JaSS code manual
(version 1.7)

December 24, 2019

Contents

1 Conventions

Physical units

JaSS uses following conventions for physical units. The atomic unit is used for the length, 1 a.u. = 0.52917720859(36) Å. 1 Ry = 13.605692 eV. 1 eV = 11600 K.

Notational conventions

This manual uses the following conventions. Filenames are in ***brick-red bold italic font***. All files with an ***.inp*** extension are input files. Files with a ***.out*** extension are output. Tokens are in *dark-blue skewed font*. Executables are in ***dark-green bold italic font***.

2 Theoretical preliminaries

The Heisenberg model used in the JaSS is written in the form

$$H = \sum_{i>j} J_{ij} \mathbf{S}_i \mathbf{S}_j. \tag{1}$$

If exchange constants are to be compared with models, where the summation runs twice over each pair of indexes (i.e. $\sum_{ij} \tilde{J}_{ij} \mathbf{S}_i \mathbf{S}_j$), then $2\tilde{J}_{ij} = J_{ij}$.

There exist a number of possible ways how these exchange constants can be calculated. In JaSS we realized three of them: the total energy, pairs and Green's function methods. Below we briefly describe these methods.

Total energy method

We assume that results of the density functional theory (DFT) calculation can be mapped onto classical Heisenberg model written in the form (1). Then the total energy difference, δE, between ferromagnetic (FM) and antiferromagnetic (AFM) solutions for an isolated pair of spins is $\delta E = 2JS^2$ and $J = \delta E/(2S^2)$. If there is a spin lattice instead of a pair spin and only the exchange interaction between nearest neighbors J is nonvanishing, then

$$J = \frac{E_{\uparrow\uparrow} - E_{\uparrow\downarrow}}{2zS^2}, \tag{2}$$

where z is the number of nearest neighbors. Thus, we see that in order to calculate one exchange parameter one needs to know the total energies of two magnetic configurations.

In general there can be N exchange paths J_{ij}. In order to calculate all of them one needs to find total energies of *at least* $N+1$ magnetic configurations and again express these total energies via J_{ij} and S^2. This is exactly what JaSS does. There are many technical details though, e.g.

- Some of these expressions for the total energies can be useless (linear dependent). E.g., in the simplest case of two spins one may calculate 4 configurations ($\uparrow-\uparrow$, $\uparrow-\downarrow$, $\downarrow-\uparrow$, and $\downarrow-\downarrow$), but only two of them are linearly independent (compare with the pairs method). JaSS automatically choses only linearly independent configurations for calculations.

- It may turn out that DFT calculations for some of the configurations either do not converge or converge to such a state, when magnetic moments are very different from what we have

in case other configurations. Such odd configurations cannot be used for calculations of J. JaSS takes care of these situations and analizes output files of DFT calculations on this issue;

- In methods like DFT+U (aka LDA+U or GGA+U) different local minima of the density functional may exist and different configurations in principle may stack at different local minima. In general such configurations also cannot be used for exchange (but there are exceptions, where indeed magnetic order is implicitly related with an orbital order).

Pairs method

In order to obtain exchange interaction parameters one can focus on single coupling J_{mn} between spins on m and n sites. Following Ref. [5] one can rewrite (1) in the following form:

$$H = J_{mn}\mathbf{S}_m\mathbf{S}_n + \mathbf{S}_m\mathbf{K}_m + \mathbf{S}_n\mathbf{K}_n + E_{other}, \tag{3}$$

where

$$\mathbf{K}_m = \sum_{i\neq m,n} J_{mi}\mathbf{S}_i,$$
$$\mathbf{K}_n = \sum_{i\neq m,n} J_{ni}\mathbf{S}_i,$$
$$E_{other} = \sum_{i,j\neq m,n} J_{ij}\mathbf{S}_i\mathbf{S}_j.$$

It should be noted that $\mathbf{K}_m$, $\mathbf{K}_n$ and E_{other} do not depend on the spin directions on sites m and n. Let's consider four collinear spin configurations (with the spins along z quantization axis), where $\uparrow_m$ or $\downarrow_m$ denote spin direction on site m:

$$\begin{array}{llllll} 1)\ldots & \uparrow_m & \ldots & \uparrow_n & \ldots \\ 2)\ldots & \uparrow_m & \ldots & \downarrow_n & \ldots \\ 3)\ldots & \downarrow_m & \ldots & \uparrow_n & \ldots \\ 4)\ldots & \downarrow_m & \ldots & \downarrow_n & \ldots \end{array}$$

In these four spin states, the spin orientations on sites other than m and n are the same. These four states have the following energy:

$$E_1 = E_{other} + J_{mn}S^2 + K_mS + K_nS,$$
$$E_2 = E_{other} - J_{mn}S^2 + K_mS - K_nS,$$
$$E_3 = E_{other} - J_{mn}S^2 - K_mS + K_nS,$$
$$E_4 = E_{other} + J_{mn}S^2 - K_mS - K_nS.$$

Then one can obtain the expression for J_{mn} as

$$J_{mn} = \frac{E_1 - E_2 - E_3 + E_4}{4S^2}. \tag{4}$$

Computationally, this approach is rather inefficient since the calculation of N exchange interactions requires the energies of $4N$ spin configurations. But in complex magnets it turns out to be very useful for analysis of multiple, heavily intertwined exchanges.

Green's function method

The first step to calculate exchange interaction parameters using the Green's function method is to project the DFT Hamiltonian onto the basis of some site-localized orbitals. These could be Wannier functions or e.g. PAW projectors, which we use in integration with VASP.

Having Hamiltonian $H^{WF}_{nm,\sigma}(\mathbf{k})$ written in the basis set of localized wavefunctions one can calculate the intersites Green's function, describing scattering process of an electron with spin σ between m and m' orbitals from i to j site at every $\mathbf{k}$ point in reciprocal space, as

$$G^{mm'}_{ij,\sigma}(\epsilon, \mathbf{k}) = [\epsilon + E_F - H^{WF}_{nm,\sigma}(\mathbf{k})]^{-\mathbf{1}},$$

where E_F is the Fermi energy.

The last step is to calculate exchange integrals using the following expression:

$$J_{ij} = -\frac{1}{2\pi}\int_{-\infty}^{E_F} d\epsilon \sum_{mm'm''m'''} Im(\Delta^{mm'}_i G^{m'm''}_{ij,\downarrow} \Delta^{m''m'''}_j G^{m'''m}_{ji,\uparrow}),$$

where m, m' are the orbital indexes, and $G^{m'm''}_{ij,\downarrow(\uparrow)}$ is Green's function integrated over the Brillouin zone and

$$\Delta^{mm'}_i = \int_{BZ} [H^{mm'}_{ii,\uparrow}(\mathbf{k}) - H^{mm'}_{ii,\downarrow}(\mathbf{k})]\, d\mathbf{k},$$

where $\Delta^{mm'}_i$ is the spin splitting.

The detailed specification of Green's function method one can find in Ref. [6].

3 Installation

To install JaSS code package you must have installed Python 3.x with the following modules: ***numpy***, ***sympy***, ***pymatgen*** and ***matplotlib***.

One does not need to compile the code, but instead it must be configured to be able to work with all external program packages (VASP, wien2k, LMTO, ALPS). See next section for description of the configuration file.

Configuration file: config.py

This file contains default values for various variables used by JaSS.

First of all, one may need to change here settings for execution of the external codes. These are *runvasp*, *runwien2k*, *runlmto*, and *runlmto_leip* variables. By default they are set up for the SLURM, but one may easily change them on any other workload manager. One may also change these variables in the ***jass.inp***, see Sec. 5, for any particular run.

Second, there are internal parameters used by JaSS algorithms. Typically one does not need to change them, but we still list them below:

nscell (integer, default = 3)
Number of cells on which we translate the unit cell along each translation vectors (a, b, and c) to find equivalent exchange paths.

precd (real, default = 0.005)
Precision, in $\mathring{A}$, used to compare meta-metal distances, while searching for equivalent exchanges.

J_dist_cutoff (real, default = 15 .0)
Cut-off distances, in $\mathring{A}$, for metal-metal bonds for which exchanges can be calculated.

4 Interaction with external codes

Currently JaSS is able to deal with three DFT codes: TB-LMTO-ASA, VASP[1], and the full potential Wien2k[2].

LMTO

VASP

We recommend to consult wiki version of the VASP manual on details of the installation, program flow, structure of input files and meaning of different parameters. Here we present a short description of the input files, which JaSS needs to calculate exchange parameters using VASP.

- INCAR - the main input file, where all control parameters are set up.
- KPOINTS - input file with information about $k-$mesh, having the form of

```
Automatic mesh
0
Gamma
 4  4  2
 0  0  0
```

  For a chosen system we advice simply to change k_x, k_y, and k_z, given in the 4th line, accordingly.

- POSCAR - file where the structural information is kept. We recommend to use Vesta[3] to create this file.
- POTCAR - file with pseudopotentials. Can be easily created using command like

```
cat ~/pot/Al/POTCAR ~/pot/C/POTCAR ~/pot/H/POTCAR >POTCAR.
```

ALPS

Wien2k

It is necessary to use the official userguide for detailed information about installation process, scripts and program flow and input/output files.

To perform GGA calculation you need only

- ***case.struct*** - the main input file defining the structure and most of control parameters for all Wien2k scripts.

If you are going to perform DFT+U calculations (for the case of strongly correlated systems) one must also have the following files to the directory with the ***case.struct***.

- ***case.inorb*** and ***case.indm*** - two files with information about atoms and orbitals you want to add to orbital potential with proper Hubbard U and Hund's J_H values for them.

5 JaSS input files

To start calculations with the JaSS code you need the only input file ***input.cfg***. Besides, there must be it in the folder must be input files for the chosen *ab initio* code. Below you can find the description of the JaSS input file.

jass.inp file

There are theree main sections in the ***jass.inp*** file: [main], [DFT], [heisenberg]. The first one is related with execution options, the second one consists of keys regulating options for the exchange calculations using DFT, the last one sets up parameters for solvers of the Heisenberg model. Order of sections and keys within a section is unimportant. In the example presented below one can see that it is in a free format form. Comments (! or #) can occur at any place.

```
[main]
code      = VASP
verbosity = 50
runcalculation = yes

[DFT]
mode = autodft
method = energies
S = 1
magion = Ni
```

[main] Execution options

code (default = VASP)
– the *ab initio* code, which is selected for calculation. Possible options: ***vasp***, ***lmto***, ***wien2k***.

ncore (integer, default=1)
– number of cores for parallel execution.

verbosity (integer, default = 30)
– the key assigned to determine an amount of information in the output file. The larger number is the longer ***output.dat*** file is.

export_xml (logical, default = false)
– If true, then JaSS produces xml file describing spin lattice, which can be used, e.g., for the Luttinger-Tizsa calculations.

mode (string, default = autodft). This key sets up the program flow. Possible values are

- ***autodft***: automatically calculates J using chosen *method* and then stops;
- ***auto***: automatically calculates J and continues with ALPS;
- ***alps***: runs only ALPS (exchanges must be set up);
- ***asis***: tries to find all possible exchanges from those calculations, which has been converged (i.e. throws away those bad or non self-consistent configurations). It can be used only if *method* = ***energies***;
- ***distances***: only calculates distances and then stops;
- ***genconf***: only generates configurations and then stops;
- ***lt***: runs Luttinger-Tisza calculations, see also Sec. 10;
- ***submitdft***: submit all DFT tasks needed to calculate J and stops;
- ***supercell***: creates supercell using key ***scell***.

runvasp, ***runwien2k***, ***runlmto***, ***runlmto_leip*** (string)
– Variables, which set up how external programs are executed by JaSS. Default values are defined in ***config.py*** file, see Sec. 3, but here one may redefine them for any particular run. E.g., one may set up

```
runvasp        = srun -n $NCORE -p hobbits vasp&
```

to run VASP using SLURM workload manager on a partition hobbits. For NCORE variable JaSS will use the setting from *ncore* option described above.

[DFT] DFT options used to calculate exchanges

magion (string, **required**, default = Co)
– Magnetic ions in DFT calculations for which exchanges should be calculated.

method (string, default = energies).
– This key defines the method which will be used for calculation of exchanges. Possible values are ***energies*** (Total energy method), ***pairs*** (Pairs method), and ***greens*** (Green's function method). For detailed description of each method see Sec. 2.

S (float, default=1.0)
– spin of the magnetic ion (current version of JaSS supports only one type of magnetic ions).

i (integer, default=1)
– index of first magnetic ion for the pairs method calculations.

j (integer, default=1)
– index of second magnetic ion for the pairs method calculations.

numJ (integer, default=10)
– maximal number of exchanges to be calculated by the total energy method.

JtoCalc (list of integers, default=0)
– one can force the code to calculate exchanges not in ascending (with respect to distances) order, but those given in this list. Exchanges should be given in the list separated by commas, e.g. JtoCalc = 0,1,3 means that JaSS should calculate exchange between nearest neighbors (0), than between 2nd and 4th nearest neighbors (1 and 3), but skip exchange between 3rd neighbors.

JtoDel (list of integers, default=0)
– list of exchange constants to be skipped in search of magnetic configurations for *mode* = ***energies***. Exchanges should be given in the list separated by commas. See also Sec. 6 for the detailed description how this tag can be used in real calculations.

scell (list of integers, default=1,1,1)
– list of multipliers used to increase the unit cell (needed to *mode* = ***supercell***) order of the real harmonics in Hamiltonian.

[heisenberg] Options for solver of the Heisenberg model

L (integer, default value is 8)
– Now many unit cells are used for simulations of the spin model (length of spin lattice)

Method (string, default value is spin)
– Name of the solvers used to solve the Heisenberg model. Possible values are
spin - classical Monte Carlo algorithm
loop - multi-cluster quantum Monte Carlo algorithm
sparse-ED - exact (sparse) diagonalization (Lanczos)
full-ED - exact (full) diagonalization

Sweeps (integer, default value is 100000)
– Name of the solvers used to solve the Heisenberg model

Thermalization (integer, default value is 5000)
– Name of the solvers used to solve the Heisenberg model

Tmax (integer, default value is 300)
– Maximal temperature in K

Tmin (integer, default value is 10)
– Minimal temperature in K

dT (integer, default value is 50)
– Step in temperature in K

[wien2k] Options for Wien2k DFT code

w2k_prefix (string, default value is "case")
– The name of your struct-file. For example, "NiO" in "NiO.struct"

lstart_energy (float, default = -6.0)
– The value of energy to separate core and valence states in Ry; needed for lstart.

numk (integer, default = 200)
– The number of k-points in the full Brillouin zone.

6 Notes and lifehacks

Use of JtoDel

This applies to *mode = energies*. The strategy used by JaSS is to remove configurations, but do this in a way to calculate as many exchange constants as possible. At some point one needs to decrease number of exchanges as well. This is done by removing exchanges corresponding to the longest exchange paths first. But the problem can be not due to these far exchanges and even not due to "bad" configurations, but because of "bad" nearest neighbor exchanges. These are exchanges, which are met the same number of times (and with the same signs) in all configurations and thus can't be calculated by the total energy method. As a result JaSS decreases number configurations and exchanges until these "bad" exchanges disappear (from the expressions for the total energies) and one can easily left out with only a few Js even if number of magnetic ions is rather large.

The solution of the problem can be to study ***jass.out*** and in particular lines after

```
Search for possible magnetic configurations to calculate
```

JaSS prints out after these lines exchanges which can't be calculated at the same time at each iteration of the configuration minimization. If there is a certain exchange, e.g., $J2$ is met all along minimization process, then this is the "bad" exchange. The suggestion would be to include this exchange to *JtoDel* and rerun JaSS. It well may be that now you will be able to calculate much more exchange constants.

Symmetry in GGA+U

There always can be interplay between spin and orbital degrees of freedom, e.g. via Kugel-Khomskii like coupling. Such a situation may realize in GGA+U calculations, where U part is orbital dependent. This is why a symmetry of electronic subsystem can be **lower** than crystalline. On the one hand, the use of the symmetry typically substantially decreases time of a calculation and improves convergency process, but on the other hand it may substantially restrict types of possible DFT+U solutions. This may lead in a situation when solution without any symmetry restrictions would have lower energy than the symmetric one. As a result exchange parameters can be wrong, if even in only one of the magnetic configurations orbitals degrees of freedom turned out be strongly coupled with spin ones. Thus, it is strongly recommended to perform DFT calculation without any symmetry, i.e., e.g., in VASP we suggest to use ISYM=0 tag.

VASP: from GGA to GGA+U

As it has been mentioned before, GGA+U is known to have numerous local minima (in contrast to GGA). Thus, one of good strategies could be to perform GGA calculation first and

then switch on U. This can be easily done in JaSS. One needs to copy whole directory with all JaSS calculations (including all subfolders like 0, 1, 2 etc.). Modify INCAR file accordingly (add all U related tags and ICHARG=1), remove OUTCAR files from all subfolders (i.e. run ***'rm */OUTCAR'***) and execute ***jass***. Then JaSS will start VASP with self-consistent GGA charge density, which strongly helps in convergency.

VASP: changing KPOINTS, U etc.

JaSS watches the parent directory and if you change number of k-points will overwrite KPOINTS files in all subdirectories and resubmit the calculations (since now they are considered as non-selfconsistent).

However, JaSS can NOT trace changes in INCAR, since this file is modified by the program. Thus, in order to run JaSS with modified INCAR one needs to ***'rm */OUTCAR'*** and then run ***jass***.

Wien2k: from GGA to GGA+U

Default type of JaSS calculation with using of Wien2k is GGA. If it is necessary to perform GGA+U calculation one has to put ***case.inorb*** and ***case.indm*** in the root directory together with ***case.struct*** and ***jass.inp***. This action leads to appearing of ***-orb*** and ***-dm*** keys in the ***Asbatch*** - the file containing execution string for each magnetic configuration.

7 Additional files

custom_configurations

File custom_configurations can be used if one needs to add ("manually") an additional configuration with a certain energy. Example of the file

```
1 1 -1 -1; -100.345
```

I.e. we add configuration 1 1 -1 -1 with the energy -100.345 eV.

block_configurations

File block_configurations can be used to block some of the configurations from the exchange calculations. Example of the file

```
-1 1 1 -1; converges to zero moments
1 -1 -1 1; converges to zero moments
```

As usual 1 and -1 denote spin up and down and after ";" one may add any comment.

8 Executables

JaSS consists of several executable routines and control files. The main program is ***jass***. It can be executed as easy as

```
[gandalf@jass_user]>  jass
```

This program requires a single input file ***jass.inp***, see Sec. 5, and corresponding starting files for DFT calculations (consult Sec. 4).

9 Output files

jass.out file

jass.out is a main output file and it is self-explanatory.

exchanges.xml file

It contains results of the exchange calculations - exchange constants, which can be used in postprocessing, e.g. as input for Luttinger-Tizsa method.

LT.xml file

Input information for ***PlotLT***. ***LT.xml*** contains all information about exchange interaction needed for the Luttinger-Tizsa calculations.

10 Postprocessing

Luttinger-Tizsa method

In frustrated systems magnetic ground state often turns out rather nontrivial and its magnetic unit cell does not necessarily coincide with crystallographic one. As a result various commensurate or incommensurate spin spiral may develop in such magnets. One of the approaches, which helps to find $\mathbf{Q_{min}}$ vector corresponding to true magnetic ground state is the Luttinger-Tizsa method[7, 8]. ***PlotLT*** postprocessing script can be used for this.

It uses information about exchange interaction saved by JaSS in ***LT.xml*** (exchange topology) files to plot $\varepsilon(\mathbf{q})$ and find its global minima.

One may also use ***PlotLT*** without performing calculation of exchange constants at all. For this one needs to generate (or copy and modify) ***exchanges.xml*** file and run JaSS with *mode*=lt. This will generate input ***LT.xml*** for ***PlotLT***. Now you can simply put your own exchange constants and perform the Luttinger-Tizsa analysis.

High symmetry direction for the band plotting can be easily modified in ***PlotLT*** script. By default they are set up as:

```
J.plot_Q_vs_J([[0, 0, 0], [0.5, 0.5, 0.5], [0.5, 0., 0.0 ] ], 20)
```

where 20 is the number of **q**-points along each of the high-symmetry lines.

11 Tutorials

Tutorial 1: Calculation of exchange parameters using VASP and the total energy method

The goal of this tutorial is to illustrate how one can calculate exchange parameters using the total energy method, see Sec. 2, and pseudopotential VASP code for $KCuF_3$. This is famous material, which has three-dimensional crystal structure, but due to a specific orbital ordering magnetic subsystem in this compound turns out to be one-dimensional [4]. In order to get correct (insulating) ground state one needs to use GGA+U method, which takes into account strong on-site electronic correlations. We chose $U = 7.6$ eV and $J_H = 1$ eV following Ref. [10].

As explained in Sec. 4 one needs to prepare 4 input files for VASP calculations (INCAR, KPOINTS, POSCAR, POTCAR). Whole set of these files is accessible via tutorial archive, but one can easily make them by oneself. We chose the crystal structure to allow so called polytype *a*[9], which contains the unit cell with 4 formula units.

The ***jass.inp*** file can be as simple as

```
[main]
ncore = 24

[DFT]
S = 0.5
magion = Cu
```

Here we only specified that 24 CPUs should be used by VASP and that the magnetic ion is Cu, which spin is $S = 1/2$.

To calculate exchange parameters just execute JaSS in the command line and study ***jass.out***. Program finds that with the given unit cell one may calculate only exchange integrals: J_0, J_1, and J_2. One may see that the largest is $J_0 = 35.7$ meV (AFM). Studying ***jass.out*** we find this exchange corresponds to distance 3.92 Å and it couples two nearest Cu ions along c axis (the translation vectors are $[0, 0, 1]$ and $[0, 0, -1]$). The exchange between nearest neighbors in the ab plane is much smaller and, moreover, it is ferromagnetic: $J_1 = -0.5$ meV, while corresponding distance is on slightly larger (4.14 Å). Thus, as it was explained above, $KCuF_3$ is one-dimensional AFM.

One may compare calculated exchange parameters with those found experimentally using optical, susceptibility and specific heat measurements[11, 12]. One needs to take into account that in corresponding papers slightly different formulation of the Heisenberg model ($2J \sum_{i>j} S_i S_j$[13]) was used, i.e. each pair was counted once (exactly as in JaSS), but the coupling parameter was set to $2J$ instead of J_{jass}. Thus, in order to compare with experiment one needs to divide exchange parameter obtained by JaSS by factor of two. Then we get $J_0 = 17.85$ meV or 207 K. This is very close to $J_0 = 187 - 190$ K obtained experimentally[11, 12].

Tutorial 2: Calculation of exchange parameters using Wien2k and the total energy method

In this tutorial we demonstrate the same total energy method to calculate exchange parameters for the same system $KCuF_3$ [4] but using of full-potential Wien2k code. As in the previous case we choose $U = 7.6$ eV and $J_H = 1$ eV.

To perform GGA+U calculation with Wien2k one needs ***jass.inp***, ***case.struct***, ***case.inorb*** and ***case.indm***. All files you can find in tutorial archive. In order to calculate more than one exchange parameter we made the supercell containing 4 Cu ions.

The ***jass.inp*** of the calculation is:

```
[main]
code      = wien2k
ncore     = 4
verbosity = 50

[DFT]
mode      = auto
method    = energies
S         = 0.5
magion    = Cu
```

Having all calculations finished we investigate the ***jass.out*** which gives us values of three exchange integrals $J_0 = 37.2$ meV (AFM), $J_1 = -0.5$ meV (FM) and $J_2 = 1.2$ meV (AFM) corresponding to nearest neighbours (3.93 Å), next nearest neighbours (4.15 Å) and next next nearest neighbours (5.72 Å). These values are in good agreement with experimental data taking into account the difference in the Heisenberg model formulation.

Tutorial 3: Calculation of exchange parameters using the pairs method

Next we will illustrate how to calculate exchange parameters with the so called pairs method, see Sec. 2. We chose cupric hydroxyhalidate, $Cu_3(OH)_2F_4$ to test it. While, there are many different possible exchange interactions, but the largest is between Cu ions with distance 3.52 Å [14]. All input files can be found at tutorial archive. First, one may calculate all the distances between magnetic ions and simultaneously check whether the pairs method can be used to evaluate exchanges corresponding to these distances. This can be done with following ***jass.inp***:

```
[DFT]
mode = distances
magion = Cu
```

The distance 3.52 Å is between Cu ions with indexes 7 and 9 and JaSS has checked that this exchange can be calculated with the pairs method. To continue one needs to explicitly state that we will use pairs method and indicate these indexes in section exchanges in ***jass.in***:

```
[main]
ncore = 32

[DFT]
method = pairs
ind1 = 7
ind2 = 9
S = 0.5
magion = Cu
```

After JaSS execution we find in ***jass.out*** that $J_{79} = 142$ K.

To do:

- Using the pairs method calculate exchange parameter corresponding to the distance 3.0Å and show that it equals to 11 K;
- Recalculate exchange coupling using the total energy method;

Tutorial 4: Calculation of magnetic susceptibility in $KCuF_3$

In tutorial 1, see Sec. 11, we have computed exchange constants for $KCuF_3$. Here we go further and will use fully automated regime *auto* to calculate magnetic susceptibility. The ***jass.in*** is nearly the same as in tutorial 1:

```
[main]
ncore = 32

[DFT]
mode = auto
S = 0.5
magion = Cu
```

If you run JaSS now it will first compute exchange constants, then construct spin lattice and execute solver for the solution of the Heisenberg model and generate file ***chi.dat***, where magnetic susceptibility is written.

In such a run we chose default method for the solution of the Heisenberg model (quantum Monte-Carlo *qmc*) and used default number of sweeps (*Sweeps*), thermalization sweeps (*Thermalization*), size of the cell (L), minimal (*Tmin*) and maximal temperature (*Tmax*) and temperature step (dT). Alternatively, one may specify them in ***jass.in*** in section Heisenberg:

```
[main]
ncore = 32

[DFT]
mode = auto
S = 0.5
magion = Cu

[heisenberg]
method = qmc
sweeps = 60000
thermalization = 2000
tmax = 600
tmin = 20
L = 8
```

Magnetic susceptibility calculated with this set of parameters is shown in Fig. 1. One may see that it has maximum at ∼250 K. This is in a fairly good agreement with experimental $\chi(T)$ having peak at 243 K[11]. All files are available at tutorial archive.

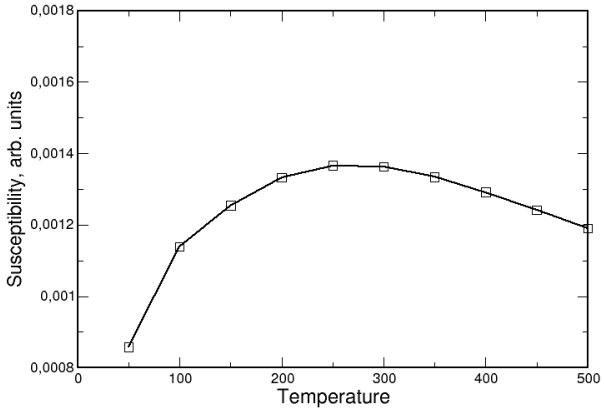

Figure 1: Magnetic susceptibility of $KCuF_3$ as obtained by JaSS.

To do: To get better impression how stable these results are and what parameters can be tuned try to change number of cells used in the QMC method and study low temperature behaviour of $\chi(T)$. Also try to change number of sweeps and choice of the solver of the Heisenberg model.

Tutorial 5: Calculation of exchange constants by Green's function method

For this tutorial we chose the LMTO code and $NaTiSi_2O_6$. This materials has pyroxene crystal structure, which main elements are the transition metal chains. In case of $NaTiSi_2O_6$ they dimerize and one might expect rather large exchange coupling in the Ti-Ti dimer[15].

In order to calculate J_{ij} by combination of LMTO and Green's function method one needs to have LMTO input files. These are CTRL and POCO (since we take into account Hubbard correlations on Ti sites) files. The ***jass.inp*** file can be like this

```
[main]
ncore = 1
code = LMTO

[DFT]
S = 0.5
magion = Ti
```

12 License

The JaSS is a free software for scientific and/or educational purposes and it is distributed under FreeBSD License (see below). To request a source code, a brief description of a planned research has to be submitted to developers at *streltsov.s@gmail.com*. The source code will be e-mailed to you if it is suitable for your study.

13 Team

Following people contributed to development of JaSS: S. Streltsov, V. Gapontsev, P. Igoshev.

14 Acknowledgements

We are grateful to the Russian Science Foundation, which supported development of JaSS through the project 17-12-01207.

References

[1] G. Kresse and J. Furthmüller, Phys. Rev. B **54**, 11169 (1996).

[2] P. Blaha, K. Schwarz, G. K. H. Madsen, D. Kvasnicka, and J. Luitz, WIEN2k, An Augmented Plane Wave + Local Orbitals Program for Calculating Crystal Properties (Techn. Universitt Wien, Wien, 2001).

[3] K. Momma, F. Izumi, J. Appl. Crystallogr. **44**, 1272 (2011).

[4] S. V. Streltsov and D. I. Khomskii, Physics-Uspekhi **60**, 1121 (2017).

[5] H. Xiang, E. Kan, S.-H. Wei, M.-H. Whangbo, X. Gong, Phys. Rev. B **84**, 224429 (2011).

[6] Dm. M. Korotin, V. V. Mazurenko, V. I. Anisimov and S. V. Streltsov, Phys. Rev. B 91, 224405 (2015).

[7] J. M. Luttinger and L. Tisza, Phys. Rev. **70**, 954 (1946).

[8] T. A. Kaplan and N. Menyuk, Philos. Mag. **87**, 6435 (2007).

[9] A. Okazaki, J. Phys. Soc. Jpn. **26**, 870 (1969).

[10] N. Binggeli, M. Altarelli, Phys. Rev. B **70**, 085117 (2004).

[11] S. Kadota, I. Yamada, S. Yoneyama, and K. Hirakawa, J. Phys. Soc. Jpn. **23**, 751 (1967).

[12] K. Iio, H. Hyodo, K. Nagata, and I. Yamada, J. Phys. Soc. Jpn. **44**, 1392 (1978).

[13] J. Bonner and M. Fisher, Phys. Rev **135**, A640 (1964).

[14] I. L. Danilovich, A. V Merkulova, I. V Morozov, E. A. Ovchenkov, F. M. Spiridonov, E. A. Zvereva, O. S. Volkova, V. V Mazurenko, Z. V Pchelkina, A. A. Tsirlin, C. Balz, S. Holenstein, H. Luetkens, A. A. Shakin, and A. N. Vasiliev, J. Alloys Compd. (2018).

[15] S. V. Streltsov and D. I. Khomskii, Phys. Rev. B **77**, 064405 (2008).