnitude of which is related to the damping parameter α, which eventually brings the system into thermal equilibrium. The typical time step for solving the differential equations is $\Delta t = 0.1$ femtoseconds, *i.e.* 10^{-16} seconds.

The effective field, $\mathbf{B}_i$, experienced by each atom i is given by the partial derivative of the Hamiltonian $\mathscr{H}$ with respect to the local magnetic moment $\mathbf{m}_i$.

Equation 2 — The effective magnetic field.

$$\mathbf{B}_i = -\frac{\partial \mathscr{H}}{\partial \mathbf{m}_i}$$

where $\mathscr{H}$ is the spin Hamiltonian that takes all relevant interactions into account. Note that in the input files to UppASD, the convention is that it is the unit vectors $\mathbf{e}_i = \frac{\mathbf{m}_i}{m_i}$ that enter the Hamiltonian. This is consistent with most electronic structure codes that outputs J_{ij} and other exchange interactions but care should be taken since in many other models found in the litterature, the Hamiltonian depends explicitly of $\mathbf{m}_i$ instead.

Equation 3 — The spin Hamiltonian.

$$\mathscr{H} = -\sum_{i,j} J_{ij}\mathbf{e}_i \cdot \mathbf{e}_j - \sum_{i,j} \mathbf{D}_{ij}\mathbf{e}_i \times \mathbf{e}_j - \sum_i K_i(\hat{\mathbf{e}}_i \cdot \mathbf{e}_i^K)^2 - \sum_i \mathbf{B}^{ext} \cdot \mathbf{e}_i + \ldots$$

The most important contribution to the Hamiltonian is typically given by the Heisenberg exchange Hamiltonian, given by the first term in Eqn. 3. There, i and j are atomic indices, and J_{ij} is the strength of the exchange interaction. These exchange interactions can be obtained from first-principles calculations, or alternatively even inferred from experiments. It is also possible (and in many cases even essential) to include other terms to the Hamiltonian, including Dzyaloshinskii-Moriya exchange, magnetic anisotropies and external magnetic fields, as are also exemplified in Eqn. 3. There are currently several other interactions available in the UppASD code and additional interactions can be implemented quite straightforwardly. Please note that the format of the Hamiltonian can be defined differently regarding prefactors, inclusion of moment magnitudes, summation convention and more. The input format in UppASD is conformal with most commonly used electronic structure codes that have the capability of calculating J_{ij} (and sometimes $\mathbf{D}_{ij}$, i.e. following the same convention of the Heisenberg Hamiltonian as Liechtenstein *et al.* [6].

1.2 License

The UppASD code is developed by the Division of Materials Theory, in the Department of Physics and Astronomy at the University of Uppsala, Sweden. The copyright of the code is help by the developers but the program is open for use and distribution according to the fXXX license. Further information concerning the license and contact information of the developers may be found on the UppASD webpage: `http://www.physics.uu.se/UppASD`. The current version of the code (5.0) is still under active development.

1.3 Installation

The source code is distributed along with documentation and a growing set of examples. To install, perform the following actions.

1. Obtain the code, by extracting the downloaded archive (or by pulling from the git repository)

   ```
   tar xvzf  UppASD_dist.tar.gz
   ```

   or in the case of git

   ```
   git clone github:uppasd
   ```

2. Generate the dependencies needed for compiling the code

   ```
   make deps
   ```

3. (Optional) Perform a system check for available compiler profiles

   ```
   make probe
   ```

4. Compile the code with the selected compiler profile

   ```
   make <profile>
   ```

   where `<profile>`is the name of the profile, i.e. `ifort`, `ifort-cuda`, `gfortran`, `gfortran-osx`, and so on. Eg. `make ifort`

5. (Optional) Test the compiled program against a selection of realistic runs

   ```
   make tests
   ```

In addition to the source files, the UppASD distribution also contains several examples (in the directory `examples_revision_controlled/`), documentation, including this file (in `docs/`) and routines and reference data (`codeTester/`) for validating the installation of the UppASD program.

1.4 Principles of the Code

When run, UppASD essentially goes through three stages:

1. Initialization: the system is set up.
2. Initial phase: an optional stage in which the system is brought into thermal equilibrium, with limited data sampling.
3. Measurement phase: the system is evolved in time, with complete data sampling being made.

During the initialization phase, all the parameters necessary to describe the system of interest, such as its geometry, dimensions, exchange couplings and boundary conditions, are set up. In addition, the initial phase also sets the simulation parameters, such as the number of simulation steps to record data over, which SDE solver to use, and the temperature at which the simulation should be run.

The initial phase, which is optional, is typically performed in order to bring the system into thermal equilibrium, so that the data recorded in the measurement phase is for a thermalized system. Obviously, if one is interested in out-of-equilibrium dynamics, then there is no need to perform this phase. The initial phase can either be performed using Spin Dynamics (SD), or the Metropolis or Heatbath Monte Carlo (MC) algorithms [7]. The latter is convenient for ground state searches, provided the system is not too complex (*i.e.* a spin glass).

During the measurement phase, the data sampling is performed. Simulations can be run in either MC or SD mode. In MC mode only magnetization averages and static correlation functions may be measured. In SD mode, a much richer set of observables are measured, including the dynamical structure factor.

2. Input files

2.1 inpsd.dat

A file with the hardcoded name `inpsd.dat` is the main input file necessary to run UppASD. Contained in this file are also the names of the files containing the exchange interactions, the atomic positions, and the atomic moments. Although the names of these files is arbitrary, in this manual they are referred to as the `jfile`, `posfile` and `momfile`, respectively. Other optional files containing information such as the uniaxial anisotropy and the Dzyaloshinskii-Moriya vectors may also be included, as described below.

The input format is keyword based. The code is programmed to search for given keywords, and then read in the values that follow. If no keyword is given, a default value is set. As an example of a standard `inpsd.dat` file layout, input for a Fe in bcc lattice is shown below (as found in the examples directory). More advanced examples like supercells and random alloys follows later, but let's keep things simple for now:

```
simid bccFe100
ncell  12              12              12  System size
BC     P               P               P   Boundary conditions (0=vacuum, P=periodic)
cell  -0.5000000000    0.5000000000    0.5000000000
       0.5000000000   -0.5000000000    0.5000000000
       0.5000000000    0.5000000000   -0.5000000000
Sym    0  Symmetry of exchange bonding vectors (0 for no, 1 for cubic, 2 for 2d cubic, 3 fc

posfile    ./posfile
momfile    ./momfile
exchange   ./jASD1
anisotropy ./kfile

do_ralloy  0
Mensemble  1
tseed      4499
```

```
maptype    1

SDEalgh    1                          SDE solver: 1=midpoint, 2=heun, 3=heun3, 4=Heun_proper
Initmag    3                          Initial config of moments (1=random, 2=cone, 3=spec.,
#restartfile ./restart.bccFe100.out

ip_mode     M                         Initial phase parameters
ip_mcanneal 1                         --
10000 300 1.00e-16 0.3                --

mode     M                            S=SD, M=MC
temp     300                          Measurement phase parameters
mcNstep  50000
Nstep    50000
damping  0.1
timestep 1.0e-16

do_avrg        Y                      Measure averages

do_cumu        Y
cumu_step      50
cumu_buff      10

do_tottraj     N                      Measure moments
tottraj_step   1000

plotenergy     1

do_sc          C
do_ams         Y
do_magdos      Y
magdos_freq    200
magdos_sigma   30
qpoints        C

do_stiffness   Y
eta_max        12
eta_min        6
alat           2.83e-10
```

While the meaning of most of the entries in this particular example may be obvious, each input field will be described later in this manual. In short, the input will perform a Monte Carlo simulation at T=300 K allowing 10 000 steps to reach equilbrium and then 50 000 steps to measure observables. However, it is also clear that more information than in inpsd.dat are required and must be read in from external files in order for the system to be fully defined. These are:

posfile

The positions of the atoms in the unit cell are given in basis vector coordinates. While these can be listed directly in the `inpsd.dat` file, it is typically more convenient to give them in an external file, `posfile`. For the example above the positions are given as :

```
1 1   0.000000  0.000000  0.000000
```

The first entry indicates the site number, whereas the second one indicates the atom type. The concept of atom type is central when setting up UppASD simulations because every `type` is associated with a particular set defined exchange couplings. In this specific case there is only one atom type (and one site), namely Fe with atomic position at the origin.

momfile

This file lists the magnetic moments of the atoms in the unit cell. Also, if the `initmag` entry is set to 3, the initial direction of the moments is read from this file. For bcc Fe:

```
1 1 2.2459 0.0 0.0 1.0
```

The first entry indicates the site number (same as first column in posfile), the second entry the chemical type (always 1 for non-random systems), and the third entry indicates the magnitude of the magnetic moment in μ_B, as calculated or estimated from an electronic structure calculation or similar. The last three entries indicate the initial x, y, and z components of the moment (assuming `initmag` is set to 3).

exchange

This file lists the exchange couplings within the system. The content and length of this file depends on the symmetry of the system, and the number of atom types present. If no symmetry is used, i.e. sym 0 (as in example), all exchange interactions within each interaction shell must be specified. For the bcc lattice, that means that first shell contain 8 interactions and so forth. If symmetry is used, then only one interaction in each shell is specified and the program will automatically found the others within the shell depending on the crystal symmetry. For the present Fe example using maptype 1, the first line reads:

```
1 1 -0.500 -0.500 -0.500 1.359407144 0.866
```

The first two entries indicate the sites, which corresponds to the types that one whishes to map, i and j. In this case as both atoms have the same type, one can indicate the interactions between atoms in site 1 and 2, as 1-1, an example on how to deal with more atom types in the unit cell will be presented afterwards.

The third, fourth and fifth entries specify the interaction vector between the atoms and depending on choice of the maptype, it has different meaning. Using maptype 1, the vector is specified in carteisan coodinates. If the SPR-KKR software is used, that is directly columns eight, nine and ten in the exchange parameter outfile. If instead maptype 2 is used, the coordination vector is put in basis coordinates and the first line in jfile modifies to:

```
1 1 -1 -1 -1 1.359407144 0.866
```

Once again taking SPR-KKR as an example, that corresponds to columns five, six and seven in the exchange parameters outfile. The sixth entry in jfile is the exchange energy in mRy and last entry (not read and optional) is the distance between atoms.

These files together with the inpsd.dat forms the minimal set that is required to run a full ASD or MC simulation. Optionally, there are plenty other external files that may be used for more specific applications and features.

2.1.1 Random alloys and more than one atom in the cell

When the system of interest contains more than one atom in the cell and/or have some chemical disorder, then the setup naturally becomes slightly more complicated. The necessary modifications in the input files are here demonstrated using the system FeCo (found in the examples directory)

with the composition 50-50 an example. We are using two different setups for the system, either using an ordered "supercell" or as a random binary alloy.

FeCo supercell (B2)

An ordered structure of FeCo with 50-50 composition can be represented in the B2 (CsCl) crystal structure which is a simple cubic lattice with two basis atoms. The Fe atoms occupy the corners and Co atoms the center. In the inpsd.dat input file, the Bravais lattice vectors needs to be specified for the simple cubic lattice:

```
cell   1.00000   0.00000   0.00000
       0.00000   1.00000   0.00000
       0.00000   0.00000   1.00000
```

The next step is to specify the basis, i.e. the positions of the Fe and Co atoms. We recommend as in the previous example of Fe, always use a separate file (posfile). Fe and Co occupy two different sites in the cell and are of different atom types, the posfile then takes the form

```
1 1   0.000000  0.000000  0.000000
2 2   0.500000  0.500000  0.500000
```

First line, denotes the Fe atom that is site number 1 and atom type 1 (first and second column) with position 0 0 0 (corner). Second line is the corresponding information for the Co atom that has position 0.5 0.5 0.5 (center in cell). Once the atom type numbers are set in this file, it will carry over the information in the other files which then needs to be consistent. Now when the simulation cell is set up, we need to specify the magnetic moments and then the (exchange) interactions between them. Starting with magnetic moments, the corresponding momfile:

```
1 1 2.7207 0.0 0.0 1.0
2 1 1.7202 0.0 0.0 1.0
```

Once again, the first line specifies the Fe (with site number 1 and chemical type 1) with moment 2.7207 Bohr (from a DFT calculation) and initial moment direction along the z-direction (initmag 3). The second line specifies the same information but for site number 2, i.e. Co that has moment 1.7202 Bohr from calculation. Now both the cell and magnetic moments on each site are specified, what is left to do is the specification of exchange interactions between the moments. From experience, this is the most crucial part in the setup and most easily to get it wrong. The full jfile in the example is longer than specified here (due to the lack of symmetry), here we only show one of the nearest neighbour interactions. We have Fe and Co moments in the cell, a Fe moment could interact with other Fe (Fe-Fe) or with Co (Fe-Co). Vice versa, a Co moment could interact with Fe (Co-Fe) or with other Co (Co-Co). To be complete, we need to specify all the interactions, i.e. Fe-Fe, Fe-Co, Co-Fe and Co-Co interactions. The jfile (using maptype 2) then contains the following blocks

```
1 1  0  0 -1   0.031818272 1.000
1 2  0  0  0   1.839624404 0.866
2 1  0  0  0   1.839624404 0.866
2 2  0  0 -1   0.059966387 1.000
```

Remember that the **types** of atoms that the exchange interactions is valid for, are given in the first two columns of the jfile which specify the **sites** i and j The sites correspond to the information given in the posfile. First line then specifies a Fe-Fe interaction, second line Fe-Co, third line Co-Fe and fourth line Co-Co.

FeCo random alloy

UppASD has the capability to deal with chemical disorder in one or several sublattices of a system. Taking Fe-Co as example, it is natually occuring in the bcc lattice (for Co concentrations less than $\approx 70\%$) with random arrangement of the Fe and Co atoms. Internally within the program, a supercell is created with the target composition set by the user. The required input files needs some modifications that are discussed here. First of all, the flag do_ralloy in the inpsd.dat file needs to set to 1. Then, as ususal, the Bravais lattice needs to be specified and in this case we are using the primitive bcc lattice with its lattice vectors

```
cell         -0.5000000    0.5000000    0.5000000
              0.5000000   -0.5000000    0.5000000
              0.5000000    0.5000000   -0.5000000
```

So far, the setup is not any different from a non-random system. However, the position file looks a bit different. Now we have two chemical types (Fe and Co), each with a certain concentration, that are both situated on the same sublattice

```
1 1 1  0.500   0.000000  0.000000  0.000000
1 1 2  0.500   0.000000  0.000000  0.000000
```

Compare to non-random systems, the posfile now has two additional columns. The third column specify the chemical type (Fe or Co), each with its concentration (fourth column). The concentrations do not need to add up to 1, if smaller then the system becomes diluted with random voids (vacancies) in it. In the present example, Fe (chemical type 1) and Co (chemical type 2) both have 50% concentration. Next, we need to specify the magnetic moments on each sublattice and for each chemical type. The corresponding momfile:

```
1 1 2.4850 0.0 0.0 1.0
1 2 1.7041 0.0 0.0 1.0
```

First column always specifies the site number (same as column 1 in the posfile) and column 2 specifies the chemical type (same as column 3 in the posfile). In the example, the first line corresponds to Fe moment and second line the Co moment. The only remaining part is the specification of exchange interactions. Somewhat similar to the FeCo B2 example, we have four distinct set of exchange interactions (Fe-Fe,Fe-Co,Co-Fe and Co-Co), however in this case all interactions are taking place within the same sublattice. A subset of the jfile (first shell) has the following shape (maptype 2)

```
1 1 1 1 -1 -1 -1 1.970049732 0.866
1 1 1 2 -1 -1 -1 1.947329604 0.866
1 1 2 1 -1 -1 -1 1.947329604 0.866
1 1 2 2 -1 -1 -1 1.238957583 0.866
```

The first and second columns are the same as the jfile for non random systems and specifies the **sites** i and j and thus their corresponding atomic (sublattice) **types**. In this case, we only have one sublattice so it is 1 for all interactions. The third and fourth columns specifies the chemical types of the atoms on that particular sublattice and from top to bottom in this example that means Fe-Fe, Fe-Co,Co-Fe and Co-Co interactions.

2.2 Input Entries

The following entries are currently implemented in UppASD. Where applicable, the default entry setting is underlined.

2.2.1 System parameters

simid The 8 character long simulation id. All output files will include the `simid` as a label.
cell The three lattice vectors describing the cell.
ncell Number of repetitions of the cell in each of the lattice vector directions.
bc Boundary conditions (P=periodic, 0=free).
natoms Number of atoms in one cell. (Not needed if a `posfile` is provided)
ntypes Number of types atoms in one cell. (Not needed if a `posfile` is provided)
momfile External file describing the magnitudes and directions of magnetic moments.
posfile External file for the positions of the atoms in one cell, accompanied with the site number and type of the atom.
posfiletype Flag to change between C=Cartesian or D=direct coordinates in posfile.
set_landeg Flag for assigning different values of the gyromagnetic factor for the moments. Set to 0 by default.

2.2.2 Hamiltonian parameters

exchange External file for Heisenberg exchange couplings on the form

Heisenberg exchange

$$\mathscr{H}_{\mathrm{XC}} = -\sum_{i \neq j} J_{ij} \mathbf{e}_i \cdot \mathbf{e}_j, \tag{2.1}$$

where J_{ij} is the Heisenberg exchange interaction between atoms i and j. For an example of the file, see the description in Sec.2.1.

dm External file for Dzyaloshinskii-Moriya (DM) exchange coupling, which takes the form

Dzyaloshinskii-Moriya (DM) exchange

$$\mathscr{H}_{\mathrm{DM}} = -\sum_{i \neq j} \mathbf{D}_{ij} \cdot (\mathbf{e}_i \times \mathbf{e}_j), \tag{2.2}$$

where $\mathbf{D}_{ij}$ is the DM vector. The format is similar to that of the exchange file, *i.e.* in a 2d square lattice it may look something like:

```
1 1  1.0000  0.0000 0.0000  0.30000  0.00000  0.00000
1 1 -1.0000  0.0000 0.0000 -0.30000 -0.00000 -0.00000
1 1  0.0000  1.0000 0.0000  0.00000  0.30000  0.00000
1 1  0.0000 -1.0000 0.0000 -0.00000 -0.30000 -0.00000
```

The first two entries specify site numbers in the chemical unit cell. The third to fifth entries specify the vector $\mathbf{r}_{ij}$ in terms of the lattice vectors, and the final three entries specify the DM vector $\mathbf{D}_{ij}$.

pd External file for anisotropic symmetric exchange coupling (pd), which takes the form

Anisotropic symmetric exchange

$$\mathscr{H}_{\mathrm{ani}} = -\sum_{i \neq j} \sum_{\alpha,\beta} \mathbf{J}_{ij}^{\alpha\beta} \, m_i^{\alpha} m_j^{\beta}, \tag{2.3}$$

where $\mathbf{J}_{ij}^{\alpha\beta}$ are the pd couplings and indices α and β refer to the x, y, and z-components of the spins. The format is similar to that of the exchange file. An example file for anisotropic symmetric exchange (here for `maptype=1` and `posfiletype=D`) is

```
1   1    0.25   0.00 -0.25   0.00 -0.01   0.00   0.00   0.00   0.00
```

The first two entries indicate the site number and the type of atom, respectively. The third, fourth and fifth entries specify the coordination shell in direct coordinates. The sixth to eleventh entry specify the coupling strength in order J^{xx}, J^{yy}, J^{zz}, J^{xy} $(= J^{yx})$, J^{xz} $(= J^{zx})$, J^{yz} $(= J^{zy})$.

bq External file for biquadratic exchange coupling, which takes the form

Biquadratic exchange

$$\mathscr{H}_{\text{bq}} = -\sum_{i \neq j} B_{ij} \left(\mathbf{e}_i \cdot \mathbf{e}_j\right)^2 . \tag{2.4}$$

The format is identical to that of the `exchange` file discussed above, with the values for the exchange couplings J_{ij} replaced by the biquadratic exchange coupings B_{ij}.

biqdm External file for effective quadratic Dzyaloshinskii-Moriya coupling [1], which takes the form

Quadratic Dzyaloshinskii-Moriya exchange

$$\mathscr{H}_{\text{biqdm}} = -\sum_{i \neq j} F_{ij} \left(\mathbf{e}_i \times \mathbf{e}_j\right)^2 . \tag{2.5}$$

The format is identical to that of the `exchange` file discussed above, with the values for the exchange couplings J_{ij} replaced by the quadratic effective Dzyaloshinskii-Morya exchange couping F_{ij}.

anisotropy External file for anisotropy strengths and directions. The single-ion, or uniaxial, anisotropy is defined as

Uniaxial anisotropy

$$\mathscr{H}^{\text{U}}_{\text{ani}} = \sum_i K^{\text{U}}_1 (\mathbf{e}_i \cdot \mathbf{e}_i)^2 + K^{\text{U}}_2 (\mathbf{e}_i \cdot \mathbf{e}_i)^4, \tag{2.6}$$

where K_1 and K_2 are the strength of the linear and four-fold term along an axis with direction $\mathbf{e}_i$. In a cubic system, one must also define the so-called cubic anisotropy, given by

Cubic anisotropy

$$\mathscr{H}^{\text{C}}_{\text{ani}} = \sum_i K^{\text{C}}_1 (m^2_{i,x} m^2_{i,y} + m^2_{i,y} m^2_{i,z} + m^2_{i,z} m^2_{i,x}) + K^{\text{C}}_2 m^2_{i,x} m^2_{i,y} m^2_{i,z}, \tag{2.7}$$

where $(m_x, m_y, m_z) = \mathbf{m}$. UppASD is able to read in either Eq. (2.6) or Eq. (2.7), or even both. For bcc Fe, a plausible `kfile` might be:

```
1    2    -0.020     0.000     0.0     1.0     0.0     0.1
2    2    -0.020     0.000     0.0     1.0     0.0     0.1
```

The first entry lists the atom number, whereas the second entry indicates if the uniaxial (`1`), cubic (`2`) or both (`7`) anisotropies are to be mounted. The second and third entries list the strength of K_1 and K_2, respectively. The fifth to seventh entries indicate the components of the vector $\mathbf{e}_i$. Finally, in the instance of the second entry being set to 7, the final entry indicates the ratio between $K^{\text{U}}_{\text{ani}}$ and $K^{\text{C}}_{\text{ani}}$.

sym Flag to determine the assumed symmetry of the system (0=none, 1=cubic, 2=2d cubic

[1] For a motivation of this coupling, see Giovannetti *et al.*, Phys. Rev. Lett. **106**, 026401 (2011)

(in *xy* plane), 3=hexagonal).
It is also possible to provide symmetry operations manually. This is done by setting `sym` to 4 and then create an additional input file *sym.mat* containing the number of symmetry operations followed by the operations in matrix form. An example of *sym.mat* for only inversion symmetry can look like:

```
2
 1.0000   0.0000  0.0000
 0.0000   1.0000  0.0000
 1.0000   0.0000  1.0000
-1.0000   0.0000  0.0000
 0.0000  -1.0000  0.0000
 0.0000   0.0000 -1.0000
```

Do not forget the identity operation when using custom symmetry operations. The symmetry operations act on `exchange`, `bq`, and `pd` couplings, but not on `dm` or `biqdm` couplings. Note that the `sym` flag only concerns how the program will treat the exchange couplings, it does thus not have to reflect the proper symmetry of the simulated system. I.e, if the exchange interactions given in `posfile` are not symmetry reduced, then `sym` should be set to 0 even if the system is cubic.

maptype Flag that determines how the coordinates for the different exchange couplings are given. For 1=coordinates the coordinates are given in Cartesian or direct coordinates (see `posfiletype`). For 2 the coordinates of a coupling vector are implicitly given by specifying that the coupling links atom *i* with atom *j* (for an example, see `dm`).

do_jtensor (0=off/1=on). This switch allows the exchange data to be read in according to the tensorial representation of the Heisenberg Hamiltonian, as implemented in the Vienna-Budapest SKKR code. [8] In this case, the exchange Hamiltonian is defined as

Tensorial Heisenberg exchange

$$\mathscr{H}_{\text{Tens}} = \sum_{i,j} \mathbf{e}_i \mathscr{J}_{ij} \mathbf{e}_j. \tag{2.8}$$

Here, $\mathscr{J}_{ij} = -J_{ij}\mathscr{I} + \mathscr{J}^S_{ij} + \mathscr{J}^A_{ij}$ is a 3×3 tensor (in which $\mathscr{I}$ is the unit matrix), the trace of which is equal to the exchange constant as defined in Eq. (2.2.2) by [8]. In this formalism, the anti-symmetric part of the tensor are proportional to the components of the DM vector $\mathbf{D}_{ij}$ in Eq. (2.2), as $D^x_{ij} = \frac{1}{2}(J^{yz}_{ij} - J^{zy}_{ij})$, $D^y_{ij} = \frac{1}{2}(J^{xz}_{ij} - J^{zx}_{ij})$ and $D^z_{ij} = \frac{1}{2}(J^{xy}_{ij} - J^{yx}_{ij})$. In order to define the first shell of exchange parameters in bcc Fe using this formalism, the exchange file would be changed to look as follows:

```
0 0 1 2 0.00134 0.0 0.0 0.0 0.00134 0.0 0.0 0.0 0.00134
0 0 2 1 0.00134 0.0 0.0 0.0 0.00134 0.0 0.0 0.0 0.00134
```

NB: `maptype` must be set to 2 in order to use the tensorial format. In addition, since SKKR prints the exchange in Ry, UppASD reads this input in Ry and not in mRy as usual.

do_prnstruct Flag to print lattice structure (0=off/1=on/2=print only coordinates). Useful for checking if the system geometry and couplings are correctly set up.

2.2.3 General simulation parameters

do_ralloy Flag to set if a random alloy is being simulated (0=off/1=on).

aunits Implement atomic units, *i.e.* set k_B, $\hbar$, ... $= 1$ (Y/N). If this is switched on, the `timestep` in SD mode should be roughly $0.1J_{ij}$.

sdealgh Switch for choosing SDE solver (1=Midpoint, 4=Heun , 5=Depondt-Mertens). The default option runs the semi-implicit midpoint solver developed by Mentink *et al.* [9] In this case, as when using the Depondt-Mertens solver [10], the `timestep` can be as large as 10^{-16} seconds, but this should *always* be checked carefully

mensemble Number of ensembles to simulate. The default value is 1, but this may be increased to improve statistics, especially if investigating laterally confined systems, such as finite clusters or other low-dimensional systems.

tseed Random number seed for the stochastic field simulating the fluctuations due to temperature. Default value is 1.

do_sortcoup Flag to specify if the arrays of couplings should be sorted or not (Y=yes, N=no). Of importance for sampling of polarization. Could be very slow if long range interactions.

2.2.4 Initialization parameters

initmag Switch for setting up the initial configuration of the magnetic moments (1=Random distrubution, 2=Cone, 3=aligned along direction defined in momfile, 4=Read from restartfile).

restartfile External file containing stored snapshot from previous simulation (used when initmag=4). The format coincides with the format of the output file `restart.simid.out`.

mseed Random number seed for magnetic moments if initmag=1. Set to 1 by default.

theta0 If `initmag`=2, the magnetic moments are randomly distributed in a cone prescribed by this angle, and `phi0`. Set to 0 by default.

phi0 Cone angle for initmag=2. Set to 0 by default.

roteul Perform global rotation of magnetization. Set to 0 by default.

rotang Euler angles describing the rotation if roteul=1.

initexc Perform initial excitation of the spin system (N=none, I=Vacancies, R=Two magnon Raman scattering).

initconc Concentration of vacancies or two magnon spin scattering.

initneigh Neighbour index referring to the list of neighbours for Heisenberg exchange. Determines which spins to swap in two magnon spin scattering.

2.2.5 Initial phase parameters

ip_mode Mode for initial phase run (S=SD, M=Monte Carlo, H=Heat bath Monte Carlo, N=none).

ip_temp Temperature for initial phase run if Monte Carlo (ip_mode=M or H).

ip_hfield External applied field (in units of Tesla) for initial phase run. This is given in Cartesian coordinates, *e.g.*,

```
hfield   1.0   0.0   0.0
```

ip_mcnstep Number of Monte Carlo sweeps (MCS) over the system if ip_mode=M or H.

ip_nphase Number of initial phases to be done with SD. This must be followed by `ip_nphase` lines containing number of steps, temperature, timestep and damping for each phase. An example (for an initialization with the temperature decreasing from 300 K to 10 K) can look like:

```
ip_nphase 3
20000 300.0 1.0d-16  0.1
20000 100.0 1.0d-16  0.1
```

```
30000 010.0 1.0d-16  0.1
```

ip_mcanneal Number of initial phases to be done with MC. This must be followed by `ip_mcanneal` lines containing number of steps and temperature for each phase. An example (for an initialization with the temperature decreasing from 300 K to 10 K) can look like:

```
ip_mcanneal 3
20000 300.0
20000 100.0
30000 010.0
```

2.2.6 Measurement phase parameters

mode Mode for measurement phase run (S=SD, M=Monte Carlo, H=Heat bath Monte Carlo).

temp Temperature for measurement phase.

hfield External applied field (in units of Tesla) for measurement phase.

mcnstep Number of Monte Carlo sweeps (MCS) over the system if mode=M or H.

damping Damping parameter α for SD measurement phase. Default value is 0.05.

timestep Time step between SD iterations. Unless `aunits`=Y, this should typically be set to a value between 10^{-17} and 10^{-15} seconds, depending on the system and SDE solver.

set_bpulse Add magnetic field pulse (0=no, $1-4$ for different shapes)

2.2.7 Parameters for measuring of observables

Typically the measurement of each observable is controlled by two parameters in a combination as follows; `do_observable` that enables the measurement and `observable_step` that determines the frequency of the measurements. Here the `observable` should be replaced by the internal name of the wanted quantity i.e. `do_avrg` and `avrg_step` for the average magnetization.

plotenergy Flag to enable the calculation of the energy of the system projected to the different components of the Hamiltonian. (0=off/1=on).

do_avrg Sample and print average magnetization, and its higher order moments. (Y/N).

do_proj_avrg Sample and print type (*i.e.* sublattice) projected average moments. (Y/N/A).

do_projch_avrg Sample and print chemical (*i.e.* sublattice) projected average moments. (Y/N/A).

avrg_step Number of time steps between sampling of averages. Set to 100 by default.

do_cumu Sample cumulants (Y/N). Automatically enabled for Monte Carlo simulations.

cumu_step Number of time steps between sampling of cumulants. Set to 25 by default.

do_tottraj Sample and print all trajectories (moments) in the system. (Y/N). Generates the (rather large) `moments.simid.out` file.

tottraj_step Number of time steps between samplings of moments. Set to 1000 by default.

ntraj Number of individual trajectories to sample and print. Followed by `ntraj` lines describing atoms to sample, time step between samples and steps to buffer. Set to 0 by default.

do_pol Sample and print average ferroelectric polarization (Y/N) according to the expression $P \propto \gamma \sum_{i,j} \hat{\mathbf{e}}_{ij} \times (\mathbf{m}_i \times \mathbf{m}_j)$. Uses the neighbour lists set up for exchange but here the sum is performed up to a threshold `max_pol_nn`. For this construction to work, it is important to set the flag `do_sortcoup`=N.

max_pol_nn Number of neighbours to use when evaluating the polarization.

pol_step Number of time steps between sampling of polarization averages. Set to 100 by default.

do_stiffness Calculation of spin-wave stiffness (and tensor) and micromagnetic exchange constant (Y=yes, N=no).

eta_min Lowest value of auxiliary convergence parameter in stiffness calculation (recommended around 6-8)

eta_max Largest value of auxiliary convergence parameter in stiffness calculation (recommended around 10-12)

alat Lattice constant (in m) for calculation of exchange stiffness

Parameters for measuring of correlation functions

do_sc Flag to determine if spin correlations should be analysed (Q=S($\mathbf{q}$,ω), N=no, C=G(r)). Setting this flag to Q or C measures space- and time-displaced correlation functions. The spatial time dependent correlation function $C(\mathbf{r},t)$ is defined as

The spatial time dependent correlation function $C(\mathbf{r},t)$

$$C^k(\mathbf{r}-\mathbf{r}',t) = \langle m^k_{\mathbf{r}}(t) m^k_{\mathbf{r}'}(0)\rangle - \langle m^k_{\mathbf{r}}(t)\rangle\langle m^k_{\mathbf{r}'}(0)\rangle, \tag{2.9}$$

where the angular brackets signify an ensemble average and k the Cartesian component. The dynamical structure factor is then obtained by Fourier transforming $C(\mathbf{r},t)$ as

The dynamical structure factor $S(\mathbf{q},\omega)$

$$S^k(\mathbf{q},\omega) = \frac{1}{\sqrt{2\pi}N}\sum_{\mathbf{r},\mathbf{r}'} e^{i\mathbf{q}\cdot(\mathbf{r}-\mathbf{r}')}\int_{-\infty}^{\infty} e^{i\omega t} C^k(\mathbf{r}-\mathbf{r}',t)dt, \tag{2.10}$$

and this function describes the energy dispersion for excited spin waves present in the simulated system. [11] . If the flag is set to C, the static correlation function G(r) and its Fourier transform S(q) are measured. By locating the maximum of S(q), the ordering vector of the simulated system can be determined. In this case it is important to have a `qfile` that includes $\mathbf{q}$-vectors in the whole Brillouin zone.
In order to obtain a useful $S(\mathbf{q},\omega)$ measurement, it is important to understand the sampling of the function that is determined by `sc_nstep`, `sc_step`, and `timestep`.

do_sc_proj Flag to determine if type projected spin correlation should be analyzed (Q=yes, C=G(r),N=no).

do_sc_projch Flag to determine if chemical type projected spin correlation should be analyzed of random alloys (Q=yes, C=G(r),N=no).

do_qt_traj Flag to determine if the time evolution of the equal time spin correlation $S(\mathbf{q})$ should be written to file (Y=yes,N=no). This works only if `do_sc`=C. The function $S(\mathbf{q})$ is sampled every `sc_sep` time step and can give insight in the phase transitions in systems with more than one magnetic order parameter. Suggested use is to first determine the magnetic phase diagram and the associated ordering vectors by sampling $S(\mathbf{q})$ (as described above). The order parameters can then be specified in a `qpoints` file and followed in simulations where the systems is driven out of equilibrium by an external perturbation in form of an applied magnetic field, a heat pulse or a two-magnon Raman scattering excitation.

sc_nstep Number of steps to sample. This number sets the resolution of time/frequency based correlation functions by deciding the number of measured times/frequencies to include in the calculation.

sc_step Number of time steps between each sampling. This number determines the time/frequency range over which correlation functions are measured. The mininum sample time is given by `timestep`*`sc_step` and the maximal sampling time is then deter-

minded by `sc_nstep*timestep*sc_step`. The minimal/maximal frequencies are then determined by the inverse of the maximal/minimal sampling time.

sc_sep Number of time steps between the start of subsequent spin correlation measurements.

qpoints Flag for for generation of q-point mesh necessary for $S(q,\omega)$ calculations. (F=external file with carteisan coordinates, A=automatic, C=full cell, P=extended plane spanned by the first and third reciprocal lattice vector, D=external file with direct coordinates).

qfile External file containing the q point mesh for $S(q,\omega)$ calculations.

sc_window_fun Choice of windowing function for the Fourier transforms used in $S(q,\omega)$ calculations (1=box, 2=Hann, 3=Hamming, 4=Blackman-Harris).

do_sc_local_axis Modify the sampling for $S(q,\omega)$ so that $S^{\perp}$ and $S^{\parallel}$ are sampled instead of S^x, S^y, S^z. This normally improves the simulated spectra for ferromagnets but should be used with care since it can, if misused, suppress low-level excitations. (Y,N)

sc_local_axis_mix Determines the rate of updating the local quantization axis used when `do_sc_local_axis=Y`. Values larger than zero can be useful if there are unwanted fluctuations such as global rotations of the whole systems, which can happen for in particular for finite systems such as clusters.

do_ams Spin wave dispersion from the Fourier transform av the exchange interactions, so-called Adiabatic Magnon Spectra (AMS) (Y=yes, N=no). This version only handles AMS in collinear magnetic structures but it is very fast and can therefore be a good option for comparison with the full dynamical spectra. If `do_ams=Y` then one must provide a qfile just as in the case of $S(q,\omega)$.

do_magdos Magnon density of states (MDOS) from AMS (Y=yes, N=no, A=read from file).

magdos_freq Number of frequencies in MDOS calculation from AMS. Around 200 is recommended.

magdos_sigma Gaussian broadening (in meV) in MDOS calculation from AMS (around 30 is recommended).

do_autocorr Flag to enable autocorrelation sampling (Y=yes, N=no).

acfile External file containing waiting times for the autocorrelation measurements.

2.3 Output files

Depending on the settings chosen in the input file, UppASD prints out a varying number of output files. These all share the suffix `.simid.out` where `simid` is the simulation handle defined in the input file.

2.3.1 Simulation and Hamiltonian output

aniso1.simid.out

Is written if the anisotropy is defined. Prints the anisotropy parameters for each atom in the format

```
aniso1.simid.out:    e_x, e_y, e_z, k_1, k_2
```

where the three first entries are the direction of the anisotropy axis.

biqdmdata.simid.out

Is written if the effective quadratic DM interaction is defined. Prints out the effective quadratic DM coupling for each atom.

bqdata.simid.out

Is written if the bq interaction is defined. Prints out the bq coupling for each atom.

coord.simid.out

Is written if `do_prnstruct` is switched on. Prints out the coordinates of each moment in the system.

dmdata.simid.out

Is written if the DM interaction is defined. Prints out the DM coupling for each atom.

dmstruct.simid.out

Is written if the DM interaction is defined and `do_prnstruct` is switched on. Prints out the coupling list for the DM vector of the system. Similar to the data presented in `struct.simid.out`.

inp.simid.out

Extensive output of the values assigned to global variables after reading `inpsd.dat` and accompanying files.

pddata.simid.out

Is written if anisotropic exchange interaction pd interaction is defined. Prints out the effective pd couplings for each atom.

struct1.simid.out

Is written if `do_prnstruct` is switched on. Prints out the neighbour coupling list for the exchange couplings of the system. Handy for checking if the system is set up correctly. *Warning* this file might be very large for a realistic system, be mindful of that.

struct.simid.out

Is written if `do_prnstruct` is switched on. Prints out the Cartesian coordinates of the exchange couplings. The entries are grouped into coordination shells. Handy for checking if the system is set up correctly. *Warning* this file might be very large for a realistic system, be mindful of that.

2.3.2 Measured observables

averages.simid.out

Is written if measurement phase is run in SD mode. Prints out the average magnetization as a function of simulation time, in the format

> **averages.simid.out:** `step` , m_x , m_y , m_z , m , σ`(m)`

where *step* is the simulation time expressed in terms of the number of time steps, m_x, m_y and m_z are the components of the intensive average magnetization (*i.e.*, $m_x = \frac{1}{N}\sum_i m_{x,i}$), $m = \sqrt{m_x^2 + m_y^2 + m_z^2}$, and so on. $\sigma(m)$ is the standard deviation of m when the number of ensembles is larger than one.

cumulants.simid.out

Prints out the running time averages of the intensive magnetization and its higher order moments, in the format

> **cumulants.simid.out:** `step` , $< M >$, $< M >^2$, $< M >^4$, U_4 , χ , C_v

where, brackets denote time averaged quantities and $U_4 = 1 - \frac{1<M>^4}{3<M>^2}$ is the fourth order 'Binder' cumulant, useful for estimating transition temperatures [7], χ is the magnetic susceptibility, and C_V is the heat capacity.

moments.simid.out

Is written if the `do_tottraj` flag is switched on. Prints the configuration of all magnetic moments at regular interval in time in the format

> **moments.simid.out:** `step , atom ,` $m_{x,i}$, $m_{y,i}$, $m_{z,i}$, m_i

Note that this file is very large. It is useful for creating animations of the evolution in time of the magnetic configuration of the system. Printed for ensemble nr 1.

polarization.simid.out

Prints out the average ferroelectric polarization as a function of simulation time, in the format

> **polarization.simid.out:** `step ,` p_x , p_y , p_z , p , $\sigma(p)$

where *step* is the simulation time expressed in terms of the number of time steps, p_x, p_y and p_z are the components of the intensive average polarization (*i.e.*, $p_x = \frac{1}{N}\sum_i p_{x,i}$,...,) and $p = \sqrt{p_x^2 + p_y^2 + p_z^2}$. $\sigma(p)$ is the standard deviation of p when the number of ensembles is larger than one.

projavs.simid.out

Is written of the `do_proj_avrg` flag is switched on. Prints out the same thermodynamic averages printed in `averages.simid.out`, but projected to each atom type sublattice. The format is also identical to `averages.simid.out`, except for the addition of a column indicating the sublattice.

restart.simid.out

The magnetic configuration of the system at a specific point in time. Can be used as input when the `initmag` flag is set to 4.

trajectory.simid.out

The trajectory as a function of time step for an individual magnetic moment on format

> **trajectory.simid.out:** `step , atom nr ,` $m_{x,i}$, $m_{y,i}$, $m_{z,i}$, m_m , m_i

If the number `ntraj` is also defined to be greater than 1, the code prints out ntraj files named `trajectory.simid.ntraj.ensemblnr.out`.

sq.simid.out

Is written if the `do_sc` flag is C. Prints out the static correlation function in reciprocal space $S(q)$ in the format

> **sq.simid.out:** `nq` , q_x , q_y , q_z , $S^x(\mathbf{q})$, $S^y(\mathbf{q})$, $S^z(\mathbf{q})$, $S(\mathbf{q})$

sqt0.simid.out

Is written if the `do_sc` flag is C and the `do_qt_traj` flag is Y. Prints out the trajectory in time of the equal time correlation function $S(q)$ in the format

sqt0.simid.out: `step` , `nq` , q_x , q_y , q_z , $S^x(\mathbf{q})$, $S^y(\mathbf{q})$, $S^z(\mathbf{q})$, $S(\mathbf{q})$

sra.simid.out

Is written if the `do_sc` flag is C. Prints out the static correlation function in real space $S(r)$ in the format

sra.simid.out: $|r|$, $S^x(r)$, $S^y(r)$, $S^z(r)$, $S(r)$

sqt.simid.out

Is written of the `do_sc` flag is switched on. Prints out the time-resolved structure factor $S(q,t)$ in the format

sqt.simid.out: `nstep` , `nq` , $\mathrm{Re}[S^x(\mathbf{q},t)]$,
$\mathrm{Im}[S^x(\mathbf{q},t)]$, $\mathrm{Re}[S^y(\mathbf{q},t)]$, $\mathrm{Im}[S^y(\mathbf{q},t)]$, $\mathrm{Re}[S^z(\mathbf{q},t)]$, $\mathrm{Im}[S^z(\mathbf{q},t)]$

projsqt.simid.out

Is written of the `do_sc_proj` flag is switched on. Prints out the same information printed in `sqt.simid.out`, but projected to each atom type present in the system.

sqw.simid.out

Is written of the `do_sc` flag is switched on. Prints out the frequency-resolved structure factor $S(q,\omega)$ in the format

sqw.simid.out:
`nq` , q_x , q_y , q_z , `nstep` , $S^x(\mathbf{q},\omega)$, $S^y(\mathbf{q},\omega)$, $S^z(\mathbf{q},\omega)$, $S(\mathbf{q},\omega)$

projsqw.simid.out

Is written of the `do_sc_proj` flag is switched on. Prints out the same information printed in `sqw.simid.out`, but projected to each atom type present in the system.

swdos.simid.out

Is written of the `do_sc` flag is switched on. Prints out the $S(q,\omega)$ 'density of states' as a function of energy.

totenergy.simid.out

Is written if the `plotenergy` flag is switched on. Prints out the total energy of the system as a function of time step.

3. Tutorial and walkthrough

Some typical examples of simulations using the UppASD software is discussed here in rather great detail with step-to-step tutorial.

3.1 Dynamics of a single spin

When treating magnetization dynamics the first example that comes to mind is an isolated spin interacting with a constant effective magnetic field. If the magnetic moment is not parallel to the field this will exert a torque and cause it to move. Due to the fact that in reality there are several channels of dissipation, the spin does not precess forever along the direction of the field, if not it is damped, and after some time it reaches the equilibrium position, i.e. aligned parallel to the effective field.

Situations such as this are at the core of spin dynamics simulations, that is one has a spin (or a series of them) and through some external perturbation, the system is driven out of equilibrium, causing it to evolve as given by the LLG equation 1, reaching the equilibrium position (at $T = 0\,\mathrm{K}$) after a certain time which is determined by the Gilbert damping parameter, α.

In general when one is interested in seeing the relaxation dynamics of a given process the procedure is the following:

1. Run a spin dynamics simulation (`mode S`) with a suitable algorithm e.g. midpoint (`SDEAlgh 1`) for a given initial condition.
2. Collect the results and analyze the trajectories, averages and/or other dynamical quantities.
3. Determine the quantity of interest, e.g. relaxation time.

For the purpouses of this walkthrough, the input files for the single spin mode will be used, these files are found in the example directory under the name SingleSpin.

First one must go to the root directory of the example:

```
cd examples_revision_controlled/SingleSpin
```

In this directory one can see that there is a folder called `BASE` as well as several bash scripts. For the moment lets ingnore the scripts and lets explore the input files, for this enter the `BASE` folder

```
cd BASE
```

In this folder there is a selection of files, the `inpsd.dat` file which is the main input file for UppASD, the files `momfile` for the magnitude and direction of the moments, `posfile` for the position of the magnetic moments in the unit cell, the `kfile` for the magnetocrystalline anisotropy and the `jfile` for the exchange interactions present in the system.

Openning the `kfile` one can see that there is an uniaxial anisotropy with an eaxy axis along the z-direction, this is used to simulate the constant field of the example (a similar behaviour can be obtained with an external magnetic field, but for this example the anisotropy is used). One can then examine the exchange interactions in the system, by accessing the `jfile` in this tutorial, is the fact that the exchange energy is set to zero, the reason behind this is that one wishes to model either a single spin i.e. it does not interact with any other magnetic moments, or a macrospin, i.e. a large magnetic moment composed of many spins but whose internal dynamics are frozen.

Now, the relaxation dynamics of a magnetic moment are intrinsically dependent on the fields acting over it and the damping parameter. In this tutorial the field is fixed, so the *damping* is the quantity of interest, hence, one must open the `inpsd.dat` to see which value it has.

In the `inpsd.dat` one can see that a given initial angle has been given to the magnetic moments by using a global rotation `roteul 1` such that the spin and the easy axis have a 30 deg angle between them. The magnitude of the damping is set to a dummy variable `GILBERT`, and the temperature has also been set to a dummy called `TEMPE`, the actual magnitudes of these parameters will be set by the scripts.

Simulations with bash scripts

To perform the actual calculations, one needs to go back to the parent directory and then first run the script `SWEEP_DAMP.sh`

```
cd ../
./SWEEP_DAMP.sh
```

This script will run the ASD simulation for several values of the damping, which will be printed in the screen and will store all the files in folders named `DAMP{value of damping}`, while setting the temperature to $T = 0\,\mathrm{K}$.

After having run the `SWEEP_DAMP.sh` file one can plot the relevant information using the `plot.gnu` script which will create a plot using `gnuplot` (these plots can be produced with other plotting programs making use of the `averages.*.out` files). To run the plotting one just needs to do

```
gnuplot plot.gnu
```

The script will produce a file named `GilbertTotal.png`, openning this file one can see the following

As can be seen in this plot one has the time evolution of the different components of the magnetic moments. In general in the limit of low dampign $\alpha < 1$ one can see that as the damping increases the system reaches the equiblibrium faster. It is of special interest to also look at the limit $\alpha = 0$, where one can see that without damping the spin will prescess around the effective field without ever aligning itself with it, this is evident by looking at the time evolution of M_z. However, in the high damping limit $\alpha > 1$ as the damping increases the magnetic moment actually slows down, this is of great importance and it is the reason why Gilbert introduced the damping in a different way than Laundau and Lifshitz originally did, as the damping acts as a viscous force, and in the infite damping limit the spin whould not be able to move.

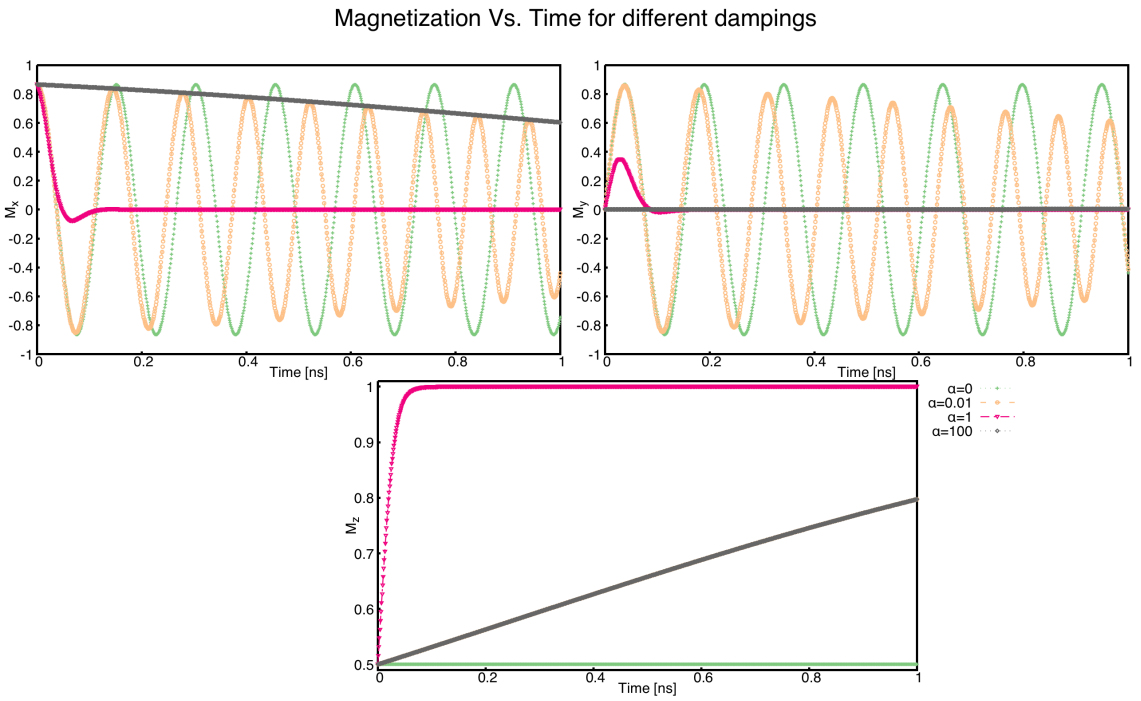


Figure 3.1: Time evolution of the different components of a magnetic moment interacting with the field created by an uniaxial anisotropy when different damping values are considered.

Simulations without bash scripts

If you do not have access to `bash` then you can try the effect of the damping parameter by creating four subdirectories, copying the contents of the `BASE` directory there. In each subdirectory you can then edit the `inpsd.dat` file manually and changing the `GILBERT` value to 0.0, 0.01, 1.0, and 100. You also need to set the temperature by changing the `TEMPE` value to 0.0. You can then run the UppASD binary in each of the different directories and plot the columns 1 and 4 in `averages.SingleSP.out` using your plotting program of choice. The outcome should be equivalent to what was obtained in Fig. 3.1.

3.1.1 Switching under external magnetic field

Now that the effect of the damping on the dynamics is understood, another aspect of great relevance for magnetization dynamics can be studied, namely the switching of the magnetic moment under the influence of an external field. For this one can run the script `SWEEP_FIELD.sh`

```
./SWEEP_FIELD.sh
```

This will create a series of folders named `FIELD{value of the field}`, in each of these folders the temperature is set to $T = 0\,\mathrm{K}$, and the damping is set to $\alpha = 0.1$. To observe the switching an external magnetic field is added in the following way

```
hfield    0.00   0.00   -hfield
```

that is, the external field is along the negative z-direction which will extert a torque over the magnetic field.

To study the effect that the external magnetic field has over the dynamics of the system lets use the script `plotB.gnu`

```
gnuplot plotB.gnu
```

this produces a plot named `Switching.png` in it one can see the time evolution of the M_z component for different magnitudes of the external magnetic field. In it one can see that after a certain critical value of the external magnetic field, the magnetic moment switches, going from tending to align itself to the possitive z-direction to the negative z-direction, i.e. using an external magnetic field one can switch the direction of the magnetic moment.

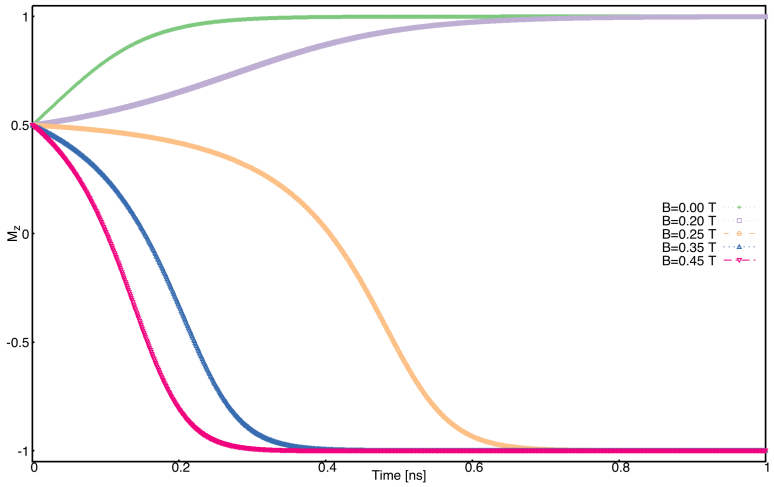


Figure 3.2: Time evolution of the M_z component of the magnetic moment under the influence of an external magnetic field of different magnitudes.

Simulations without bash scripts

If you do not have access to `bash` then you can try the effect of the external field by creating multiple subdirectories, copying the contents of the `BASE` directory there, and changing the `GILBERT` value to 0.1 in the `inpsd.dat` file manually. You also need to set the temperature by changing the `TEMPE` value to 0.0. To add an external field, you add the line `hfield 0.0 0.0 -HFIELD` where you can put `HFIELD` to 0.00, 0.20, 0.25, 0.35, and 0.45 in the different directories. You can then run the UppASD binary in each of the different directories and plot the columns 1 and 4 in `averages.SingleSP.out` using your plotting program of choice. The outcome should be equivalent to what was obtained in Fig. 3.2.

3.1.2 Thermal effects

Temperature is known to have great importance on the magnetic properties of materials, and henceforth in the dynamics of the magnetic moments. In this section of the tutorial the switching of a magnetic moment thanks to thermal fluctuations. This can be done by running the `SWEEP_TEMP.sh` script.

```
./SWEEP_TEMP.sh
```

The simulations that will be perfomed by this script are very similar to the previous ones, in this case one has a spin aligned along the positive z-direction with a magnetic easy axis along the z-direction. The damping for all these examples is set to $\alpha = 0.01$ and the temperature is varied. An analysis of some of the key features of the influence of thermal fluctuations can be observed by using the plotting script `plotTemp.gnu`, this script will generate a couple of plots. Hence one must run the script

```
gnuplot plotTemp.gnu
```

First lets study the one called `singleTemp.png`, an example of this can be seen in Fig. 3.3, in which one can see how the spin tends to fluctuate between two orientations ($+z$ and $-z$), this is due to the uniaxial anisotropy present in the system in conjuction with the thermal fluctuations, which allow the spin to overcome the energy barrier given by the anisotropy.

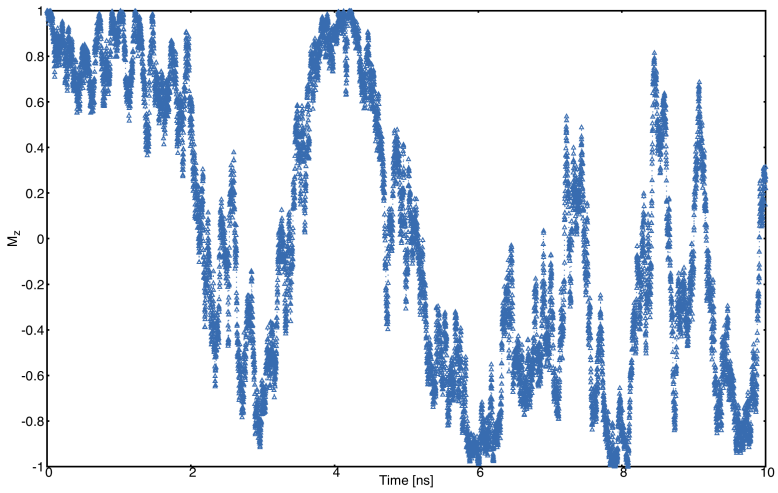


Figure 3.3: Time evolution of the M_z component of a magnetic moment at $T = 4\,\mathrm{K}$.

The next plot named `relaxation.png` is an average over 200 replicas (`Mensembe 200`) of the simulation which are then averaged over. In here one can see instead the average relaxation time of the spin, showing how it can vary with temperature, thus showcansing the importance of

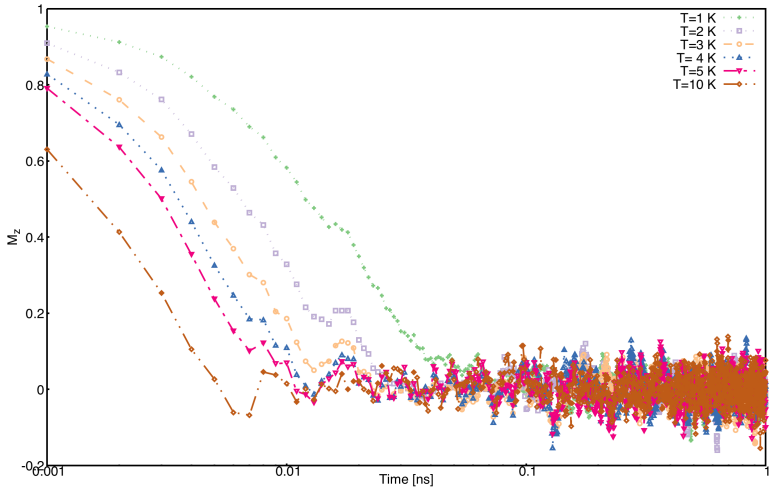


Figure 3.4: Average time evolution of the M_z component of a magnetic moment for several temperatures.

Simulations without bash scripts

If you do not have access to `bash` then you can try the effect of the temperature by creating a subdirectory, copying the contents of the `BASE` directory there, and changing the `GILBERT` value to 0.1 in

the `inpsd.dat` file manually. You now need to set the temperature by changing the `TEMPE` value in the different directories. A suggested temperature values is 4 (K). You can then run the UppASD binary in each of the subdirectory and plot the columns 1 and 5 in `averages.SingleSP.out` using your plotting program of choice. The outcome should be equivalent to what was obtained in Fig. 3.3. To see the effect of statistics, you can take the same simulation, but add the line `Mensemble 200` at the end of `inpsd.dat`. Remove all output files and rerun the UppASD binary. If you now plot plot the columns 1 and 5 in `averages.SingleSP.out` , the outcome should be equivalent to what was obtained in Fig. 3.4.

3.2 Determination of T_c of a ferromagnetic material

When one talks about a ferromagnetic material, one refers to a class of substances that have as a ground state ($T = 0\,\mathrm{K}$) all its spins aligned in the same direction. As temperature increases disorder enters to the system until at a certain critical (Curie) temperature T_c the system has no net magnetization. The Curie temperature could be seen as the highest possible working temperature for the material and is of great importance for applications.

In this walkthrough we will use Monte Carlo simulations to determine the magnetization as function of temperature and from that the T_c. Overall, the procedure is as follows:

1. Run Monte Carlo simulations using either Metropolis algorithm (`mode M`) or Heat bath algorithm (`mode H`) for a set of different temperatures.
2. Collect the results and analyse the temperature dependent magnetization and/or other thermodynamic properties
3. Determine T_c

For this step-by-step walkthrough, we will use the input files for Fe as example that is found in the example directory and displayed in Chapter 2.

First of all, standing in the root directory, navigate to the input files of Fe:

```
cd examples_revision_controlled/Fe
```

The program works in such a way that we need to repeat simulations for different temperatures. For the purpose, it is recommended to use a script that does simplify a lot of things but first we need to prepare the input files for that. We will create a "Base" directory where the raw input files are stored.

```
mkdir Base ; mv * Base/
```

Next, we need to copy two run-scripts (runme.sh and printM.sh) that will run a set of simulations at different temperatures from the scripts-directory:

```
cp ../scripts/runme.sh . ; cp ../scripts/printM.sh .
```

Before starting the run-scripts, the inpsd.dat file needs to be prepared in such a way that the temperature in both initial and measurement phase are set to "TEMP". Using any editor of choice, change temperature from "300" to "TEMP" of lines 25 and 28. With that, everything is prepared to run the simulations using the run-script:

```
./runme.sh
```

Depending on computer system, it may take some time to run through all temperatures. After all calculations are finished, a new directory for each temperature has been created with output files for that particular temperature. To create a summary, we are using the other script (printM.sh) that reads certain information in output files and collect them in a single file (thermal.dat) using the command:

```
./printM.sh
```

Let's investigate the generated thermal.dat file.

```
# Temp.   Mavg     UBinder    Susc.      Cv
    10  2.237941  0.666667  0.000007  1.021710
   100  2.164441  0.666664  0.000071  0.976500
   200  2.078111  0.666654  0.000166  1.024818
   300  1.986299  0.666631  0.000275  1.048131
   400  1.886474  0.666586  0.000422  1.079090
   500  1.777491  0.666515  0.000563  1.172849
   600  1.650539  0.666357  0.000827  1.277333
   700  1.504614  0.666068  0.001138  1.492052
   800  1.308315  0.665102  0.001981  1.770288
   900  1.007354  0.658843  0.005374  2.151073
   950  0.749161  0.635858  0.012374  2.260679
  1000  0.415532  0.542287  0.015266  1.487550
  1050  0.267658  0.473693  0.008910  1.004156
  1100  0.204595  0.442776  0.005369  0.784146
  1150  0.171373  0.429213  0.003673  0.670025
  1200  0.151207  0.444319  0.002641  0.556371
  1250  0.137225  0.455091  0.002093  0.471770
  1300  0.128381  0.431053  0.001813  0.418473
  1500  0.102811  0.464573  0.000898  0.283877
```

The contents of the file are as follows: first column list the temperature, magnetization (in Bohr) in the second column, the Binder cumulant (see below) in the third column, the susceptibility are found in the fourth columns and the fifth column contains the specific heat (in units of k_B). If the magnetization (col 1) is plotted against the temperature (col 1) we immediately see that T_c is around 1000 K. That is also reflected in the peak of the susceptibility (col 4) and the specific heat (col 5) around that temperature. However, in order to obtain a more precise value of T_c, the cumulant crossing method that originally was suggested by Binder is very powerful and useful technique. The (4'th order) cumulant U_L , defined as:

$$U_L = 1 - \frac{\langle M^4 \rangle}{3\langle M^2 \rangle^2} \tag{3.1}$$

has unique properties that makes it easy to locate T_c without resorting to advanced finite size scaling analysis or calculation of critical exponents of the transition. As the system approaching infinite size, $U_4 \to 4/9$ for $T > T_c$ and $U_4 \to 2/3$ for $T < T_c$. However, the crucial part is for large enough systems, the curves of U_4 for different lattice sizes cross in a fixed point U^* and the location of the fixed point is T_c. Practically, that means that all simulations are repeated using some other lattice size of the simulation box. As an example, change the ncell keyword in the inpsd.dat file (line 2) in the Base directory from using a cube of size 12 to 20 (i.e 20 20 20) and repeat the calculations once again using the runme.sh script. Since the simulation box is now larger, it will take longer time to finish (with all other parameters fixed, the simulation time scales linearly with number of atoms in the cell, i.e. $(20/12)^3 \approx 4.6$ longer execution time than previous). Once finished, gather and collect the output using the printM.sh script. Now plot the cumulant as function of temperature and one get a figure similar to as shown in Fig. 3.5. There are some statistical noise at high temperatures which can be reduced by running the simulations with more steps and more ensembles. Nevertheless, there is a distinct crossing around 955 K which is the calculated T_c for the specific system.

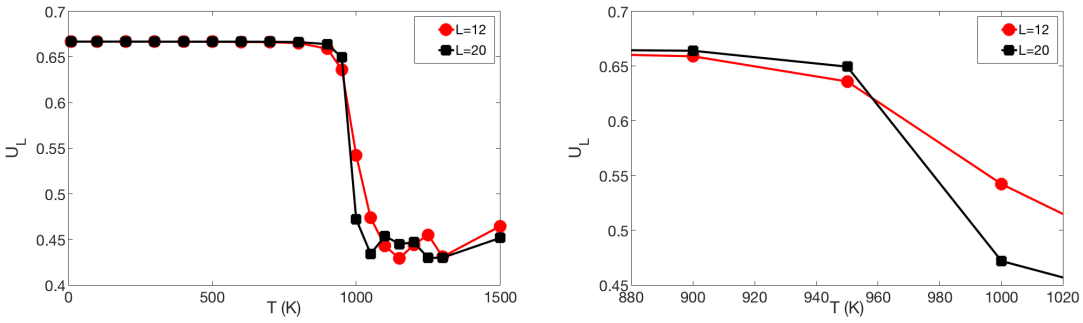


Figure 3.5: (left) Cumulant of Fe as function of temperature using cubic simulation box of size L=12 and L=20. (right) Zoom in of the crossing point and the location of T_c.

Simulations without bash scripts

If you do not have access to `bash` then you can still try the example above by entering the `examples_revision_controlled/Fe` directory and there create multiple subdirectories by hand. You can then copy the files in the base (`Fe`) directory to the subdirectories and in each of you directories you can then edit `inpsd.dat` by changing the temperature from the default value of 300 to a number of your choice (suggestion: 100, 500, 800; 900;1000). The temperature is controlled both by the keyword `temp` and by the second number in the line below the `ip_mcanneal 1` line. After running the UppASD executable in each of the directories you can then compare your outcomes with that of Fig. 3.5.

3.3 Dynamical correlations and magnon spectra

A very useful functionality of UppASD is the possibility to simulate the dispersion relations for magnons. This is done by sampling the dynamical structure factor $S(\mathbf{q},\omega)$, as introduced in Sec.2.2.7. The simulated magnon dispersion relations, or magnon spectrum, can straightforwardly be compared with experimental inelastic scattering measurements as a benchmark of the theoretical model.

3.3.1 Ferromagnetic magnons

In this tutorial we will show how to obtain the magnon spectra from ASD simulations of $S(\mathbf{q},\omega)$ as well as from linear spin wave theory, which gives the zero-temperature adiabatic magnon spectra of the same Hamiltonian used for the ASD simulations. The tutorial uses the simple example of a 1d-Heisenberg spin chain found among the examples provided with the UppASD distribution.

```
cd examples_revision_controlled/HeisChain
```

There, the input file `inpsd.dat` can be inspected and the following lines controls the sampling of the correlation function.

```
do_sc Q
sc_window_fun  2
do_sc_local_axis N
sc_local_axis_mix  0.0
```

```
sc_nstep 2000
sc_step 10

qpoints F
qfile   ./qfile
```

Checking the given parameters with their description in Seq. 2.2.7, we see that here the sampling will be performed with a Hann windowing function, without transforming the system to a local reference frame and with q-points given by the external file `qfile`. Simulating the system as-is gives a magnon spectrum that looks as the left panel of Fig.3.6. In order to visualize the magnon spectrum, scripts are provided both for `gnuplot` and for `MATLAB/Octave`. The plot in Fig.3.6 was obtained by running the script `Sqw/sqw_map.sh`. In order to get a feeling of which parameters that

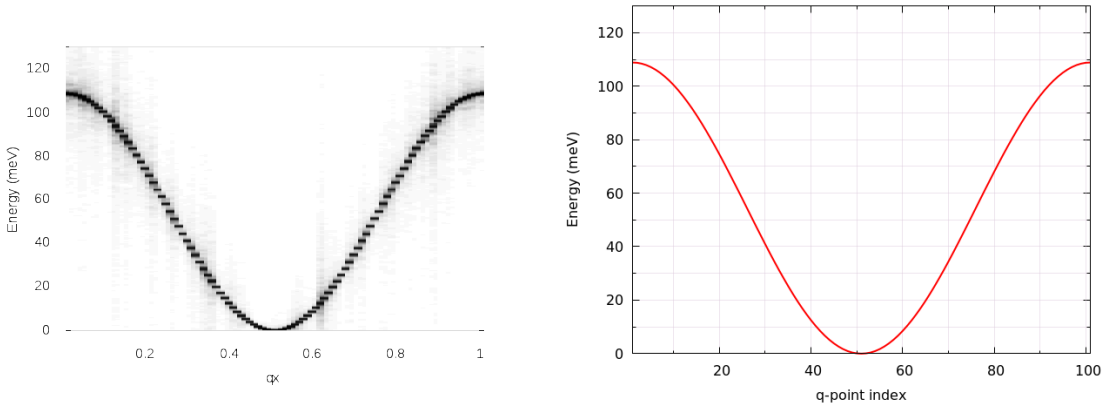


Figure 3.6: (Left) Simulated magnon spectrum for a ferromagnetic Heisenberg spin chain. (Right) Adiabatic magnon spectum for the same system.

determine the range and quality of the simulated spectra, it is recommended to change the values of `sc_step`, `sc_nstep`, `sc_window_fun`, and `do_sc_local_axis`. Other general simulation parameters that also affects the $S(\mathbf{q},\omega)$ are also `timestep` and `damping`. Since the simulation window (in the frequency domain) is determined by `timestep`, `sc_step`, and `sc_nstep`, these parameters can be varied to get as efficient sampling as possible. As an example, a weakly coupled system have low-lying excitations and sampling these takes longer time than high-energy magnons. This can then be achieved by increasing `sc_step` but in these cases it is often possible to increase `timestep` as well, as the effective magnetic field, and the resulting torques results in slower precessions of the system and thus a coarser timestep can work. This always has to be tested carefully. The damping strongly affects the magnon spectrum and if a very clean signal is wanted, for careful identification of the magnon energies, then `damping` can be put to a value much lower than what is realistic (i.e. 10^-5).

An adiabatic magnon spectrum (AMS) can also be obtained by UppASD, in that case put the parameter `do_ams=y` and provide a `qfile` as for the $S(\mathbf{q},\omega)$ simulations. The AMS can conveniently be ran at the same run as $S(\mathbf{q},\omega)$ so that the two different approaches can be compared to each other. By definition, the agreement should be good, and if the two approaches give varying result for a system with low damping and close to zero temperature, then that is a strong indication that either the system is not ferromagnetic or that it might not have been correctly set up. An example of the AMS for the Heisenberg chain is shown in the right panel of Fig.3.6. Thanks to the simplicity of this 1d nearest-neighbour model, the AMS can be derived by hand and compared with the simulated results, which is left as an exercise for the reader.

3.3.2 Magnons in antiferromagnets and spin spirals

Since the simulated $S(\mathbf{q}, \omega)$ only depend on the configurations and trajectories of the simulated magnetic moments, it is not restricted to ferromagnetic systems. Although non-ferromagnetic systems typically needs to be treated a bit more carefully than ferromagnets, it is still possible to obtain magnon spectra for such systems as well. This can be illustrated by running the provided examples `HeisChainAF` which has anti-feromagnetic exchange interactions, and `HeisChainDM` which has ferromagnetic interactions but also competing Dzyaloshinskii-Moriya interactions, and compare the output with the previously simulated ferromagnetic Heisenberg chain. Starting with the anti-ferromagnetic system `HeisChainAF`, running it the same way as the `HeisChain` example should give the following outputs. Here one can notice the linear dispersion of magnon energies

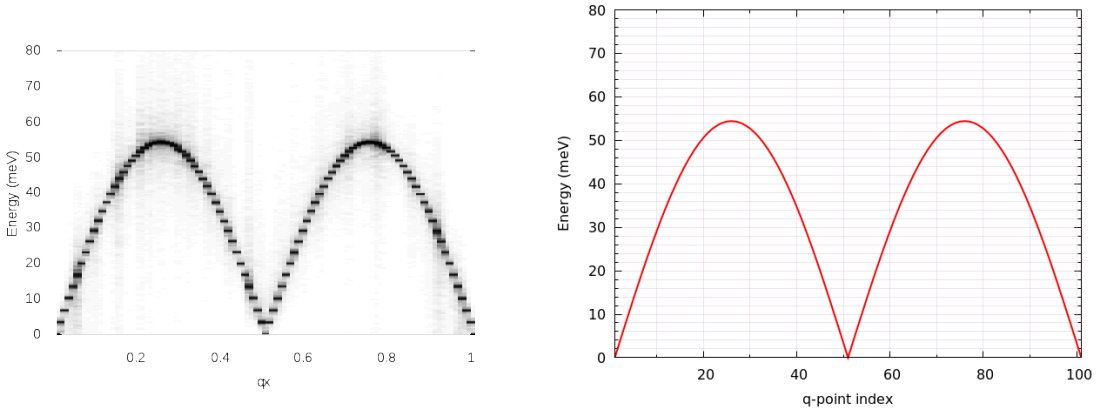


Figure 3.7: (Left) Simulated magnon spectrum for an anti-ferromagnetic Heisenberg spin chain. (Right) Adiabatic magnon spectum for the same system.

close to the Γ-point which is always obtained for anti-ferromagnets.

In the `HeisChainDM` example, the competition between Heisenberg and Dzyaloshinskii-Moriya exchange results in a helical spin spiral with a pitch-vector along $\hat{z}$ and the moments rotate in the $\hat{x}\hat{y}$-plane. The corresponding magnon spectrum is shown in Fig. 3.8 where it can be noticed that the minimum energy is not found for the Γ-point but for the q-point q_0 corresponding to the wave-vector of the resulting spin spiral. It can also be seen that the agreement between the AMS and $S(\mathbf{q}, \omega)$ is good but not perfect here. This highlights the important fact that the AMS currently does not have a general support for treating DMI interactions and while it can be expected to perform well for co-planar spin spirals, as found in this case, it should be handled with care. It can also be noted that the AMS only picks up one of the two non-degenerate magnon branches while both q_0^+ and q_0^- are sampled by the $S(\mathbf{q}, \omega)$.

For spin spiral systems, the magnon dispersions do not behave as they in collinear systems. Instead there is a much stronger variation of the dispersion relations depending on which axis the excitations are sampled along. This can be observed by changing the `do_sc_local_axis` parameter and compare the simulated spectra. Also, running the `sqw_map.sh` post-processing script creates a combined figure of the magnon spectra along all cartesian/local axes in the file `sqw_parts.png` which is plotted for this system in the right panel of Fig. 3.8

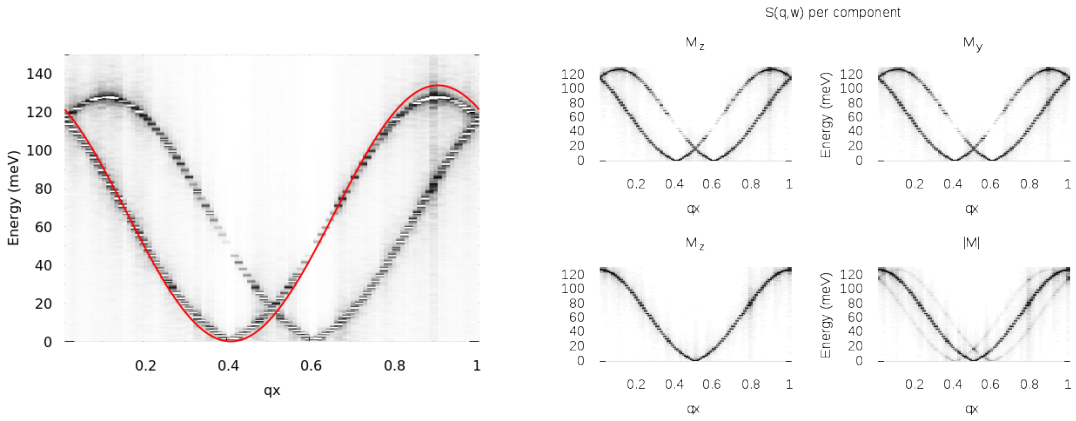
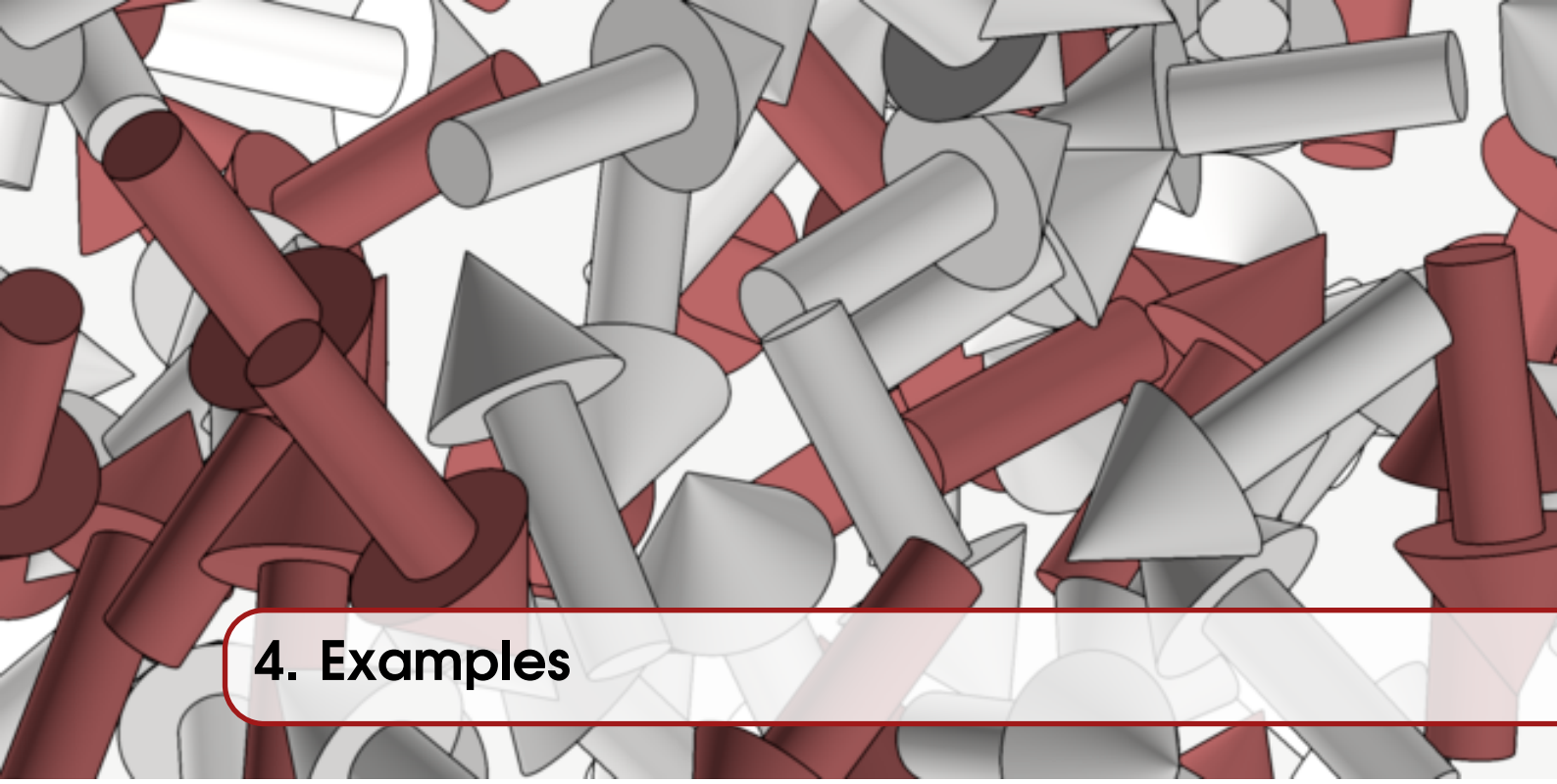


Figure 3.8: (Left) Simulated magnon spectrum for a Heisenberg spin chain with DM interactions along with the corresponding AMS. (Right) Projections of $S(\mathbf{q}, \omega)$ to the cartesian components S^{α} and the total magnitude $|S|$ for the same system.



4. Examples

The best way to learn the code is to work through the examples in the directory `examples`. The current set of examples illustrates the range of the functionality of UppASD, and gives a feel for the data analysis that is necessary in order to interpret the results of the simulations.

4.0.1 bcc Fe

The `bccFe` directory contains the necessary input in order to simulate bcc Fe. There are a couple of Matlab scripts enclosed that plot quantities such as the average magnetization as a function of time. By varying input parameters such as the temperature, one may get a feel for how these affect these observable quantities.

The `bccFe-Tsweep` directory contains very similar input to the `bccFe` directory. However, a small wrapper is included in order to run SD simulations as a function of temperature. This is often useful for checking the transition temperature of a given system.

4.0.2 Heisenberg Spin Chain

The `HeisChain` directory contains possibly the simplest system that can be set up with UppASD: a Heisenberg ferromagnetic chain. This example is set up to allow the spin wave spectrum of the system to be probed. Since the dispersion of this system can be derived analytically (as discussed any many textbooks concerning solid state physics), it is a handy example to get acquainted with spin correlations and dynamics in UppASD.

By copying over the `sqw.simid.out` and `inpsd.dat` files into the `Sqw` directory, and running the `sqw_map.sh` script, $S(q,\omega)$ as a function of q is plotted, giving rise to the spin wave displersion. The dynamics of spin waves have been studied in this way in the paper by Bergman *et al.* [11]

4.0.3 Two-dimensional Systems

The `2d-systems` directory contains a couple of two-dimensional systems: the surface simulating the Co/Cu(001) monolayer system (`fcc001`), and the square lattice Heisenberg system (`sc`, which stands for simple cubic). In the latter example, the flag `aunits` is switched on. By running

the simulation as a function of temperature, the magnetization can be benchmarked against data obtained from the ALPS software package[1], which is contained in the `scL64ALPS.dat` file.

For both the `fcc001` and `sc` examples, a directory `Snapshots` is included, that allows snapshots of the lattices to be taken. The scripts are written in VTK, and require `coord.simid.out` and `moments.simid.out` files to work.

4.0.4 fcc Co

Another very common structure in solid state physics is the fcc structure. This example allows the measurement of the dyamical structure factor, $S(q,\omega)$, for fcc Co, as in the Heisenberg spin chain example.

4.0.5 GaMnAs

The `GaMnAs` directory contains input data that sets up a GaMnAs dilute magnetic semiconductor.

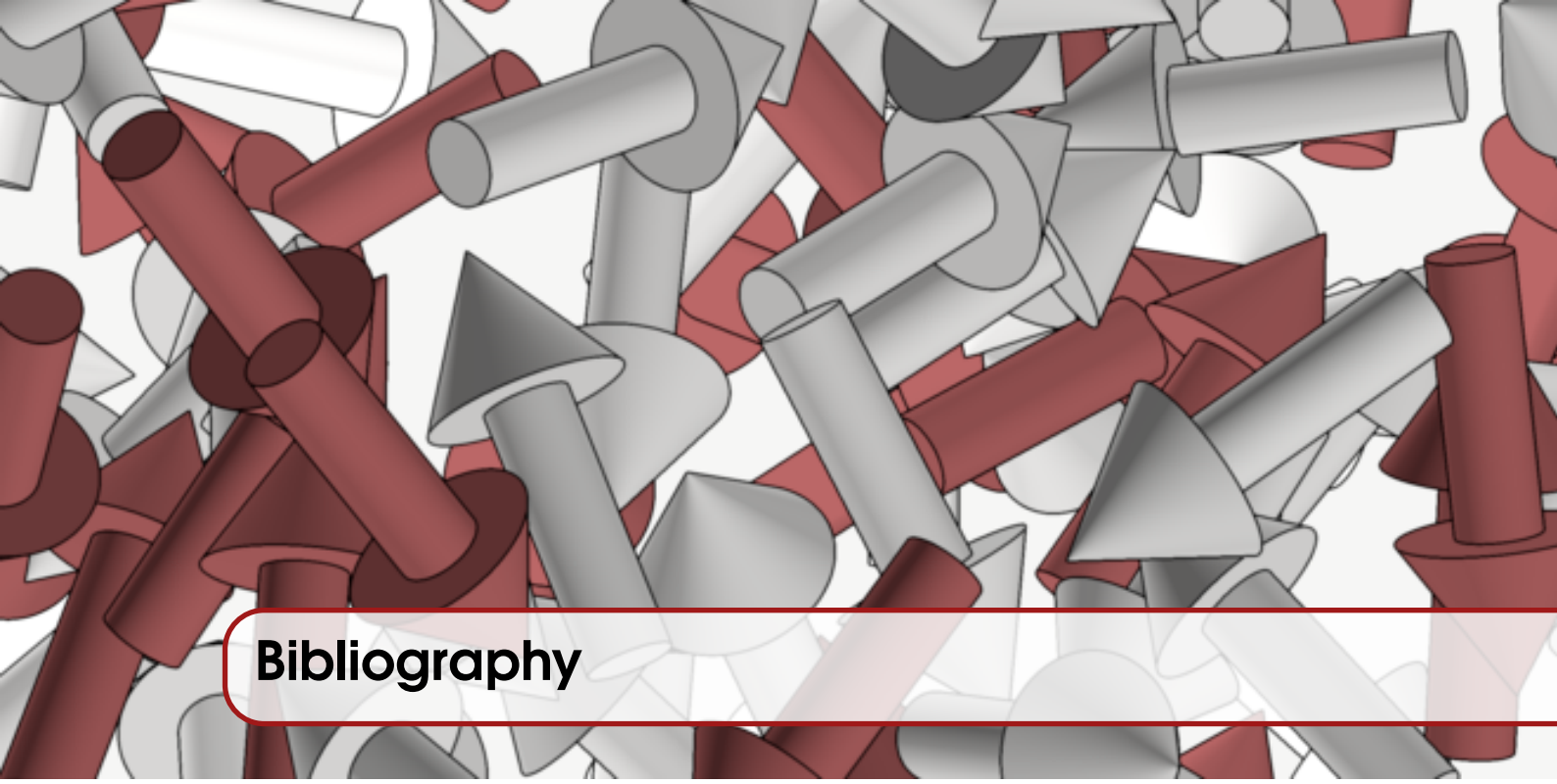
4.0.6 Random Alloy

The `Ralloy` directory contains input data necessary to set up a dilute random alloy.

4.0.7 SKKR Input (test)

The `xctensortest` directory contains the same data as in the `bccFe` directory, but set up in the tensorial format that arises from the Vienna-Budapest SKKR code. [8]

[1] `http://alps.comp-phys.org`



Bibliography

[1] B. Skubic, J. Hellsvik, L. Nordström, and O. Eriksson
J. Phys.: Condens. Matter, **20**,315203, 2008.

[2] V. P. Antropov, M. I. Katsnelson, B. N. Harmon, M. van Schilfgaarde, and D. Kusnezov
Phys. Rev. B, **54**,1019, 1996.

[3] J. L. García-Palacios and F. J. Lázaro
Phys. Rev. B, **55**,1006, 1997.

[4] R. E. Watson, M. Blume, and G. H. Vineyard
Phys. Rev., **181**,811, 1969.

[5] O. Eriksson, A. Bergman, L. Bergqvist, and J. Hellsvik
Atomistic Spin Dynamics - Foundations and Applications.
Oxford University Press, Oxford, 1st edition, 2017

[6] A.I. Liechtenstein, M.I. Katsnelson, V.P. Antropov, and V.A. Gubanov
J. Magn. Magn. Mater., **67**,65, 1987.

[7] K. Binder and D. P. Landau
A Guide to Monte Carlo Simulation in Statistical Physics.
Cambridge University Press, Cambridge, 3rd edition, 2009.

[8] L. Udvardi, L. Szunyogh, K. Palotás, and P. Weinberger
Phys. Rev. B, **68**,104436, 2003.

[9] J. H. Mentink, M. V. Tretyakov, A. Fasolino, M. I. Katsnelson, and Th. Rasing
J. Phys.: Condens. Matter, **22**,176001, 2010.

[10] Ph. Depondt and F.G. Mertens
J. Phys.: Condens. Matter, **21**,336005, 2009.

[11] A. Bergman, A. Taroni, L. Bergqvist, J. Hellsvik, B. Hjörvarsson, and O. Eriksson Phys. Rev. B, **81**,144416, 2010.