SpinW

SW SpinW

A crash course on SpinW

SpinW (spin-double-u) is a MATLAB library that can optimize magnetic structures using mean field theory and calculate spin wave dispersion and spin-spin correlation function for complex crystal and magnetic structures.

PRESENTED BY

Simon Ward

Scientific Software Developer - ESS

Johan Hellsvik

Senior Postdoc - Nordita

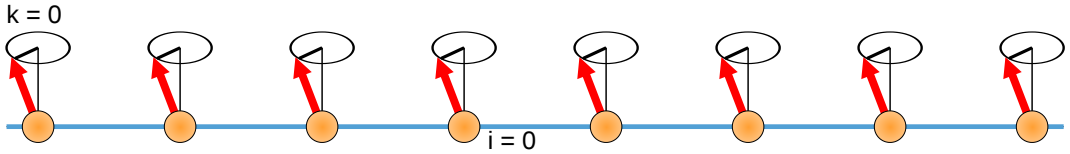
SW

Linear Spin Wave Theory Review

15 minute introduction on LSWT

Spin waves

Spin waves (magnons) are propagating disturbances of an ordered magnetic lattice



The magnetic ordering arises due to the exchange interactions J_{ij} between electrons on atomic sites i and j.

The dispersion relation $\omega(\mathbf{q})$ of spin waves can be measured by inelastic neutron scattering, and directly depends on the exchange interactions

Spin waves are deviations from an ordered state, so in principle can only be seen in the ordered state

Linear Spin Wave Theory

We'll now derive the spin wave dispersion for a ferromagnet in linear spin wave theory

The steps involved in the derivation are:

1. Beginning with a spin Hamiltonian, express the spin operator vector $\mathbf{S}_i = (\hat{S}_i^x, \hat{S}_i^y, \hat{S}_i^z)$ as the ladder operators $(\hat{S}_i^+, \hat{S}_i^-, \hat{S}_i^z)$

2. Map the raising (lowering) operators to bosonic anihilation (creation) operators $a^\dagger$ (a) via the Holstein-Primakoff transformation.

3. The transformed Hamiltonian is a series expansion in powers of $a^\dagger, a$, but in linear spin wave theory we take only the terms which are linear in $a^\dagger a$.

4. Fourier transform the Hamiltonian.

5. Diagonalise the Hamiltonian, ensuring that the commutation relation of the bosonic operators are respected (for two-sublattice system such as antiferromagnetic, the Bogoliubov transformation can be used for this purpose).

Heisenberg Hamiltonian:

The Hamiltonian of an ordered magnet can be expressed in terms of the spin operators for site i, $\mathbf{S}_i$.

For example, the Heisenberg Hamiltonian is:

$$\mathcal{H} = \sum_{mi,nj} J_{mi,nj} \mathbf{S}_i \cdot \mathbf{S}_j$$

where $J_{mi,nj}$ is the exchange interaction with the indices i and j running over magnetic atoms within a single magnetic unit cell, and m and n label different unit cells.

$\mathbf{S}_i = (\hat{S}_i^x, \hat{S}_i^y, \hat{S}_i^z)$ is the spin operator for site i, which can be re-expressed in terms of the ladder operators $\hat{S}_i^+, \hat{S}_i^-, \hat{S}_i^z$:

$$\begin{aligned}\mathcal{H} &= \sum_{mi,nj} J_{mi,nj} (\hat{S}_i^x \hat{S}_j^x + \hat{S}_i^y \hat{S}_j^y + \hat{S}_i^z \hat{S}_j^z) \\ &= \sum_{mi,nj} J_{mi,nj} ((\hat{S}_i^+ \hat{S}_j^- + \hat{S}_i^- \hat{S}_j^+)/4 + \hat{S}_i^z \hat{S}_j^z)\end{aligned}$$

since $\hat{S}^{\pm} = \hat{J}^x \pm i\hat{S}^y$ so $\hat{S}^{x,y} = \frac{\sqrt{\pm 1}}{2}(\hat{S}^+ \pm \hat{S}^-)$

Holstein-Primakoff transformation

To determine the magnons (magnetic normal modes), we map the excitations to a simple harmonic operator

The ordered ground state is the state with the maximum azimuthal quantum number $|m = S\rangle$, and the action of the ladder operator is:

$$\hat{S}^{\pm}|m\rangle = \sqrt{(S \mp m)(S + 1 \pm m)}|m \pm 1\rangle$$

We now map the ladder operators to boson creation and anihilation operators $a^{\dagger}$ and a:

$$a|n\rangle\sqrt{n}|n-1\rangle \qquad a^{\dagger}|n\rangle = \sqrt{n+1}|n+1\rangle$$

mapping also the ordered state $|m = S\rangle$ to the vacuum state $|n = 0\rangle$, such that the action of $a^{\dagger}|n\rangle$ is analogous to $\hat{S}^{-}|S\rangle$

So, substituting $m = S - n$ and re-arranging the ladder operator terms:

$$\hat{S}^{+}|n\rangle = \sqrt{(2Sn)(1 - \frac{n-1}{2S})}|n-1\rangle$$
$$\hat{S}^{-}|n\rangle = \sqrt{2S(n+1)(1 - \frac{n-1}{2S})}|n+1\rangle$$

In linear spin wave theory we ignore the terms in red

Using the identities $n = aa^\dagger$ and $S^z|m\rangle = m|m\rangle$ we have the following mappings

$$\begin{aligned}\hat{S}_i^+ &= a_i\sqrt{2S_i} \\ \hat{S}_i^- &= a_i^\dagger\sqrt{2S_i} \\ \hat{S}_i^z &= S_i - aa^\dagger\end{aligned}$$

So the Heisenberg Hamiltonian (ignoring terms which are not linear in $aa^\dagger$) is now:

$$\mathcal{H} = \sum_{mi,nj} J_{mi,nj}\left[\frac{\sqrt{S_iS_j}}{2}\left(a_ia_j^\dagger + a_i^\dagger a_j\right) - S_ia_ia_i^\dagger - S_ja_ja_j^\dagger\right]$$

Fourier transformation

In the expression for the spin Hamiltonian we have a sum over all magnetic sites. This can be simplified to just sites i and j within the magnetic unit cell by Fourier transforming the Hamiltonian. The Fourier transform of the Holstein-Primakoff operator is:

$$a^{(\dagger)} = \frac{1}{\sqrt{N}} \sum_{\mathbf{q}} \exp(-i\mathbf{q}\cdot\mathbf{r}) b_{\mathbf{q}}^{(\dagger)}$$

Let us take one term from the Hamiltonian, $\sum_{mi,nj} J_{mi,nj} a_i a_j^\dagger$; its Fourier transform is:

$$\mathrm{FT}\left[\sum_{mi,nj} J_{mi,nj} a_i a_j^\dagger\right] = \frac{1}{N} \sum_{mi,nj} \sum_{\mathbf{q},\mathbf{q}'} J_{mi,nj} \exp(-i\mathbf{q}\cdot\mathbf{r}_m) \exp(-i\mathbf{q}'\cdot\mathbf{r}_n) b_{i,\mathbf{q}} b_{j,\mathbf{q}'}^\dagger$$

The periodicity of the crystal implies that the exchange interaction is translationally invariant such that $J_{mi,nj} = J_{ij}(\mathbf{d})$ where $\mathbf{d} = \mathbf{r}_m - \mathbf{r}_n$ so the Fourier transform becomes:

$$\frac{1}{N} \sum_{\mathbf{q},\mathbf{q}'} \left[\sum_{ij} J_{ij}(\mathbf{d}) \exp(-i\mathbf{q}'\cdot\mathbf{d})\right] \exp(-i(\mathbf{q}-\mathbf{q}')\cdot\mathbf{r}_m) b_{i,\mathbf{q}} b_{j,\mathbf{q}'}^\dagger$$

where the terms in the square brackets is the Fourier transform of the exchange interactions $J_{ij}(\mathbf{q})$ where the indices ij label only sites within the magnetic unit cell. We now use the identity $(1/N)\sum_{\mathbf{q},\mathbf{q}'} \exp(-i(\mathbf{q}-\mathbf{q}')\cdot\mathbf{r}) = \delta_{\mathbf{q}\mathbf{q}'}$ to obtain:

$$\mathrm{FT}\left[\sum_{mi,nj} J_{mi,nj} a_i a_j^\dagger\right] = \sum_{ij} J_{ij}(\mathbf{q}) b_{i,\mathbf{q}} b_{j,\mathbf{q}}^\dagger$$

So the Fourier transformed Hamiltonian becomes:

$$\mathcal{H}_{\mathbf{q}} = \sum_{ij} J_{ij}(\mathbf{q}) \left[\frac{\sqrt{S_i S_j}}{2} \left(b_{i,\mathbf{q}} b^{\dagger}_{j,\mathbf{q}} + b^{\dagger}_{i,\mathbf{q}} b_{j,\mathbf{q}} \right) - S_i b_{i,\mathbf{q}} b^{\dagger}_{i,\mathbf{q}} - S_j b^{\dagger}_{j,\mathbf{q}} b_{j,\mathbf{q}} \right]$$

Which can be expressed as a matrix equation:

$$\mathcal{H}_{\mathbf{q}} = \begin{pmatrix} b_1^{\dagger} & \dots & b_N^{\dagger} & b_1 & \dots & b_N \end{pmatrix} \begin{pmatrix} \left[\frac{\sqrt{S_i S_j}}{2} J_{ij}(\mathbf{q}) - \delta_{ij} S_i J_{ij}(\mathbf{q}=0) \right] & 0 \\ 0 & \left[\frac{\sqrt{S_i S_j}}{2} J_{ij}(\mathbf{q}) - \delta_{ij} S_i J_{ij}(\mathbf{q}=0) \right] \end{pmatrix} \begin{pmatrix} b_1 \\ \vdots \\ b_N \\ b_1^{\dagger} \\ \vdots \\ b_N^{\dagger} \end{pmatrix}$$

where the square brackets denote a block matrix and we have omitted the $\mathbf{q}$ indices on the bosonic operators.

The off-diagonal blocks are only zero in this case because we are using the Heisenberg Hamiltonian. For anisotropic Hamiltonians (e.g. XXZ or Dzyaloshinskii-Moriya interactions), these blocks will not be zero

Setting:

$$\mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_N \\ b_1^\dagger \\ \vdots \\ b_N^\dagger \end{pmatrix}$$

The Hamiltonian is:

$$\mathcal{H}_\mathbf{q} = \mathbf{b}^\top h_\mathbf{q} \mathbf{b}$$

With the matrix $h_\mathbf{q}$ as shown previously and $\top$ indicates a complex conjugate transpose.

At each $\mathbf{q}$ the eigenvalues of the matrix $h_\mathbf{q}$ will be the magnon energies $\omega(\mathbf{q})$ and the eigenvalues can be used to calculate the spin-spin correlation function and hence the neutron scattering cross-section.

Heisenberg Ferromagnet

For a ferromagnet, there is only one magnetic atom in the unit cell, so the Hamiltonian is diagonal already:

$$\mathcal{H}_{\mathbf{q}} = \begin{pmatrix} b^\dagger & b \end{pmatrix} \begin{pmatrix} \frac{S}{2}J(\mathbf{q}) - SJ & 0 \\ 0 & \frac{S}{2}J(\mathbf{q}) - SJ \end{pmatrix} \begin{pmatrix} b \\ b^\dagger \end{pmatrix}$$

The Fourier transform is just:

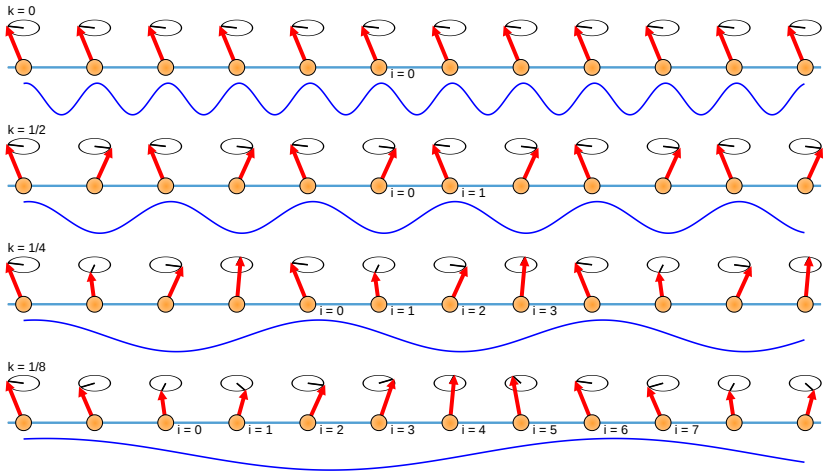
$$\begin{aligned} J(\mathbf{q}) &= J\left(\exp(-i\mathbf{q}\cdot\mathbf{d}) + \exp(i\mathbf{q}\cdot\mathbf{d})\right) \\ &= 2J\cos(\mathbf{q}\cdot\mathbf{d}) \end{aligned}$$

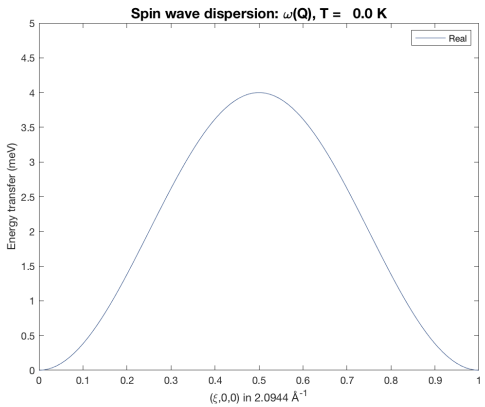
So we obtain $\omega(\mathbf{q}) = JS\left(\cos(\mathbf{q}\cdot\mathbf{d}) - 1\right)$

In general, however, the Hamiltonian is not diagonal and so must be diagonalised.

This can be done using the Bogoliubov transformation for an antiferromagnet

SpinW uses a more general method due to Colpa, as explained in the SpinW paper





Now we have $\omega_{\mathbf{q}}$ and half the story. Remember for neutron scattering:

$$\frac{d^2\sigma}{d\Omega dE} \propto F^2(\mathbf{Q}) \sum_{\alpha,\beta} (\delta_{\alpha\beta} - \hat{q}_\alpha \hat{q}_\beta) S^{\alpha\beta}(\mathbf{q}, \omega)$$

With the correlation function:

$$S^{\alpha\beta}(\mathbf{k}, \omega) = \frac{1}{2\pi\hbar} \int dt e^{-i\omega t} \langle S^\alpha(\mathbf{q}, 0) S^\beta(-\mathbf{q}, t) \rangle$$

$\langle S^\alpha(\mathbf{q}, t) \rangle$ is related to the Fourier transform of the eigenvectors of the Hamiltonian at $\mathbf{q}$.

Summary

Linear spin wave theory calculations requires:

- The exchange interactions
- The magnetic structure

It expands the spin ladder operators as a power series in bosonic creation/anihilation operators, keeping only linear terms

For each momentum transfer $\mathbf{q}$ a Hamiltonian matrix has to be diagonalised to obtain the magnon energies and spin-spin correlation functions

The size of the matrix is proportional to the number of magnetic atoms in the unit cell

So, larger (or more complex) magnetic structures require more memory; and more $\mathbf{q}$-points require longer processing time

References

S. Toth and B. Lake, J. Phys.: Condens. Matter **27** 166002 (2015) arxiv:1402.6069

S. Petit Collection SFN **12** 105-121 (2011) 10.1051/sfn/201112006

SW

General spin Hamiltonian - SpinW

How does SpinW work?

General spin Hamiltonian:

$$H = \sum_{mi,nj} \mathbf{S}_{mi}^T \cdot J_{mi} \cdot \mathbf{S}_{nj} + \sum_{mi} \mathbf{S}_{mi}^T \cdot A_i \cdot \mathbf{S}_{mi} + \mu_B \mathbf{H}^T \sum_{mi} g_i \mathbf{S}_{mi}$$

With the anisotropic and antisymmetric (Dzyaloshinskii-Moriya) exchange interactions:

$$J_S = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix}; J_A = \begin{bmatrix} 0 & D_z & -D_y \\ -D_z & 0 & D_x \\ D_y & -D_x & 0 \end{bmatrix}$$

Key Points:

Non-Bravais lattice
- Additional rotation on every site within unit cell

General interactions
- Multi-q magnetic ground states are possible

SpinW

- Solves the general spin Hamiltonian
- Calculates spin-spin correlation function
- Numerical and symbolical
- Can apply crystal symmetry operators on the Hamiltonian - Solving single-q magnetic structures
- Solves multi-q magnetic structures on a magnetic supercell
- Open source, runs on MATLAB and now python
- More information: http://www.spinw.org
- Download from: https://www.github.com/spinw/spinw

Operating System

- All modern OS's are supported

MATLAB

- All versions after R2014b are fully supported.
- OPTIONAL: Symbolic Toolbox
- OPTIONAL: Parallel Computing Toolbox

Notes

- More memory is needed if you have more magnetic atoms
- More q-points takes longer
- Python is now technically supported but will not be covered here. See https://www.github.com/spinw/PySpinW

Now lets install

Get the code - The manual way:

SpinW is available at:
https://www.github.com/spinw/spinw/releases/latest

Steps:

- Unzip the archive into your preferred directory
- Open MATLAB and run `install_spinw` from inside this directory
- Verify with `s = spinw;`

Updating:

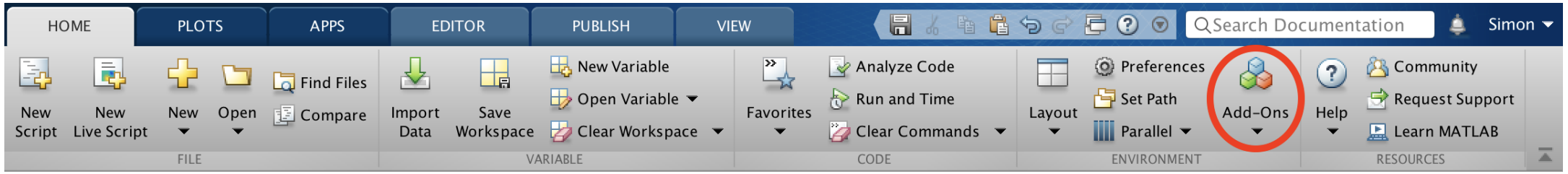
I am trying to make releases 2-3 times a year. It's always nice to be on the latest code!
There is a self update function `sw_update`
This retrieves and installs the package from the link above.

Developing:

If you want to help develop a feature and contribute, please git clone https://www.github.com/spinw/spinw.git and create a pull request (to development branch)

Using the MATLAB package - The easy way

SpinW is now a MATALB add-on and can be downloaded directly from Mathworks.



Then search for SpinW and hit install.

Verify with `s = spinw;`

Updating:

Add-Ons → Check for updates → Update

NOTE

This version may not correspond to the `sw_update` version!

Function help

For any function that starts with `sw_*` use:
`help sw_*`

SpinW class methods

for spinw class methods use:
`help spinw.function_name`.
For help on plotting commands, use:
`help swplot`.

Online Documentation

All help can be found on http://www.spinw.org or https://spinw.github.io/spinwdoc

SW

Crash course in MATLAB

SpinW runs in MATLAB and Python, but today we will be focusing on MATLAB.

The concepts.... Arrays, Indexing, Objects

They are created using square brackets – comma or space separation for columns in a row, semi-colon separation for new rows, higher dimensions using indexing / repmat command

1D arrays

```
w_1row = [1,2,3];
w_1col = [1;2;3];
w_1col_to_1row = [1;2;3]';
```

2D arrays

```
w_rowcol = [1,2,3;4,5,6];
```

3D arrays

```
w3d = repmat(w_rowcol,[1,1,3])
w3d(:,:,1) = w_rowcol;  w3d(:,:,2) = w_rowcol+2;
```

Creating Strings

```
my_string = ‘hello world’;
```

Strings arrays are concatenated similar to numeric arrays....

```
my_string2 = [my_string, ’ hello everyone’];
```

ARRAYS - STRUCTURED

Structure arrays provide a way of storing more general information, referenced by *fields*.

```
w = struct();
w.tom = [1,2,3];
w.dick = ‘hello world’;
w.harry = [4,5,6,7;8,9,10,11];
```

Cell arrays are another generic data storage mechanism, with a matrix-like structure.

```
cell1 = {[1,2,3], 'hello_world', [4,5,6;7,8,9]};
cell2 = {[1,23], 'hello_world'; [4,5,6;7,8,9], cell1};
```

Note that the cell array is described by a curly bracket, not a square bracket!

Arrays can be indexed. Note that the ordering is the reverse to Python!.

```
w1 = [1,3,5,7]
```

So `w1(3) = 5` etc.

```
wcell = {'hello','world',[1,2,3]}
```

So `wcell{3} = [1,2,3]` etc.

In 2D:

```
w2 = [1,2,3; 4,5,6]
```

So `w2(2,3) = 6`

```
w2(:,[1:2]) = [1,2;4,5]
w2(:,[1,3]) = [1,3;4,6]
```

Objects are data structures with defined properties, and internal self-consistency checks.

- Cell arrays, structure arrays etc. are examples of Matlab's built-in objects.
- You can define your own, which is what we have done with SpinW

You create a SpinW object with:

```
s = spinw();
```

Methods are the functions that work on defined objects.

SpinW has many methods. e.g.

```
s.plot()
```

Getting help for objects and methods

To find all the methods working on an object, use the `methods` function which the name of the object class.

```
methods(spinw)
```

Get help if you already know the name of a particular method for an object class *e.g.* `addatom`:

```
help spinw/addatom
doc spinw/addatom
```

Function help

For any function that starts with `sw_*` use:
`help sw_*`

SpinW class methods

for spinw class methods use:
`help spinw.function_name`.
For help on plotting commands, use:
`help swplot`.

Online Documentation

All help can be found on http://www.spinw.org or https://spinw.github.io/spinwdoc

SW

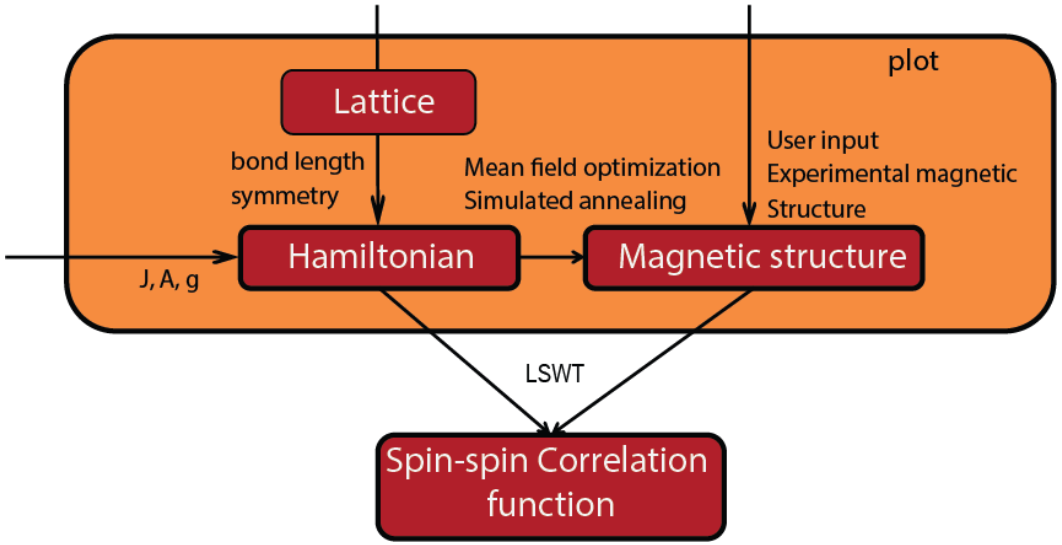
For the things we have to learn before we can do them, we learn by doing them.

Aristotle

SW

Introductory Tutorials

Getting started in SpinW - Referring to Kim Lefmanns talk for spinwave theory.



To do a calculation, you need 3 parts. *Crystal structure, exchange structure, magnetic structure*

SW

The FM spin chain - A recap

Download the script here: `sw_tutorial_00.m`

SW

Excitations on a triangular lattice

Download the script here: `sw_tutorial_01.m`

SW

$k = [1\,1\,0]/3$ magnetic structure

Let's try this with a $k = [1\,1\,0]/3$ magnetic structure

Creating the lattice

```
tri = spinw;
tri.genlattice('lat_const',[3 3 4],'angled',[90 90 120])
plot(tri)
```

We have:

- Created a SpinW object
- Generated a lattice of $a = 3\text{Å}$, $b = 3\text{Å}$, $c = 4\text{Å}$ and $\alpha = \beta = 90°$, $\gamma = 120°$

In the plot window, you can zoom with the mouse wheel, pan by pressing the Ctrl button while dragging. Change the plot range and view direction by pressing the corresponding button on the top.

Questions:

What is the default symmetry and what does it mean?

Adding atoms

```
tri.addatom('r',[0 0 0],'S',3/2,'label','MCr3')
plot(tri)
```

We have added an magnetic Cr^{3+} at position $[0,0,0]$ with spin $S = 3/2$

Creating the Spin-Hamiltonian

We create an antiferromagnetic first neighbor Hamiltonian plus easy plane single ion anisotropy

$$H = \sum_{mi,nj} \mathbf{S}^T_{mi} \cdot J_{mi} \cdot \mathbf{S}_{nj} + \sum_{mi} \mathbf{S}^T_{mi} \cdot A_i \cdot \mathbf{S}_{mi}$$

Where there is only one interaction of value `J = 1` and easy plane single ion anisotropy `A = 0.1`

Creating the Spin-Hamiltonian

We create implement this in code with:

```
A0 = 0.1;
J1 = 1;
tri.addmatrix('label','J1','value',J1)
tri.addmatrix('label','A','value',[0 0 0;0 0 0;0 0 A0])
```

We then assign these matrices to bonds:

```
tri.gencoupling
tri.addcoupling('mat','J1','bond',1)
tri.addaniso('A')
```

Red ellipsoids represent the single ion anisotropy on the plot (equienergetic surface)

Questions:

What have we done in each code part?
Examine the plot and test different values of `A0` with different signs

Creating the magnetic structure:

We have seen the ground state magnetic structure of the above Hamltonian is a spiral, with propagation vector of $(1/3, 1/3, 0)$.
We define the plane of the spiral as the ab plane

```
tri.genmagstr('mode', 'helical', 'S', [1;0;0], 'k',[1/3 1/3 0], 'n', [0 0 1], 'nExt', [1 1 1])
plot(tri, 'range', [3 3 1/2], 'cellMode', 'inside', 'magColor', 'red')
```

Careful: the given spin vector is column vector!

Questions:

What are the angles between nearest neighbor moments?

Calculating the spin wave dispersion

We calculate the spin wave dispersion along the $(H, H, 0)$ high symmetry direction

```
spec = tri.spinwave({[0 0 0] [1 1 0] 500}, 'hermit', false);
figure
sw_plotspec(spec, 'mode', 'disp', 'imag', true, 'colormap', [0 0 0], 'colorbar', false)
axis([0 1 0 5])
```

Questions:

How many modes are there and why?
What does the red line mean?
Did you get any warning?

Calculating the spin-spin correlations

The spin-spin correlations are already calculated, however it contains 9 numbers per Q-point per mode. It is not possible to show this on a single plot. But:

1. We can calculate the neutron scattering cross section
2. We can select one of the components $S^{\alpha\beta}(\mathbf{Q},\omega)$
3. We can sum up the diagonal $S^{\alpha\alpha}(\mathbf{Q},\omega)$

```
spec = sw_egrid(spec, 'component', {'Sxx+Syy' 'Szz'}, 'Evect', 0:0.01:5);
% Try other components!
figure
sw_plotspec(spec,'mode','color','dE',0.2,'imag',false)
axis([0 1 0 5.5])
caxis([0 3])
```

Questions:

How is it related to the magnetic propagation vector?
Why are some modes gapped? Which correlations are gapped?
Why do we have Szz?

SW

$k = 0$ magnetic structure

Let's try this with a $k = 0$ magnetic structure and explore SpinW deeper

$k = 0$ magnetic structure

Duplicate the original object using the `.copy()` command,

```
triNew = copy(tri);
triNew.genmagstr('mode','rotate','n',[0 0 1])
phi1 = atan2(triNew.magstr.S(2,1),triNew.magstr.S(1,1));
triNew.genmagstr('mode','rotate','n',[0 0 1],'phi',-phi1)
plot(triNew,'range',[3 3 1])
```

Compare the energy per spin of the old magnetic structure and the new magnetic structure using the `spinw.energy()` function.

Questions:

Why are we using the `.copy()` command?
How does the magnetic structures compare?
Are they the same? Why the rotation?

Calculating the spin wave dispersion

We calculate the spin wave dispersion along the $(H, H, 0)$ high symmetry direction

```
spec = triNew.spinwave({[0 0 0] [1 1 0] 500}, 'hermit', false);
figure
subplot(2, 1, 1)
sw_plotspec(spec, 'mode', 'disp', 'imag', true, 'colormap', [0 0 0], 'colorbar', false)
axis([0 1 0 5])
spec = sw_egrid(spec, 'component', 'Sperp', 'Evect', 0:0.01:5.5);
subplot(2, 1, 2)
sw_plotspec(spec, 'mode', 'color', 'dE', 0.2, 'imag', false)
axis([0 1 0 5.5])
caxis([0 3])
```

Questions:

How many number of modes are there and why?
Is there more than before?
Why are there vertical lines in the dispersion?
Which structure is the correct one?

SW

The FM kagome lattice

Download the script here: `sw_tutorial_02.m`

EXAMPLE 2

This tutorial will be up to you, using what you have learned in tutorial 1.

Help is available by the MATLAB command, SpinW website and for a limited time.... Me.

SpinW

The Organic Materials Database

The organic materials database is an open access electronic structure database for 3-dimensional organic crystals. It provides tools for search queries based on data-mining and machine learning techniques.

PRESENTED BY

Johan Hellsvik

Senior Postdoc - Nordita

SpinW

Using the OMDB and SpinW

The organic materials database has the spin-wave spectra computed for a few materials. This can also be done using SpinW. We will introduce you on how to model a material found in the OMDB through examples.

PRESENTED BY

Simon Ward

Scientific Software Developer - ESS

Johan Hellsvik

Senior Postdoc - Nordita

SW

Spin wave excitations of magnetic metalorganic materials

Introduction to the OMDB and excitations in magnetic metalorganic materials.
All examples can be found:omdb.mathub.io

SW

Modeling $Co(2\text{-}pymS)_2$ (2-pymSH = 2-mercaptopyrimidine)

The OMDB link is: **here**, paper is: **here**

Open up the **OMDB link**, so you have all the necessary parameter information.

Creating the lattice

- Using tutorial 1 as a template, create a SpinW object
- Define the lattice. As a hint, you will have to specify the symmetry of *P 21 21 21* (space-group 19)

```
s = spinw();
s.genlattice('lat_const', [# #, #], 'angled', [#,  #,  #], 'sym', #)
```

- Add an atom of Co^{2+} at `[0.08183, 0.80958, 0.37295]` where the spin is 1

```
s.addatom('r', [#, #, #], 'S', #, 'label', '#', 'color','b')
s.plot()
```

Questions:

- *Plot the structure. How many Cobalt atoms have been generated?*
- *Why are they in that position?*
- *Compare with the image on the OMDB site.*

Generate Bonds

- Use `gencoupling` to generate all bonds with a cutoff of 50 angstroms.
- Print out the first 50 bonds.

```
s.gencoupling('maxDistance', #)
s.table('bond',#)
```

Exchange interactions

From the OMDB website, the exchange interactions are:

i	j	r [Å]	J_{ij} [meV]
1	4	5.80	-6.92
3	2	5.80	-6.92
1	3	5.77	-6.90
2	4	5.77	-6.90
1	2	10.18	0.08
4	3	10.18	0.08
2	1	9.15	0.07
3	4	9.15	0.07
1	2	9.91	0.05
3	4	9.91	0.05

Implementing the exchange interactions

Now we need to create the exchange matrices and assign them to bonds.

- Use `addmatrix` to add a matrix to your SpinW object
- Use `addcoupling` to assign the above matrix to a bond

Note the translation of bonds between OMDB and SpinW!

```
s.addmatrix('label', '##', 'value', ##, 'color', '##')
s.addcoupling('mat','##','bond',#)
```

Questions:

- *Why are there 10 exchanges on the OMDB site? What are they in SpinW?*
- *How does this effect the value of the exchange?*

Create the magnetic structure:

The magnetic structure of this compound can be described in the chemical unit cell where the spins are anti-ferromagnetic in the z-direction.

- The spins can be given using the *'direct'* option.
- The spins are `0 0 0 0; 0 0 0 0; 1 1 -1 -1`

```
s.genmagstr('mode', '###', 'S', ###)
```

Careful: the given spin vector is column vector!

- Plot the magnetic structure and verify it is as expected.

Calculate the spectrum

We can calculate the spin-wave spectrum at Q-points using the `spinwave` function. On the OMDB website we have high symmetry trajectories throughout the Brillouin zone. The sequence of high symmetry points are for eight legs in the BZ, from \Gamma to T. The coordinates from these high symmetry can be read out from the `KPOINTS.gz` file that are posted on the OMDB page. Use 500 points along these trajectories to get the Spin-Spin correlations. These need to be converted into an observable and plotted using `sw_plot`.

```
spec = s.spinwave({[ _StartingPoint_ ], [ _trajectory1_ ], [ _trajectory2_ ] ...
    [ _trajectory3_ ], [ _trajectory4_ ], ...
    [ _trajectory5_ ], [ _trajectory6_ ], ...
    [ _trajectory7_ ], [ _trajectory8_ ] ...
    500});
sw_plotspec(spec, 'mode', 'disp', 'imag', true, 'colormap', [0 0 0], 'colorbar', false)
```

Questions:

- *How does this compare with the result on the OMDB page?*
- *Comment on the results and suggest a reason for the discrepancies.*

SW

Modeling $Co(2\text{-}pymS)_2$ (2-pymSH = 2-mercaptopyrimidine)

The result can be found here: `sw_tutorial_04.m`

SW

Modeling $C_4H_6MnO_6$

The OMDB link is: **here**, paper is: **here**

Generate the spin-wave spectra of $C_4H_6MnO_6$

In the previous section we showed a step by step guide on how to use OMDB to model a real material. Now it's up to you to model $C_4H_6MnO_6$. Remember the following steps:

- Generate the crystal structure.
- Generate the Hamiltonian.
- Generate the magnetic structure (*Hint, it's the same as in the previous example*)
- Plot and comment on the results.

Don't be afraid to ask Johan and Simon for help.

SW SpinW

Thank you!

Well done if you're still awake!

SpinW