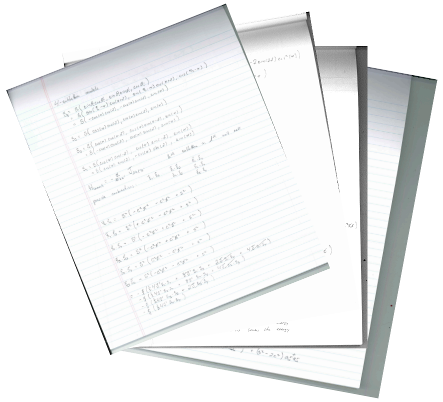
User-friendly software for modeling collective spin wave excitations

Steven Hahn, Garrett Granroth, Ovidiu Garlea

Neutron Scattering Division
Oak Ridge National Laboratory

Inelastic Neutron Scattering

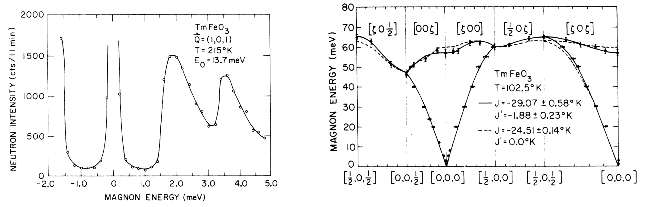
Macroscopic Modeling



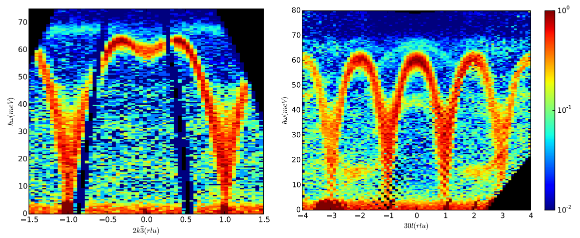
SpinWaveGenie

Data Acquisition & Analysis

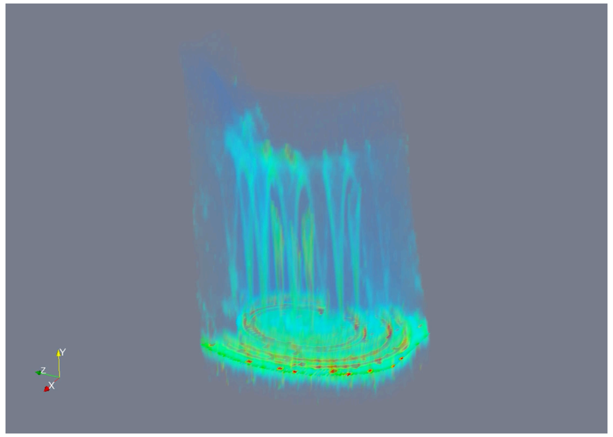
TmFeO$_3$



YFeO$_3$



S. Shapiro *et al.*, Phys. Rev. B, **10**, 2014 (1974) S. E. Hahn *et al.* ,Phys. Rev. B, **89**, 014420 (2014)



OAK RIDGE National Laboratory | HIGH FLUX ISOTOPE REACTOR | SPALLATION NEUTRON SOURCE

Calculating Spin Waves

- Write Hamiltonian in terms of local spin operators

$$\bar{\boldsymbol{S}}_i = \underline{U}_i \boldsymbol{S}_i$$

- Holstein–Primakoff transformation

$$S_+ = \sqrt{2S}a \quad S_- = \sqrt{2S}a^\dagger \quad S_z = S - a^\dagger a$$

- expand in powers of $\frac{1}{\sqrt{S}}$

$$H = H_0 + H_1 + H_2 + ...$$

- Determine spin configuration that minimizes the classical energy (H_0).
- Check that the first order terms (H_1) vanish.

Calculating Spin Waves, cont.

- Determine elements in matrix

$$H_2 = \sum_{\boldsymbol{q}} \boldsymbol{v_q}^{\dagger} \cdot L(\boldsymbol{q}) \cdot \boldsymbol{v_q} \qquad \boldsymbol{v_q} = \left(a^1_{\boldsymbol{q}}, ..., a^M_{\boldsymbol{q}}, a^{1\dagger}_{-\boldsymbol{q}}, ..., a^{M\dagger}_{-\boldsymbol{q}}\right)$$

- Bogoliubov transformation
  - basis transformation such that H becomes diagonal

$$H_2 = \sum_{\boldsymbol{q}} \boldsymbol{w}^{\dagger}_{\boldsymbol{q}} \cdot L'(\boldsymbol{q}) \cdot \boldsymbol{w_q}$$

- Inelastic neutron scattering cross section
  - spin-spin correlation function

$$S_{\alpha\beta}(\mathbf{q},\omega) = \frac{1}{2\pi N} \int dt\, e^{-i\omega t} \sum_{i,j} e^{-i\mathbf{q}\cdot(\mathbf{R}_i - \mathbf{R}_j)} \langle S_{i\alpha}(0) S_{j\beta}(t) \rangle$$

$$\langle \boldsymbol{S}^{\alpha}_i(0) \boldsymbol{S}^{\beta}_j(t) \rangle = \langle (\underline{U}_i^{-1} \bar{\boldsymbol{S}}_i(0))^{\alpha} (\underline{U}_j^{-1} \bar{\boldsymbol{S}}_j(t))^{\beta} \rangle$$

JT Haraldsen, RS Fishman, J. Phys.: Condens. Matter 21 216001 (2009)
RS Fishman et al., Phys. Rev B 87 134416 (2013)

Introducing SpinWaveGenie

Fork me on GitHub

- Https://GitHub.com/SpinWaveGenie/SpinWaveGenie
- BSD 3-clause license
- Platforms
  - MacOS (10.13 High Sierra or later)
    - Homebrew formula
  - Linux (GCC 7+)
    - .spec file for building rpms
  - Windows (MSVC 2017)
- Required Dependencies
  - Boost, Eigen, Python, Numpy

Simplifying Spin Wave Calculations

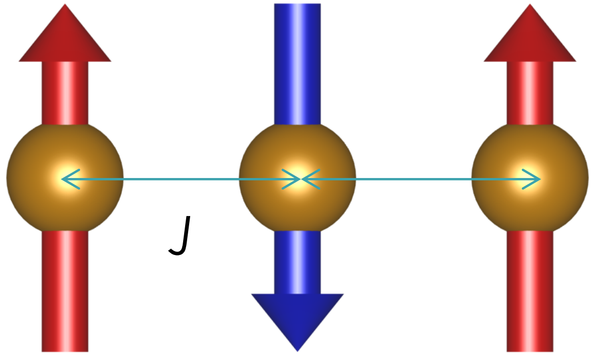
- Python interface
  - Written in C++
- Abstraction & Encapsulation
  - Generate unit cell

```
cell = swg.Cell()
cell.setBasisVectors(1.0,10.0,10.0,90.0,90.0,90.0)
```

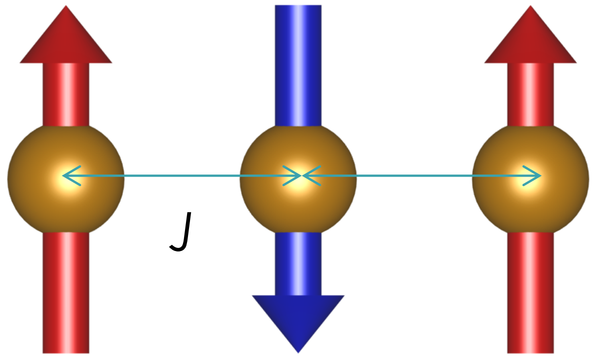
  - Cell contains Sublattices

```
Spin0 = swg.Sublattice()
spin0.setName("Spin0")
spin0.setMoment(1.0,0.0,0.0)
cell.addSublattice(Spin0)
spin1 = swg.Sublattice()
spin1.setName("Spin1")
spin1.setMoment(1.0,180.0,0.0)
cell.addSublattice(spin1)
```

pybind11



Simplifying Spin Wave Calculations



- Sublattice contains atoms

```
cell.addAtom(“Spin0”,0.0,0.0,0.0)
cell.addAtom(“Spin1”,0.5,0.0,0.0)
```

- Interactions are between sublattices

```
builder = swg.SpinWaveBuilder(cell)
interactions = swg.InteractionFactory()
direction = [1.0,0.0,0.0]
builder.addInteraction(interactions.getAnisotropy(“D”,1.0,direction,”spin0”))
builder.addInteraction(interactions.getAnisotropy(“D”,1.0,direction,”spin1”))
builder.addInteraction(interactions.getExchange(“J”,-1.0,”spin0”,”spin1”,0.49,0.51))
```

- Calculating spin waves

```
afm = builder.createElement()
afm.createMatrix(0.1,0.0,0.0)
afm.calculate()
results = afm.getResults()
```

Adding interactions

- Inherit from abstract base class

```
class Interaction {

public:
    …
    virtual void calculateEnergy(const Cell &cell, double &energy) = 0;

    virtual void calculateFirstOrderTerms(const Cell &cell, Eigen::VectorXcd &elements) = 0;

    virtual void calcConstantValues(const Cell &cell) = 0;

    virtual void updateMatrix(const Eigen::Vector3d &K, Eigen::MatrixXcd &LN) const = 0;
};
```

- InteractionsFactory

Available Interactions

- Isotropic Exchange

$$H_{exch} = -\frac{1}{2}\sum_{i\neq j} J_{ij}\mathbf{S}_i \cdot \mathbf{S}_j$$

- Dzyaloshinskii-Moriya

$$H_{DM} = -\frac{1}{2}\sum_{i\neq j} \mathbf{D}_{ij} \cdot \mathbf{S}_i \times \mathbf{S}_j$$

- Single-Ion Anisotropy

$$H_{anis} = \sum_{i} K_i \left(\hat{\boldsymbol{u}}_i \cdot \mathbf{S}_i\right)^2$$

- External Magnetic Field

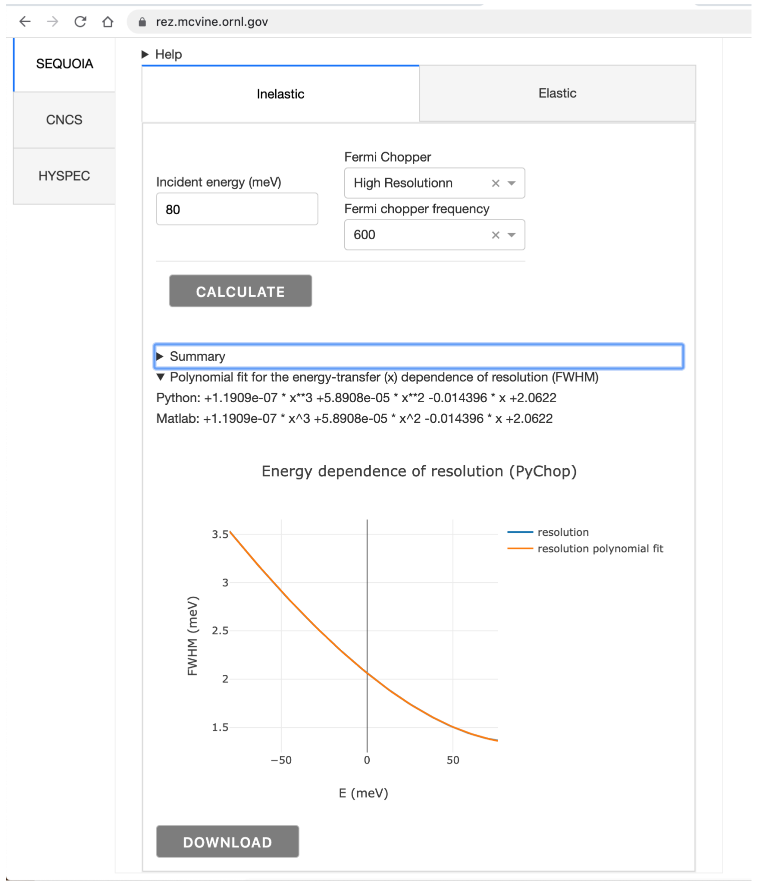
$$H_B = -\mathbf{B} \cdot \sum_{i} \mathbf{S}_i$$

Resolution Function

```
class OneDimensionalShapes {
public:
  virtual void setTolerance(double InTolerance) = 0;

  virtual double getMinimumEnergy() = 0;

  virtual double getMaximumEnergy() = 0;

  virtual void setFrequency(double frequency) = 0;

  virtual double getFunction(double energy) = 0;
};
```

- OneDimensionalFactory

- EnergyDependentGaussian
  - Parameters for ARCS, SEQUOIA, CNCS
  - https://dgsres.mcvine.org



Incident energy (meV): 80
Fermi Chopper: High Resolutionn
Fermi chopper frequency: 600

Polynomial fit for the energy-transfer (x) dependence of resolution (FWHM)

```
Python: +1.1909e-07 * x**3 +5.8908e-05 * x**2 -0.014396 * x +2.0622
Matlab: +1.1909e-07 * x^3 +5.8908e-05 * x^2 -0.014396 * x +2.0622
```

Post processing

```
class SpinWavePlot
{
public:
   …
   virtual const Cell &getCell() const = 0;

   virtual const Energies &getEnergies() = 0;

   virtual void setEnergies(const Energies &energies) = 0;

   virtual std::vector<double> getCut(double kx, double ky, double kz) = 0;
};
```

- Combine multiple functions by making one the input of other

Additional Post-Processing

- Convolution with resolution function

$$I(\boldsymbol{Q}_0,\omega_0)=\iint F_Q^2\,S\,(\boldsymbol{Q},\omega)\,R\,(\boldsymbol{Q}-\boldsymbol{Q}_0,\omega-\omega_0)\,d\boldsymbol{Q}\,d\omega$$

- Integrating over unseen directions

$$I(Q_x,\omega)=\frac{1}{A}\iint I\,(\boldsymbol{Q},\omega)\,dQ_y\,dQ_z$$

- Powder Averaging

$$I(Q,\omega)=\int\frac{d\Omega_{\hat{q}}}{4\pi}I(\boldsymbol{Q},\omega)$$

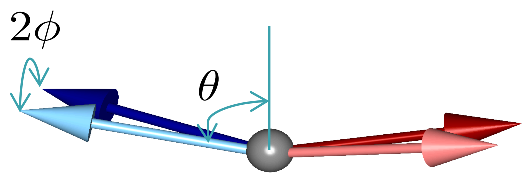
- Constant-energy cuts

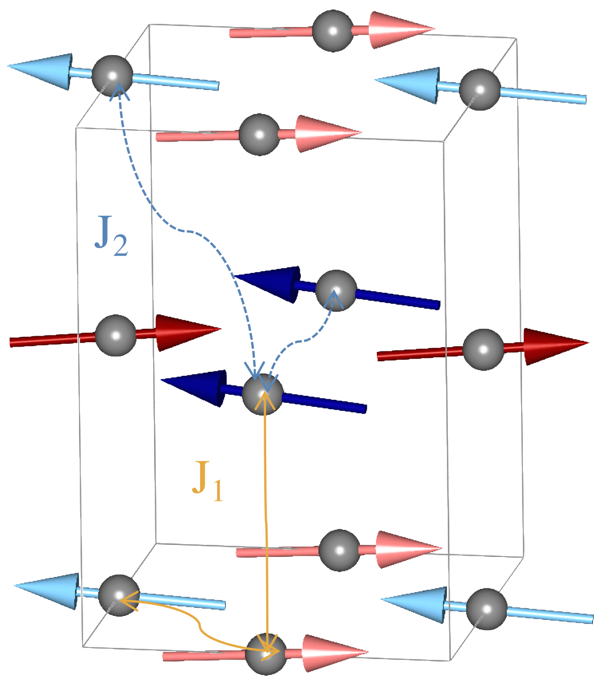
Output Formats

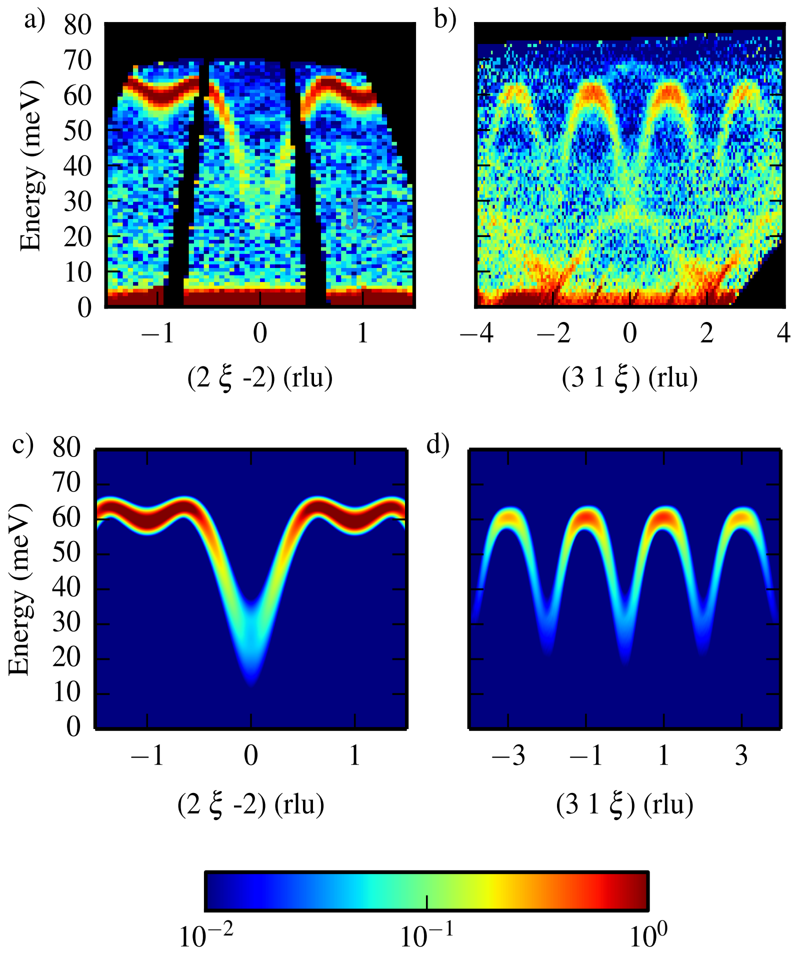
- 1 Dimension
  - Frequency/intensity pair
  - NumPy array
- 2 Dimensions
  - NumPy array
  - ASCII file
- 3 Dimensions
  - vtkStructuredGrid

YFeO$_3$

$$
\begin{aligned}
H = &- J_1 \sum_{\langle i,j \rangle} \boldsymbol{S}_i \cdot \boldsymbol{S}_j - J_2 \sum_{\langle i,j \rangle'} \boldsymbol{S}_i \cdot \boldsymbol{S}_j \\
&- D_1 \sum_{\boldsymbol{R}_j = \boldsymbol{R}_i + a(\hat{x} \pm \hat{y})} \hat{y} \cdot \boldsymbol{S}_i \times \boldsymbol{S}_j \\
&- D_2 \sum_{\boldsymbol{R}_j = \boldsymbol{R}_i + a(\hat{x} \pm \hat{y})} \hat{z} \cdot \boldsymbol{S}_i \times \boldsymbol{S}_j \\
&- K_a \sum_i (\boldsymbol{S}_i^x)^2 - K_c \sum_i (\boldsymbol{S}_i^z)^2
\end{aligned}
$$



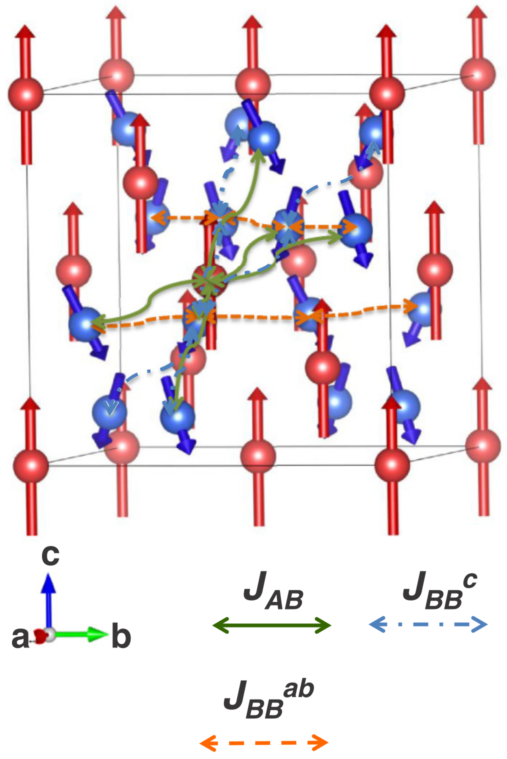


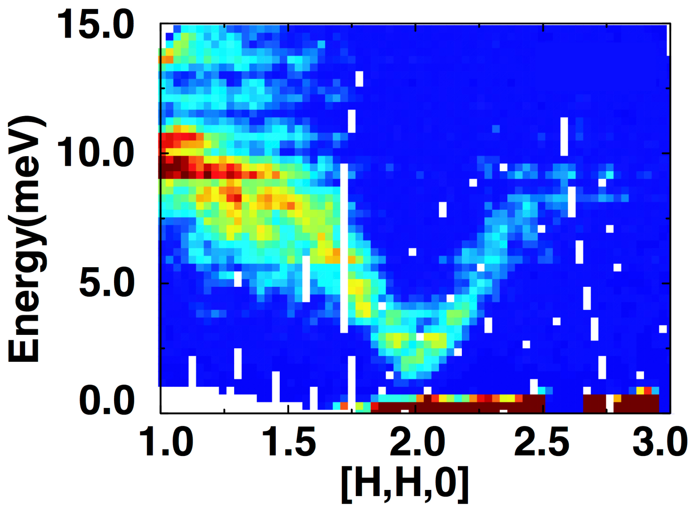


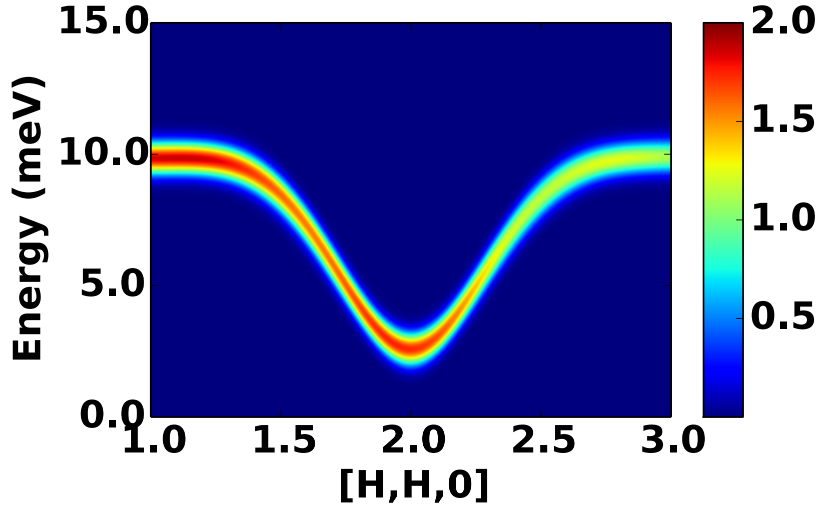
S E Hahn et al, PRB 89, 014420 (2014)

$Mn_{1-x}Co_xV_2O_4$

$$
\begin{aligned}
H = &-J_{AB} \sum_{(p,q)(i,j,k,l)} (\boldsymbol{S}_p + \boldsymbol{S}_q) \cdot (\boldsymbol{S}_i + \boldsymbol{S}_j + \boldsymbol{S}_k + \boldsymbol{S}_l) \\
&- J_{BB}^{ab} \left(\sum_{i,j} \boldsymbol{S}_i \cdot \boldsymbol{S}_j + \sum_{k,l} \boldsymbol{S}_k \cdot \boldsymbol{S}_l \right) \\
&- J_{BB}^{c} \sum_{(i,j)(k,l)} (\boldsymbol{S}_i + \boldsymbol{S}_j) \cdot (\boldsymbol{S}_k + \boldsymbol{S}_l) \\
&+ D_A \sum_{r=p,q} (\hat{z} \cdot \boldsymbol{S}_r)^2 + D_B \sum_{s=i,j,k,l} (\hat{u_s} \cdot \boldsymbol{S}_s)^2.
\end{aligned}
$$

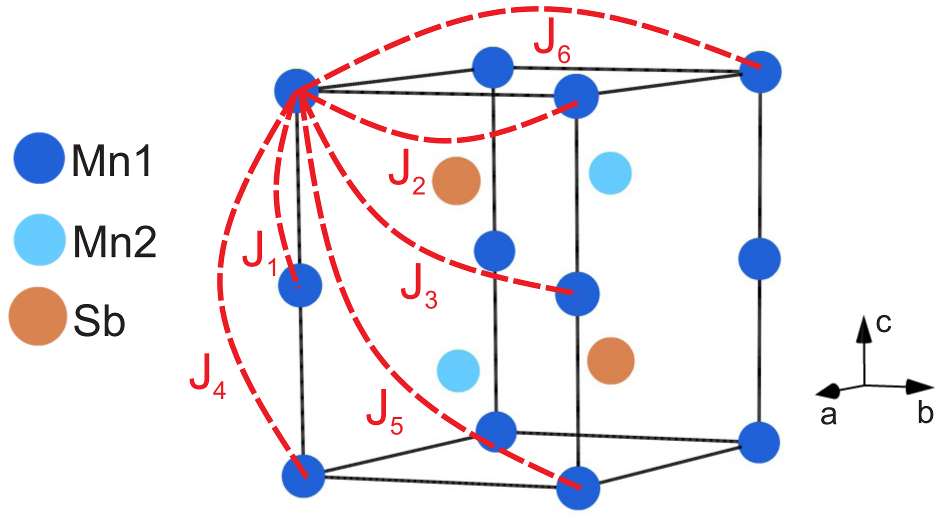




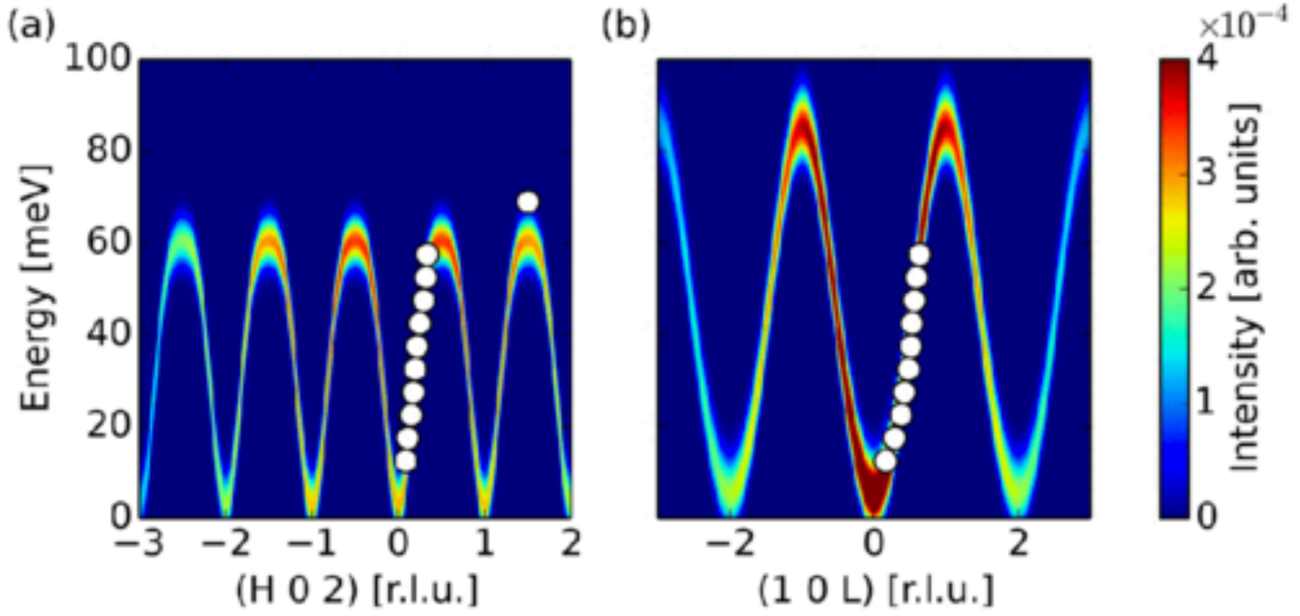


J. Ma, *et al.*, Phys. Rev. B **91**, 020407(R) (2015)
J. H. Lee *et al.*, Scientific Reports **7** 17129 (2017)

$Mn_{1+x}Sb$



$$H_{exch} = -\frac{1}{2}\sum_{i\neq j} J_{ij}\mathbf{S}_i \cdot \mathbf{S}_j$$

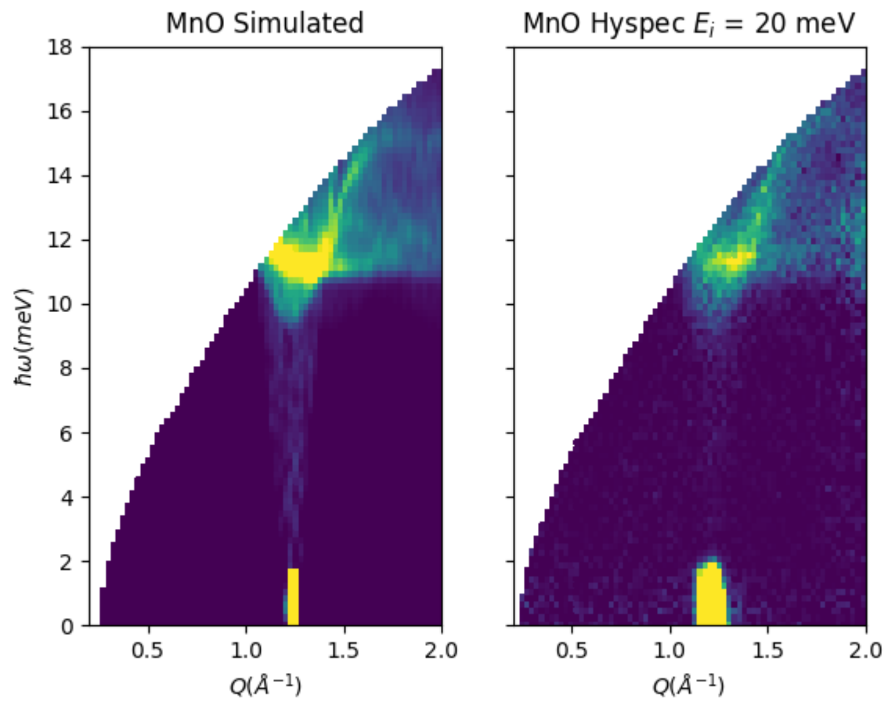


A. E. Taylor *et al.*, Phys. Rev. B **91**, 224418 2015

MnO

- Unit cell setup with Atomic Simulation Environment

- Powder calculation using Monte Carlo Integration



G. Pepy J. Phys. Chem. Solids. 1974. Vol. 35. pp. 433-444.
G. E. Granroth *et al*., in progress

Conclusions

- SpinWaveGenie abstracts and automates large portions of the work required to perform a detailed spin wave analysis in non-collinear systems.
  - Reduces time between measurement and publication
  - Opens inelastic neutron scattering to a larger community of scientists and engineers
- Post-processing routines for direct comparison with experimental data

Questions?

- Https://GitHub.com/SpinWaveGenie/SpinWaveGenie

S.E.H. acknowledges support by the Laboratory's Director's fund, Oak Ridge National Laboratory. This research used resources at the High Flux Isotope Reactor and Spallation Neutron Source, a DOE Office of Science User Facility operated by the Oak Ridge National Laboratory.