HPC ISING MODEL SOLVER: APPLICATION IN MATERIAL DESIGN

Aleksandra Dujović[1,2], Sergiu Arapan[1], Dominik Legut[1], Pablo Nieves[1], and Andrzej P. Kądzielawa[1,3]

1) IT4Innovations, VŠB - Technical University of Ostrava, Ostrava-Poruba, Czechia; 2) Faculty of Physics, Astronomy and Applied Computer Science, Jagiellonian University, Kraków, Poland; 3) Institute of Theoretical Physics, Jagiellonian University, Kraków, Poland

Introduction

The Density-Functional Theorem, used for modeling magnetic materials, usually provides us with a well-defined ground state of a given system, but not its transitional temperatures. For obtaining these properties, we present a generic and high-performance method for **mapping** an **ab-initio system** onto the **Heisenberg model** using our original software, JorGπ [1].

The most crucial part of the algorithm is identifying a set of **metastable states** of the system, achieved through solving a 3D Ising model. This being also the most computationally expensive part of the process, we **parallelized** it – using both distributed and shared memory models.

JorGπ: software for mapping DFT onto the Heisenberg model

Given the ground state of a magnetic structure, JorGπ:

1. processes an ab-initio model of the system,
2. generates a set of metastable states by solving the Ising model,
3. calculates the Heisenberg model from the output.

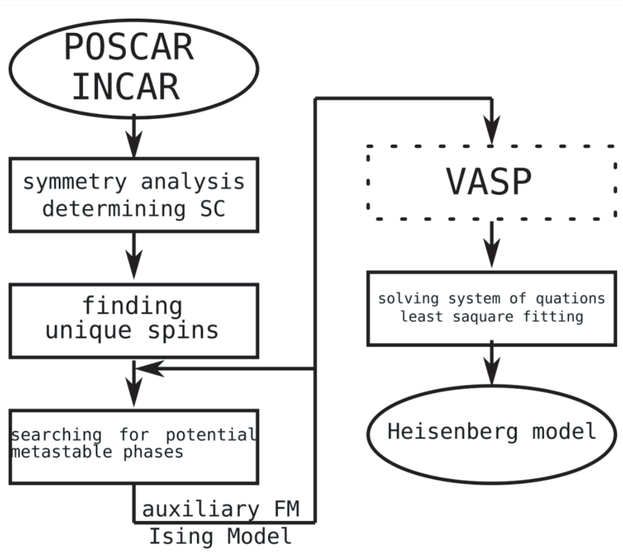


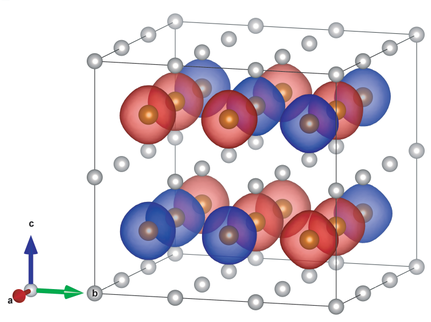
Fig. Workflow of JorGπ [1]

EXEMPLARY RESULTS - FEPT

Space group: P4/mmm (123):

a = b = 2.66084 Å
c = 3.68635 Å
α = β = γ = 90°

	μFe(μB)	μPt(μB)
FLEUR[6]	2.866	0.384
VASP[5]	2.823	0.318
JorGπ	2.849	0.315



	Distance(Å)	FLEUR[2]	SPR-KKR[3]	**JorGπ**	Ref. [4]
J[100](meV)	2.661	25.73	31.52	**35.77**	22.25
J[001](meV)	3.686	8.82	-3.72	**13.28**	1.15
J[110](meV)	3.763	19.89	17.56	**19.24**	18.58
J[101](meV)	4.546	8.20	8.58	**15.67**	16.03
J[111](meV)	5.268	-15.32	-10.58	**-21.07**	-19.75

JorGπ requires Python 3.6 with numpy 1.16.0, scipy 1.0.0, spglib 3.0.0, setuptools 40.8.0, and defusedxml. The Ising model solver is written in C++17 and utilizes GNU Scientific Library, OpenMP, and, optionally, MPI (the Message Passing Interface).

The Heisenberg model

Knowing the Heisenberg model, we can easily estimate e. g. T_C – the **Curie temperature** (above which a material loses its magnetic properties), hence, essentially, the temperatures at which the designed magnet will work.

The Heisenberg Hamiltonian is given by

$$H = -\frac{1}{2}\sum_{i \neq j} J_{ij} s_i s_j ,$$

where J_{ij} are exchange interaction magnitudes of lattice site spins s_i.

- We can find the form of the Hamiltonian (i. e., its coupling parameters – J_{ij}) by identifying its metastable states.
- Furthermore, if $s_i s_j < 0 \Leftrightarrow J_{ij} < 0$, finding the excited states of the **Heisenberg model** is equivalent to finding the excited states of a **strictly ferromagnetic Ising model**.

Solving the Ising model

The **energy** of a spin configuration of the Ising model is described by the Hamiltonian

$$H = -\sum_{i \neq j} J_{ij} s_i s_j ,$$

where J_{ij} are exchange interaction magnitudes of lattice site spins s_i.

Because the metastable states correspond to spin configurations with locally minimal energy, we can find them by minimizing the Hamiltonian.
To accomplish this, JorGπ performs **adaptive simulated annealing** (SA) [5] of a **ferromagnetic 3D Ising model**.

Parallel Solver

Since solving the Ising model is the most computationally expensive part of JorGπ (generating a single metastable state requires running SA at least once), we parallelized it.

- The parallel solver distributes the generation of states between **cluster nodes** via **MPI** (the Message Passing Interface). It employs a coarse-grained communication with SA running separately on each node.
- Additionally, **within** each **node**, loop calculations are parallelized via **OpenMP** using available **CPU cores**.
- To further increase efficiency, we intend to implement a hybrid method that combines parallel simulated annealing with a **genetic algorithm** [6] for improving initial states and, in this way, accelerating convergence.

Acknowledgments

The authors gratefully acknowledge financial support from the Czech Science Foundation through grant No. 20-18392S, as well as the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140).

Poster designed by **Konrad Pasternak**.

References

[1] https://github.com/Mellechowicz/JorG
[2] Stefan Blügel, et al., FLEUR-project, flapw.de/site/ (2019); acc. 30.10.2019.
[3] Hubert Ebert, SPRKKR, ebert.cup.uni-muenchen.de (2019); acc. 30.10.2019.
[4] Oleg N. Mryasov, J.M.M.M. 272-276, 800 (2004).
[5] Hong Guo et al., Physica Scripta. Volume T38, 40-44 (1991).
[6] Ding-Jun Chen et al., J. Glob. Optim. (2007) 39:261–289.