# ILL internship report

Joshua Horswill

September 11, 2020

## Contents

Credits go to Marie-Bernadette Lepetit and Elisa Rebolini from the ILL theory group for their supervision of this project

# 1 Introduction

## 1.1 Aims and Methods

The aim of this project is to create a driver to compute the magneto-electric coupling of a multiferroic system. Multiferroic materials are defined by the possession of a coupling between at least two ferroic orders. The foci of this report are the calculations for materials that exhibit couplings between electric and magnetic properties. Specifically it would demonstrate a net magnetic moment, an intrinsic polarisation and a linear magneto-electric coupling. This is a type II multiferroic system, type I being a material whose transitions from paraelectric and ferroelectric states are distinct from magnetic transitions [8][7].

In order to determine this linear magneto-electric coupling for the type II material experimentally, the associated tensor $\bar{\bar{\alpha}}$ must be measured. This tensor can be described as the response of the polarisation of a system as a function of the applied magnetic field. It can simultaneously be described as the response of the magnetic order parameter as a function of the applied electric field (this parameter is given by $\vec{M} = \sum_j \vec{\mu}_j$ or for an antiferromagnetic material, $\vec{M} = \sum_j (-1)^j \vec{\mu}_j$) where $\vec{\mu}_j$ is the magnetic moment for the jth electron. These responses are linear. The driver processes inputs and outputs of the various ab-initio functions (discussed later) so that it can generate the computation of this tensor $\bar{\bar{\alpha}}$ from a simulation of a system's electronic structure.

This report will discuss specifically how to obtain $\bar{\bar{\alpha}}$. Quantum chemical background theory will be provided, discussing the many body Schrödinger equation, Hartree-Fock methods and density functional theory. This will hopefully generate a better picture of the context surrounding the current model for multiferroic systems. Configurational interactions (CI) and the various associated methods will be investigated, including techniques that increase the accuracy and computational speed of the CI calculations. Finally, the development, mechanisms and purpose of the driver algorithms will be explored.

This work was performed under the supervision of Marie-Bernadette Lepetit and Elisa Rebolini in the ILL theory group, in order to increase the automation and convenience of the $\bar{\bar{\alpha}}$ calculation.

## 1.2 Possible Applications and Reasons for Investigation

The understanding and manipulation of this enigmatic magnetoelectric coupling can lead to many possibilities in electronics, for example in data storage. The electrically-induced change in the magnetization of a storage device allows the use of a small voltage rather than a large current in allocating 'magnetic information'. The rate of change of the logic state of the device in response to an input can become sub-nanosecond, and due to the low voltage the energy cost is similarly small. The electric polarisation of the material stabilises the allocated data through the displacement of the atoms. There are likely many more applications for this in the future that are yet to be apparent, which is why probing the properties of rare materials is so important. For example, discoveries such as graphene, silicon semiconductors, carbon fiber/nanotubes and metallic foams have been essential to modern technological development and offer vastly more design opportunities than before. Hence, it is arguably necessary to place more importance on research into unique-property materials if the same or increased rate of development is desired.

## 2 Relation of the $\bar{\bar{\alpha}}$ tensor to the exchange integral $J$

In order to generate an expression for the magneto electric coupling, one must look to assertions made by soviet physicist Lev Landau. Landau introduced a physical theory that creates a general model of continuous and therefore second-order phase transitions. It is versatile in that it can be used in systems that are subject to external fields. He proposed that the free energy of any system should be analytic (continuous and therefore differentiable), and obey the same symmetry as the Hamiltonian. This allows the generation of a phenomenological expression for the free energy as a Taylor expansion in the order parameters - electric field and magnetic field [11].

Using this theory it is possible to write an expression for $\bar{\bar{\alpha}}$ in terms of the second derivative of the free energy with respect to the electric and magnetic fields ($\vec{\mathcal{E}}$ and $\vec{\mathcal{B}}$ respectively):

$$\bar{\bar{\alpha}} = -\left.\frac{\partial^2 \mathcal{F}}{\partial \vec{\mathcal{E}} \partial \vec{\mathcal{B}}}\right|_{\vec{\mathcal{E}}=\vec{0}, \vec{\mathcal{B}}=\vec{0}} \tag{1}$$

For multiferroic systems, low-energy excitations are of a magnetic nature due to the material being a magnetic insulator. This means it is possible to describe this property by implementing an effective Hamiltonian that only takes into account the magnetic degrees of freedom. One of the models that fits these requirements particularly well is the Heisenberg Hamiltonian. For example, let us suppose that a system can be described by this Hamiltonian for the Fermi level (magnetic) properties:

$$\hat{H} = E_0 - \sum_{\langle i,j \rangle} J_{i,j} \hat{\vec{S}}_i \cdot \hat{\vec{S}}_j \tag{2}$$

where $E_0$ is the energy associated with all non-magnetic degrees of freedom and $\langle i, j \rangle$ represents nearest neighbour exchanges (since the interaction is local) between the ith and jth magnetic ions, $J_{i,j}$ is the associated magnetic exchange integral between these atoms, each with atomic spin $\hat{\vec{S}}_i$ and $\hat{\vec{S}}_j$ respectively. By observing the polar magnetic phase in which the magneto-electric coupling occurs, it is possible to write a statistical mechanical equation of state for the free energy ($\mathcal{F} = U - TS$ where U is internal energy, T is temperature and S is entropy):

$$\mathcal{F} = E_0 - \sum_{\langle i,j \rangle} J_{i,j} \langle \hat{\vec{S}}_i \cdot \hat{\vec{S}}_j \rangle - \sum_i g\mu_B \langle \vec{S}_i \rangle \cdot \vec{\mathcal{B}} - \vec{P} \cdot \vec{\mathcal{E}} - TS \tag{3}$$

where $\langle \hat{\vec{S}}_i \cdot \hat{\vec{S}}_j \rangle$ denotes the thermal average of the spin inner product (as seen in equation (2)), $g$ is the g-factor characterising the magnetic moment of the atom, $\mu_B$ is the Bohr magneton and P is the polarisation of the system.

It is known that the thermal probability of a state being occupied, in an energetically gapped system such as this, is varying very quickly as soon as the temperature $T$ becomes slightly larger or smaller than the critical temperature of the transition ($T_c$). The paramagnetic phase is the name assigned to the case where $T > T_c$, where for all magnetic states, thermal energy

$$E_{th} = k_B T \gg E_I - E_0$$

where $E_I$ is the energy of the Ith magnetic state. In this paramagnetic phase, all magnetic eigenstates and their corresponding energy eigenvalues have an equivalent probability

$$P(E = E_I) = \exp(-\beta E_I)/Z \simeq 1/N$$

where N is the number of magnetic eigenstates and Z is the partition function for the system.

Consequently, the free energy $\mathcal{F}$ is dominated by the entropy term. However, in the phase where the system is magnetically ordered ($T < T_c$), the probability $P(E = E_{ground})$ dominates so that $\mathcal{F}$ is dominated not by entropy but by the energetic term $U$. Now we can neglect the entropy contribution in the calculation of $\bar{\bar{\alpha}}$ as soon as the temperature is slightly smaller than $T_c$.

To first order, combining equation (1) and (3) $\bar{\bar{\alpha}}$ can resultantly be written as:

$$\bar{\bar{\alpha}} = \sum_{\langle i,j \rangle} \left.\frac{\partial J_{i,j}}{\partial \vec{\mathcal{E}}}\right|_{\vec{\mathcal{E}}=\vec{0}} \otimes \left( \left.\frac{\partial \langle \vec{S_i} \rangle}{\partial \vec{\mathcal{B}}}\right|_{\vec{\mathcal{B}}=\vec{0}} \cdot \langle \vec{S_j} \rangle + \langle \vec{S_i} \rangle \cdot \left.\frac{\partial \langle \vec{S_j} \rangle}{\partial \vec{\mathcal{B}}}\right|_{\vec{\mathcal{B}}=\vec{0}} \right) \tag{4}$$

This means that the derivative of the exchange integrals with respect to the electric field and the derivative of the local magnetic moments with respect to the magnetic field will need to be calculated. The former will require accurate ab-initio evaluation of the $J_{i,j}$ integrals and the latter can be obtained using standard spin wave calculations (see [1, 10, 13]). In section 3 we will discuss the details of how these goals will be achieved.

## 3 Computing the magnetic exchange term from nuclei displacement

The magnetoelectric coupling is computed partially from the first derivative of the exchange integral $J$ as a function of an applied electric field $\mathcal{E}$, as seen in equation (4). This is the main difficulty of obtaining $\bar{\bar{\alpha}}$. The main effect of an electric field on an ionic insulator is to displace the ions (nuclei). This displacement can be computed from:

- The Hessian matrix of the energy $\mathcal{H} = \frac{\partial^2 E}{\partial \mathbf{d}^2}$ obtained from a mean-field density functional calculation (see section 4 for details)
- The Born dynamical effective charges $q^*$ which corresponds to the derivative of the energy with respect to the field and displacement: $Z_{ij} = -e\frac{\delta \mathcal{F}_\rangle}{\delta \mathcal{E}_|}$ [12]
- Newton's 2nd law

$$q^* \mathcal{E} = -\mathcal{H} d \tag{5}$$

which implies that the displacement induced by an electric field is equivalent to:

$$d = -\mathcal{H}^{-1} q^* \mathcal{E}$$

This equation also suggests that the main response of a ferroelectric compound to an applied electric field will be a geometry modification that can be evaluated by the displacement of the system ions from their equilibrium position.

We can evaluate the Hessian and the Born charges with density functional theory (DFT) as they both depend on the electron density functional. A functional is a function where the input is itself also a function. The electron density functional takes the electronic density function(s) as an input and gives system energy as an output. We can use this approximation of the system's properties because correlation interactions among the Fermi electrons will have only negligible effects on these quantities, and so it is safe to use DFT methods with the suitable functionals.

In summary our approach is as follows:

1. Optimise the structure's geometry to minimise the total energy of the system using DFT
2. From this, obtain the associated optimal Hessian matrix and Born tensor
3. Solve the resultant linear system of equations in equation (5) for charge displacement
4. For a range of displacements calculate the magnetic exchange integral $J$ using an embedded fragment, wavefunction-correlated method since DFT cannot generate accurate calculations for this part. This involves the consideration of many Slater's determinants: the evaluation of many antisymmetric superposition of states that each represent a given system configuration. This is required because the properties of J (governed by the Fermi level electrons) rely on a strongly-correlated, configuration-sensitive system. An example of this would be the SAS+S method (see section 4.4 and 4.5 for more details). This allows us to generate $\frac{\partial J}{\partial d}$
5. Computing displacement for a range of $\mathcal{E}$ allows for the generation of $\frac{\partial d}{\partial E}$ and indirectly $\frac{\partial J}{\partial d}$, and therefore the numerical evaluation of the derivative of the spin Hamiltonian parameters as a function of the applied electric field:

$$\frac{\partial J}{\partial \mathcal{E}} = \frac{\partial J}{\partial d}\frac{\partial d}{\partial \mathcal{E}}$$

6. Use non ab-initio spin wave methods to determine $\frac{\partial \langle S_{i,j} \rangle}{\partial H}$ and $\langle S_{i,j} \rangle$
7. Combine these terms to obtain $\bar{\bar{\alpha}}$

This method has successfully been applied to $YMnO_3$ using SAS+S to calculate the exchange integral for each value of the displacement, since it was designed specifically for the purpose of precise evaluation of magnetic excitations [19].

### 3.1 The Simulation Environment

The embedded fragments are designed to include the configuration states that describe the prioritised physical processes. It has been argued that in wide-gap magnetic insulators, $J$ is determined from strongly local electronic interactions involving two magnetic centres [2]. These fragments were embedded in an environment reproducing the main effects as the rest of the crystal. To achieve the calculation of J in the same way, we must include the Pauli exclusion effects of surrounding electrons and the influence of a non-local Madelung potential (nuclei-electron interactions simplified through ions being approximated as a set of point charges) within the fragment. The exclusion effect can be modelled through a surrounding total ionic pseudopotential (TIP, see [20] for an example of this method). The Madelung potential can be recreated through an array of renormalised point charges assigned to atomic positions [5]. In this way we effectively design the environment to match all the crystallographically independent phenomena in the compound.

## 4 Prerequisite Theories

### 4.1 Many-Body Schrödinger equation

Since DFT is a quantum mechanical modelling method it will be more comprehensive to explore some of the associated mathematical background. Starting from fundamental principles we will discuss the formula that is usually involved with describing quantised, microscopic systems - the Schrödinger equation.

Specifically, the nonrelativistic time-dependent Schrödinger equation for an individual particle is given by:

$$\left[-\frac{\hbar^2}{2m}\nabla^2+\mathcal{U}(\mathbf{r},t)\right]\Psi(\mathbf{r},t)=i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) \tag{6}$$

where $\mathcal{U}(\mathbf{r},t)$ is the external potential and $\Psi(\mathbf{r},t)$ is the wavefunction of the particle being described. This can be generalised to a system of $N$ electrons, the positions and mass of which are denoted by $r_i$ and $m_e$ respectively, and $M$ nuclei, whose positions and mass are given by $R_{I_i}$ and $m_{I_i}$ respectively. '$i,j$' are indexes of the ith or jth atom, as before. If we use atomic units ($\hbar = m_e = \frac{e^2}{4\pi\epsilon}$) we can expand the Schrödinger equation from equation (6) by adding the many-body terms:

$$\left[-\frac{1}{2}\left(\sum_i \nabla^2_{\mathbf{r}_i}+\sum_i\frac{\nabla^2_{\mathbf{R}_{I_i}}}{m_{I_i}}\right)+\frac{1}{2}\sum_{i\neq j}\frac{1}{|\mathbf{r}_i-\mathbf{r}_i|}-\sum_{i,j}\frac{Z_{I_j}}{|\mathbf{r}_i-\mathbf{R}_{I_j}|}\right.$$
$$\left.+\frac{1}{2}\sum_{i\neq j}\frac{Z_{I_i}Z_{I_j}}{|\mathbf{R}_{I_i}-\mathbf{R}_{I_j}|}\right]\Psi=i\frac{\partial\Psi}{\partial t}$$

where the term

$$\hat{T}=-\frac{1}{2}\left(\sum_i \nabla^2_{\mathbf{r}_i}+\sum_i\frac{\nabla^2_{\mathbf{R}_{I_i}}}{m_{I_i}}\right)$$

is the kinetic contribution and

$$\hat{V}=\frac{1}{2}\sum_{i\neq j}\frac{1}{|\mathbf{r}_i-\mathbf{r}_j|}-\sum_{i,j}\frac{Z_{I_i}}{|\mathbf{r}_i-\mathbf{R}_{I_j}|}+\frac{1}{2}\sum_{i\neq j}\frac{Z_{I_i}Z_{I_j}}{|\mathbf{R}_{I_i}-\mathbf{R}_{I_j}|}$$

is the potential contribution, taking into account interactions between electrons and other electrons, electrons and nuclei, and nuclei and other nuclei. $i\frac{\partial}{\partial t}$ is the energy operator, giving energy eigenvalues from the total wavefunction.

For most systems this equation cannot be solved exactly and so we must include firstly the Born Oppenheimer approximation. This uses the fact that an electron is almost 2000 times less massive than the nucleus, and so the positions of the nuclei can be treated as fixed when trying to generate the electron wavefunction, and therefore the source of the Coulomb interaction is always static. The electron wavefunction for the nth electron becomes separable, with an electronic part $\Psi_e$ and a nuclear part $\Psi_{nu}$ where $R_i$ are treated only as parameters:

$$\Psi(\mathbf{r}_i,\mathbf{R}_i)=\Psi_e(\mathbf{r}_i;\mathbf{R}_i)\Psi_{nu}(\mathbf{r}_i)$$

This allows us to generate two energy eigenvalue equations for each wavefunction and try to solve for the systemic wavefunction $\Psi(\mathbf{r_1},\mathbf{r_2},...\mathbf{r_N})$ [15]. The next section will discuss how to approach solving the Schrödinger equation in the time independent case.

## 4.2 Hartree-Fock Theory

The Hartree-Fock method is the basis of molecular orbital theory, which postulates that in a material, each electron's motion can be described by a single-particle function that does not depend explicitly on the neighbouring electron's instantaneous motion. These orbitals are only approximations of reality however; the only exact eigenfunctions of the full electronic Hamiltonian are for hydrogen or He$^+$ where the system contains only one electron. However, as long as we consider these molecules near their geometrical energetic equilibrium (see step one of section 3), Hartree-Fock theory can in many cases provide a good starting point for more detailed theoretical methods; for example the many-body perturbations, single-reference configuration interactions, seen in section 4.4.

It is designed to solve the electronic Schrödinger equation resulting from the reduced time independent equation after using the Born-Oppenheimer approximation, as in section 4.1.

For this we need a mathematical way of satisfying the fundamental physical conditions of a charged fermion inhabiting a mean-field environment. This is done by introducing the concept of a Slater determinant. Slater determinants satisfy the fermionic antisymmetry principle of a wavefunction while also superimposing all possible occupied states in a 'Hartree product'. It also satisfies the normalisation condition $\langle\Psi|\Psi\rangle = 1$, and since the corresponding Hamiltonian must commute with the spin operators, the wavefunctions must also be eigenfunctions of $\hat{S}_z$ and $\hat{S}^2$. However, except for a few simple cases, Slater determinants are not eigenfunctions of $S^2$ for N-electron systems. Despite some shortcomings, this approach allows for simpler formulas when constructing matrix elements for N-electron Hamiltonians or similar operators between different Slater determinants, and a more intuitive interpretation of the wavefunction.

For a two electron system we can write an antisymmetric spin orbital as

$$\Psi(\mathbf{x}_1)\Psi(\mathbf{x}_2) = \frac{1}{\sqrt{2}}[\chi_1(\mathbf{x_1})\chi_2(\mathbf{x_2}) - \chi_1(\mathbf{x}_2)\chi_2(\mathbf{x}_1)]$$

This can be written as a determinant:

$$\Psi(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{\sqrt{2}}\begin{vmatrix} \chi_1(\mathbf{x}_1) & \chi_2(\mathbf{x}_2) \\ \chi_1(\mathbf{x}_2) & \chi_2(\mathbf{x}_2) \end{vmatrix}$$

which can be generalised to:

$$\Psi = \frac{1}{\sqrt{N}}\begin{vmatrix} \chi_1(\mathbf{x}_1) & \chi_2(\mathbf{x}_1) & ... & \chi_N(\mathbf{x}_1) \\ \chi_1(\mathbf{x}_2) & \chi_2(\mathbf{x}_2) & ... & \chi_N(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \chi_1(\mathbf{x}_N) & \chi_2(\mathbf{x}_N) & ... & \chi_N(\mathbf{x}_N) \end{vmatrix} \tag{7}$$

If you try to put two electrons in the same orbital i.e. set $\chi_1 = \chi_2$ then the wavefunction becomes zero. This is one representation of Pauli's exclusion principle. If we have a list of the occupied orbitals we can shorten this form to Dirac notation, with the normalisation factor implied: $|ij...k\rangle$; see section 4.4. This way of writing electron wavefunctions assumes that electrons move independently from the others, and so Hartree-Fock theory is referred to as a 'mean field' theory.

Using this Dirac notation we can shorten the Hamiltonian to the following forms in order to remove some of the complexity in the evaluation of integrals. The single-electron operator can be written as:

$$h(i) = -\frac{1}{2}\nabla_i^2 - \sum_A \frac{Z_a}{r_{iA}}$$

and the two-electron operator can be written as:

$$v(i,j) = \frac{1}{r_{ij}}$$

so that the Hamiltonian becomes:

$$\hat{H} = \sum_i h(i) + \sum_{i<j} v(i,j) + V_{NN}$$

where $V_{NN}$ is a constant for the fixed set of nuclear coordinates, so we can ignore it for now as this term only shifts the eigenvalues, with no effect on the eigenfunctions. Now we need an approach to obtaining the molecular orbitals. If we are looking for a Hartree-Fock wavefunction in normalised Slater determinant form, we can start by asserting that the electronic energy is given by:

$$E_{electron} = \langle \Psi | \hat{H} | \Psi \rangle$$

For symmetric energy expressions we can use something called the 'variational theorem', which asserts that the calculated energy for the ground state is always an upper bound to the true minimum. We can choose a 'trial wavefunction' that depends on one or multiple parameters, and find the parameter configuration that minimises the expectation value of the energy in the observed functional space. This allows for a better approximation of the system wavefunction. This is a key method in the the Hartree-Fock approach. In other words, $E_{electron}$ will be an overestimation of the true ground state energy unless $\Psi$ is equal to the exact ground state wavefunction of the system, but a tighter upper bound can be generated from tuning the trial wavefunction [17]. The energy-minimising orbital configuration can be obtained as a linear combination of the basis functions [18].

We can rewrite the Hartree-Fock energy to achieve this purpose, by expressing it in terms of the integrals of one and two electron operators:

$$E_{electron} = \sum_i \langle i|h|i \rangle + \frac{1}{2} \sum_{ij} [ii|jj] - [ij|ji] \tag{8}$$

where

$$\sum_i \langle i|h|i \rangle = \int d\mathbf{x_1} \; \chi_i^*(\mathbf{x_1}) h(\mathbf{r_1}) \chi_j(\mathbf{x_1})$$

and

$$[ij|kl] = \int d\mathbf{x_1} d\mathbf{x_2} \; \chi_i^*(\mathbf{x_1}) \chi_j(\mathbf{x_1}) v(i,j) \chi_k^*(\mathbf{x_2}) \chi_l(\mathbf{x_2})$$

which can be calculated using common existing computer algorithms.

### 4.2.1 Hartree Fock Equations

We must minimise the value of $E_{electron}$ from equation (8) with respect to the changes in the orbitals $\chi_i$. So far the orbitals $\chi$ have been assumed to be orthonormal, and so we must keep them that way through our variational procedure. This is done by employing 'Lagrange's method of undetermined multipliers', where we use a Lagrangian $\mathcal{L}$ such that:

$$\mathcal{L}[\{\chi_i\}] = E_{electron}[\{\chi_i\}] - \sum_{ij} \epsilon_{ij} (< i|j > - \delta_{ij})$$

where the coefficients $\epsilon_{ij}$ are undetermined Lagrange multipliers and the $< i|j >$ orbital inner product represents the overlap between the ith and jth spin orbitals. Setting the first variation of the Lagrangian equal to zero ($\delta\mathcal{L} = 0$) and doing some algebra we can generate the Hartree-Fock equations defining each orbital:

$$h(\mathbf{x_1})\chi_i(\mathbf{x_1})+\sum_{j\neq i}\left(\int d\mathbf{x_2}\, |\chi_j(\mathbf{x_2})|^2 r_{12}^{-1}\right)\chi_i(\mathbf{x_1})-\sum_{j\neq i}\left(\int d\mathbf{x_2}\, \chi_j^*(\mathbf{x_2})\chi_i(\mathbf{x_2})r_{12}^{-1}\right)\chi_j(\mathbf{x_1}) = \epsilon_i\chi_i(\mathbf{x_1}) \tag{9}$$

where $\epsilon_i$ is the energy eigenvalue corresponding to the ith orbital eigenfunction. This problem can be solved in two ways, the first being numerically, and the second solving it in a space spanned by the basis functions. Both solutions depend on the orbitals, and so we need to guess some initial orbitals and generate increasingly educated guesses iteratively. The first term in equation (9) represents the Coulomb interaction of an electron described by $\chi_i$ with the average charge distribution of the other electrons (mean field). These concepts and systems of equations can be a good starting point for employing DFT in a practical case, where the correlations between electrons are not the highest priority. Meaning we can use an orthonormal orbital, mean field approach to modelling the ground state energy of the electronic structure of a multiferroic material.

## 4.3 Why and when to use DFT

In systems where the investigative focus is not on the correlation interactions of the Fermi-level electrons, density functional theory presents a convenient workaround for the computational complexity of solving the many-body Schrodinger equation analytically (see section 4.1). It can generate the total energy of the system by calculating a unique functional of the electron density. The exact dependence of the energy on this functional is still unknown, however attempts have been made to approximate these relations. Kohn and Sham transformed the problem of computing the ground state density of an N-electron system into a set of N linear Schrödinger equations corresponding to each electron [9].

Exchange correlation energy is a term in the Kohn Sham equations that can be interpreted as the contributions of detailed correlation and exchange to the system energy i.e. specific interactions between electrons in the electronic structure of the system. There are Pauli and Coulomb correlations as well as the correlation-kinetic effects are included in this term [9, 16]. The calculation of the exchange-correlation energy between atoms is treated using correlation functionals. DFT has been proved reliable in many cases, as long as the correlation energy is considerably less than the kinetic energy. This is not the case for strongly correlated systems i.e. when the exchange-correlation energy is dominant [14]. This is the property of Fermi-level electrons and their magnetic interactions. In transition metal oxides, they are localised practically on the d-orbitals of the metal ions. The Coulomb and exchange interaction energies in this case are of a larger magnitude than the electron's kinetic energy.

The domination of these interaction energies result in charge, spin or orbital occupation fluctuations that come from the competition between the different configurations in the electronic structure. The multi-configurational nature of the ground state system means that the ab-initio single-determinant method encounters accuracy issues. A different model must therefore be used to describe the effects that are dominated by these electrons [6]. However, for most cases, lattice geometry, dynamics and many other properties are still approximated effectively by DFT. In the next section we will discuss the cases where the property that we are investigating relies on a strongly correlated system, and the alternative techniques that are used to deal with this.

## 4.4 Accurately Employing Wavefunction Correlated Methods (WCMs)

Strongly correlated systems are characterised by strong Coulomb repulsions, which result in wavefunctions (for ground and low lying excited states) involving many Slaters determinants (SD's). With weakly correlated systems there can also be many SD's, but there is one main reference configuration with the largest weighting. A reference configuration is a convenient baseline where all subsequent configurations are referenced from (see below for diagrams of ground state reference configurations and their subsequent excitations). The non-reference SD's have a much smaller weighting in the configuration. In strongly correlated systems each SD possesses the same weighting and we must consider them all equally (there is not a dominant SD like in the former case).

When all of the orbital configurations are strongly coupled, a model that represents the electron distribution as a non-interacting, single-determinant density functional is not optimal if we want to accurately calculate effects that result from the Fermi level electron's correlation effects. This means we must rely on wavefunction correlated methods that are specifically designed for these mechanisms.

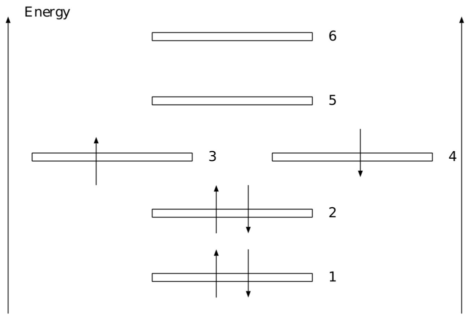
One such example is the difference-dedicated configuration interaction method (DDCI). This was born from the fact that many methods of determining the molecular properties of the ground state had difficulty treating excited states in an accurate way. Methods similar to the 'full configuration interaction' (FCI) approach (such as cluster methods, which include the effect of triple excitations), where all Slater determinants of the proper symmetry are included in the virtual procedure, have been extended to deal with excitation energy calculations. In other words, FCI-type methods determine the model Hamiltonian using exact diagonalization of selected configuration spaces formed by the SD's, which can be done on embedded fragments. Figures (1) and (2) are useful examples of configuration + excitement diagrams of a 6 electron system, with 4 electrons in the ground state, 2 active atoms and 2 virtual orbitals.

In general, extended FCI models are very effective at accurately approximating these energies, but some discrepancies appear. An example is the CC3 cluster method which generates excellent agreement with regular-model FCI results when the excited states are dominated by a single excitation, but displays considerable error when faced with a large double-excitation character. Another example is the multi-reference method CASPT2, which gives very good results for a large number of systems, but significant errors appear when the reference SD space and outer SD space are not sharply separated.

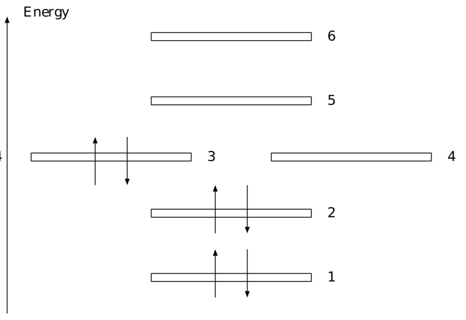
DDCI solves these problems by building a configuration interaction with a small reference space, in the subspace of singles and doubles which contributes to the energy difference on the grounds of second-order perturbation considerations, i.e. a subspace defined by the 'complete active space' (CAS) for all singles, and the doubles which involved at least one active orbital [4]. If faced with a system with numerous open-shells per atom (e.g high-spin manganese, cobalt, iron oxides), the computational cost of using CAS+DDCI or the LCAS+S ('large complete active space + single excitation' method) becomes prohibitive since the size of the space to diagonalise scales exponentially with the number of magnetic electrons. We can establish a simple physical criterion to select important reference configurations, and derive from it a novel method with a strongly reduced computational cost. This is because configurations involved LCAS+S for example, are far too numerous compared to the necessary methods for a good quality modelling of low-lying energy state physics. Something needs to change when approaching this type of problem.

Figure 1: All ground state configurations in the CAS for this system (values denoted as $x, \bar{x}$ represent a spin up and spin down electron occupying level $x$, respectively).

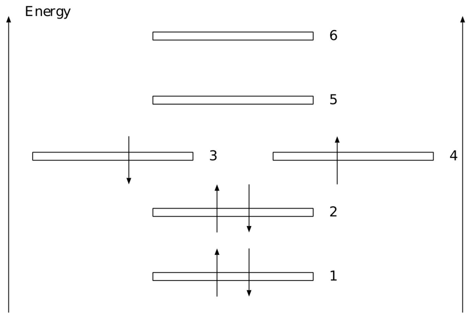
(a) Slater determinant $|1\bar{1}2\bar{2}3\bar{4}\rangle$



(b) Slater determinant $|1\bar{1}2\bar{2}3\bar{3}\rangle$



(c) Slater determinant $|1\bar{1}2\bar{2}\bar{3}4\rangle$



(d) Slater determinant $|1\bar{1}2\bar{2}4\bar{4}\rangle$

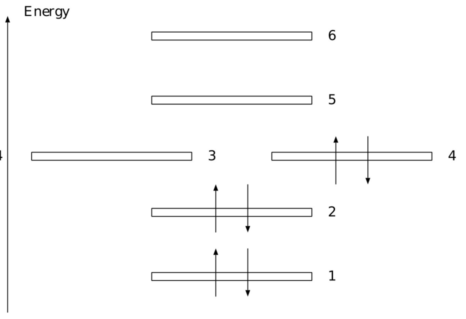
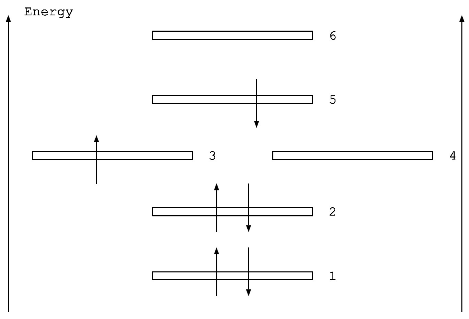
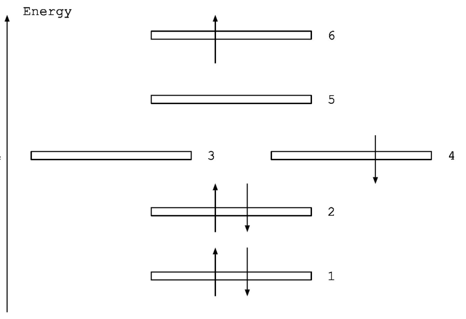


Figure 2: Ground state from figure (1) + single excitation examples (many more exist; number of configurations scale exponentionally with electron population N)

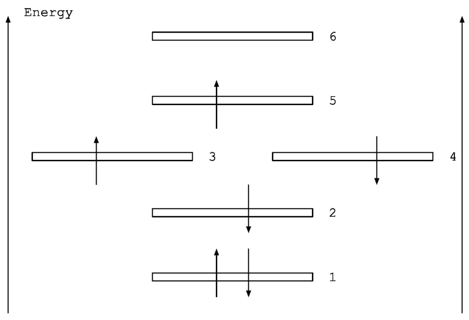
(a) Slater determinant $|1\bar{1}2\bar{2}3\bar{5}\rangle$



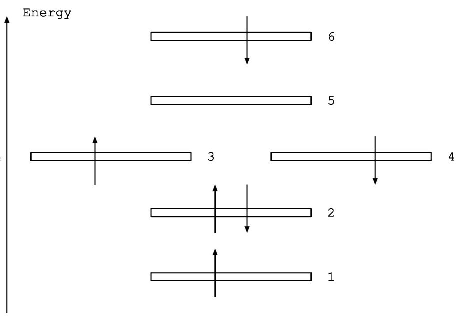
(b) Slater determinant $|1\bar{1}2\bar{2}\bar{4}6\rangle$



(c) Slater determinant $|1\bar{1}\bar{2}3\bar{4}5\rangle$



(d) Slater determinant $|12\bar{2}3\bar{4}\bar{6}\rangle$



### 4.5 Modifications of the Standard WCMs and Effective Hamiltonian Criteria

When solving this problem several modifications were made to the existing configuration methods by solid-state physicists. Compared to the LCAS+S, the multiple metal-to-metal and ligand-to-metal charge transfer configurations were removed from the reference part of the wavefunction (a ligand is an ion or molecule bound to a central metal atom; the charges being transferred are electrons between these bodies). When comparing to the CAS+DDCI, the multiple metal-to-metal charge transfer configurations and associated screening configurations were removed.

Additionally, double excitations were restricted to screening excitations on the ligand-to-metal charge transfers. This meant that the number of configurations in a magnetic interaction between two high-spin Mn III ions, for CAS+DDCI and LCAS+S, were reduced from approximately 10 billion and 60 billion, to 20 million, respectively. This less computationally expensive approach is called the 'selected active space' method (SAS+S); see [6] for more on this approach. This optimised, embedded fragment, quantum chemical ab initio method allows for analysis of the local fluctuations of magnetic couplings in metal compounds like manganites, which were known to be influential in the macroscopic magneto-resistance effects but not accessible experimentally. This approach is mentioned because it is utilized in this project's simulation chain; it is important to generate a configuration approach calibrated to studying a certain property. In this case, SAS+S has the best accuracy to calculation time ratio for studying the magnetoelectric coupling of multiferroic materials.

After the system geometry optimisation in CRYSTAL, Env takes the positions of a chosen quantum fragment in the ground state and prepares it for embedding. The results are fed through a program called Env2Seward (see section 6). Then the MOLCAS chain (another ab initio computational chemistry program) handles the SAS+S method, enabling us to provide an adequate description of the electronic correlation interactions and leads us closer to discovering the magnetoelectric properties of the system.

## 5 Calculation of the Displacement with CRYSTAL

CRYSTAL is a program initially written by V.R. Saunders, R. Dovesi, C. Roetti and others from the Theoretical Chemistry Group at the University of Torino as well as the Daresbury Laboratory in Cheshire, England. It performs ab initio calculations of the ground state energy and the energy gradient of a condensed system. It also has the potential to generate the electron wavefunctions and consequently their behaviour in a periodic system. Hartree-Fock and Kohn-Sham Hamiltonians are typically used, employing postulates of DFT, to achieve these outcomes. Periodic systems are approximated by expansion of the single-particle wavefunctions as a linear combination of Bloch functions [3].

A wave function that can be written as a plane wave modulated [1] by a periodic function is a 'Bloch wave', typically used to describe crystallographic electrons. Bloch functions are modulated by local functions or 'Atomic Orbitals', which are linear 'Gaussian type functions' used in the 'Linear Combination of Atomic Orbitals' (LCAO) method. The various molecular symmetries (s, p, d, f, sp e.t.c) of the structure are used to modulate these LCAO orbitals [3].

From a DFT calculation with CRYSTAL, after the optimal geometry is obtained with the execution keyword `OPTGEOM`, we can print the Hessian matrix to file with

[1] Modulation is the varying of the properties of a periodic waveforml; an example is frequency modulation (FM) and amplitude modulation (AM) in radio systems

the keyword `PRINTHESS`, which is processed by the script discussed in section 6. The Born charges are printed automatically, and with the lattice parameter and unit cell data, we can begin to approach obtaining the atomic displacements for a given system, incident upon which is a chosen uniform electric field.

# 6 Solving the displacement equation

## 6.1 Generating the atomic displacements

Once we have optimised the geometry of a system in CRYSTAL in order to minimise the total energy, i.e. finding the Hessian or minimum critical point of the $3N$ dimensional vector space, where $N$ is the number of atoms in the unit cell, we can generate the output files to be processed in order to solve the displacement of each atom for a given homogeneous electric field.

## 6.2 Details of Script Mechanisms

`disp_solve.py` is the second python driver script that I wrote at ILL. The script's first purpose is to parse the `.loto.out` or the `.DAT` output file from a CRYSTAL environment after the geometry of the system has been optimised and the consequential Hessian matrix and Born tensor have been calculated. The final output of the script is a grid of unit cell atomic coordinate files whose ions have been displaced by a range of uniform incident electric fields in various directions. The CRYSTAL output file(s) contains several pieces of information that are required for the next step towards this output.

Firstly, `disp_solve.py` parses for and stores the total number of atoms in the unit cell for a system '$N$'. This gives us the first dimension of the Born and Hessian matrices ($3 \times 3$ Cartesian-basis Hessian components $\mathcal{H}_{ij}$ or the diagonalised $3 \times 3$ Born charge elements $q_{ij}$ for N atoms adds up to a total dimension of $3N$) so we can allocate the correct amount of space to append the parsed values to. Then the script parses for the aforementioned $N$ sets of $3 \times 3$ Born charge matrices and arranges them into the required diagonalised format. Lets say $a_{ij}$ is the matrix element of the ith row and the jth column, then the element with $i = j = 3n + 1$, where $n \in \mathbb{Z}$ and bounded by $0 \leq n \leq N - 1$, is the centre for the nth Born charge matrix. Any Born tensor element separated by an index of more than one from any of these centres is zero.

Once the Born tensor has been generated, the lower triangular symmetric Hessian from the `.loto.out` file is parsed for and mapped so that $a_{ij} \equiv a_{ji}$ for $i < j$ and it becomes square and symmetric across the diagonal. If the Hessian data is not contained within this file, there is a contingency to scan and symmetrise the `HESSIAN.DAT` file. For both of these matrices there is a small machine error from the CRYSTAL output that needs to be removed, so for every value $\leq 10^{-12}$ it is reduced to zero to minimise uncertainty. Once we have both formatted matrices (the Hessian $\mathcal{H}$ and Born tensor $q$) we can solve the force equation for displacements $d$ caused by an incident uniform electric field $E$ on the crystal that is chosen beforehand (recall equation (5)):

$$q.E = -\mathcal{H}.d$$

using numerical python linear algebra modules (`numpy.linalg.solve`) that employs the Lapack routine `_gesvx` written in Fortran. This equation will take place uniformly in Hartree atomic units in Cartesian coordinates as this is the output format from CRYSTAL.

### 6.2.1 Conversion derivations

In order to achieve this format for equation (5), we must convert the user's electric field inputs to Hartree field units.The electric field is entered into the above equation as $E_1, E_2, E_3$ in the $x, y, z$ Cartesian basis, in units of $kV/m$, to be converted to atomic units in the same basis.

$$(Ex, Ey, Ez) = conversion * (E_1, E_2, E_3) \tag{10}$$

The electric atomic field conversion is given as:

$$\frac{E_h}{ea_0} = 5.142... \times 10^{11}(V/m)$$

where $E_h$ is the Hartree energy, $e$ is the unit electric charge and $a_0$ is the Bohr radius. To convert from $kV/m$ to these atomic field units, we must multiply our input values by 1000 to obtain the same base SI units ($kV/m$ to $V/m$) as the atomic field units so that the conversion value is dimensionless, and then divide by $E_h/ea_0$:

$$\text{Input units to a.u. conversion} = \frac{kV/m}{E_h/ea_0} = \frac{1000(V/m)}{5.142... \times 10^{11}(V/m)} = 1.94... \times 10^{-9}$$

If this conversion is included in equation (10), the correct units will be obtained for the electric field vector to be used in equation (5).

Once we have computed '$d$' for a given incident field in this format, the next step is to add the displacements to the unit cell file containing all of the relevant atomic coordinates. Before this is possible, it is necessary to convert to the displacements to fractional units in the crystallographic basis. This is the format of the cell files taken as input for Env15, and so our displacement basis and units must match. In order to make this conversion, we must parse the CRYSTAL `.loto.out` file for the basis conversion matrix ($x, y, z$ to $a, b, c$). Since this matrix is given in Angstrom units, the displacements must be converted from a.u. (Bohr) to Angstrom. Once the Bohr to Angstrom conversion has been applied, we can multiply our Cartesian displacements for each atom by this $3 \times 3$ conversion matrix, and then divide the results by the corresponding lattice vector. See below for an example:

Bohr to Angstrom conversion:

$$B2A = 0.529177210903$$

Parsed basis conversion matrix in Angstrom:

$$C = \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{pmatrix}$$

Cartesian coordinates for the nth atom:

$$d = \begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix}$$

Combine these with the lattice vectors $a, b, c$ to obtain the displacements in the new basis and units:

$$d' = \begin{pmatrix} a_n/a \\ b_n/b \\ c_n/c \end{pmatrix} = \begin{pmatrix} 1/a \\ 1/b \\ 1/c \end{pmatrix} \cdot \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{pmatrix} \cdot B2A \cdot \begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix}$$

Perform this action for all N sets of coordinates and the displacements can now be added to the unit cell.

### 6.3 The Cell File

The initial cell file is another output file of the CRYSTAL program that gives the position of the atoms in the unit cell in fractional coordinates alongside their corresponding charges. If all the required information is stored in this file, the `unit_source` parameter 'direct' can be called and all unit cell information is parsed from this file. Sometimes the cell file can be generated by CRYSTAL in a different spatial group, and so the number of coordinate sets in this file can be different to $N$. However, if `unit_source` = auto, `disp_solve` can scan this initial cell file for the charge magnitudes of each atom and reformulate the cell file by parsing the CRYSTAL output file for the unit cell coordinates and irreducible groups.

A new unit cell file is then written for the disturbed system containing double-precision floats with the associated converted displacements added. Entering `charge` instead of `auto` does the same thing but requires the user to enter the charge values for each irreducible atom group. We know which atom corresponds to the nth index and so we can merge the two sets of coordinates. This is then done on a large scale by specifying a range of electric fields in the $x, y, z$ directions. A grid of cell files specified by the $E_x, E_y, E_z$ inputs are generated and stored in a directory location reported by the command line, named after the specified ranges chosen for the incident field.

Another option to solve $q.E = -H.d$ would be directly employing the LAPACK Fortran drivers that specify to the type of matrix you are trying to process (in this case a symmetric semi-definite Hessian) which can save lots of processing time and generate increased accuracy. The trialled module was the symmetric system driver DSYEVX, that was ran in tandem with the python script using an executable batch file so that the displacement results could be compared. It was concluded that the numpy linear algebra framework was sufficient for the calculation purposes, and so the next step was to add calculated displacements to the atom coordinates from the CRYSTAL output files and process them with Env, before reformatting them with Env2Seward for different values of the electric field [2]. This will be automated so that we can effectively calculate $\frac{\partial d}{\partial E}$ [3] (recall step 5 of section 3).

## 7 Processing Env output into Seward

Env2Seward is the first driver script that I wrote my time at ILL; it was created to process the output files from the Env15 program, `prefix.env.sew0` and `prefix.env.psd`, that contain the fragment, TIP and Madelung potential coordinates and basis sets (see [19] as well as [5] and the associated local tools manual). It operates by parsing the input files for the various atom types and ordering the atom's coordinate data into sets categorised by each homogeneous atom type. It writes their associated basis sets from either a default path or chosen library file, chosen and input by the user.

Coordinates are compared between the `sew0` and `psd` files to confirm the identity and atom-type of the coordinate set. The script is quite general, allowing for 3 different formats of atomic identification in the Env output files, including contingencies for a symmetric basis in the `psd` file and a system exit if the coordinates do not match between the two input files. The user is prompted intuitively for

[2]Please see the appendix for an instruction manual that covers more detail how to use this software.

[3]See section 8 for more information on this process.

input of the basis set corresponding to each atom and the desired basis library, if no input file is passed on the command line. However, the script can be automated (if the user desired a conversion loop in a batch file), by calling it from command line with a separate input text file as a second argument, with the libraries and basis sets contained in in this input file. This command line input file can be generated automatically by the program through user input, or manually by the user. A third available format is the interactive Jupyter notebook that provides instructions in markdown alongside the python 3 cells.

Regardless of the input approach taken, the coordinates are organised into quantum fragment, ionic pseudopotential and Madelung potential categories in a way that is readable by the SEWARD input algorithm after being assigned to a certain atom type. See figure (3) for an excerpt of the fragment output.

```
\&SEWARD
Title
Excerpt

Expert
Verbose

*** Fragment ***************************************************
Basis set
Mn.ano-rcc.Roos.21s15p10d6f4g2h.6s4p3d1f0g.
spherical
Mn01       0.000000000000       0.000000000000       2.749761859316        Bohr
Mn02       0.000000000000       0.000000000000      -2.749761859316        Bohr
End of basis
****
Basis set
O.ano-rcc.Roos.14s9p4d3f2g.4s3p1d0f
spherical
O203       2.231731397024      -0.875526023191       0.000000000000        Bohr
O204      -2.231731397024       0.875526023191       0.000000000000        Bohr
O305       2.118901193775      -1.064763543958      -5.367483621542        Bohr
O306      -2.119040147228       1.064763543958      -5.367483621542        Bohr
O407       1.412183935493       3.326902091439      -2.642519536557        Bohr
O408       1.412183935493       3.326902091439       2.642519536557        Bohr
O409      -1.412183935493      -3.326902091439       2.642519536557        Bohr
O410      -1.412183935493      -3.326902091439      -2.642626886230        Bohr
O311       2.118901193775      -1.064763543958       5.367483621542        Bohr
O312      -2.119040147228       1.064763543958       5.367483621542        Bohr
End of basis
****
```

Figure 3: Excerpt of fragment coordinates from env2seward output file generated for SEWARD input of the GdMn2O5 system. No specified library so the default is used.

Again, the appendix contains an instruction manual that goes into more depth about specifically how to use this software.

## 8 Linking CRYSTAL output, Disp_Solve, Env15 and Env2Seward

As discussed before, the CRYSTAL optimised geometry output files contain the associated minimum energy Hessian and Born charge tensor data, alongside the unit cell atomic coordinates and lattice parameters. CRYSTAL also produces the unit cell coordinates alongside the atom's charges in a **cell file**. This data can be used to calculate the displacements from an induced uniform electric field using 'disp_solve', and a grid of new cell files are generated with electrically disturbed positions, based on the initial cell file passed as an input or the coordinates found in the `.loto.out` output file [4].

Another program that I wrote for this project was `crys2seward.py`. Crys2Seward edits the xenv15 (the executable Fortran file for Env) `envin` or `env.in` input files so that the env15 program can take cell files of different displacements from the disp_solve outputs. It does this by matching the number of atoms in the unit cell with the input cell file, changing the cell file taken as input from the initial cell file to each displaced cell file in the grid, and adding the displacements to the central magnetic atom coordinates. Xenv15 then produces a grid of output files named:

- `prefix.<field_values>.env.sew0`
- `prefix.<field_values>.env.psd`

These are the new input files passed to the Env2Seward program (the prefix corresponds to the simulated system). Crys2Seward modifies the Env2Seward input data so that the script processes every pair of `.psd` and `.sew0` files in the grid, so that each set of fragment coordinates can be categorised into a prefix.sew.in format, to be later input into SEWARD. Each file grid (cell files, env15 output files, sew.in files) is contained in a directory named after the electric field ranges and filetype.

The largest calculation time sink for this script is running xenv15 for each cell file; this gets exponentially more expensive as the number of atoms in the unit cell increases. Program performance was tested by running the program chain for several systems including YMnO3 and CuO, with Env2Seward and Disp_Solve being tested on systems such as GdMn2O5 and TbMn2O5.

[4]This is a summary of sections 5 and 6, see those for more details.

## Software Manual Appendix

## 9 Author and Permissions

This code was written by **Joshua Horswill** during an internship with the ILL theory group in June-November of 2020 under supervision of **Marie-Bernadette Lepetit** and **Elisa Rebolini**.

It is permissible to use and diffuse this code provided J. Horswill and MB Lepetit are kept informed. Modifications to this code can be made provided that the developer sends modifications to J. Horswill, M.B. Lepetit or E. Rebolini, so they may be included in the source code.

MB Lepetit
Institut Néel, CNRS UPR 2940
25 rue des Martyrs, BP 166, Bât. K
38042 Grenoble cedex 9
FRANCE

**Email :** Marie-Bernadette.Lepetit@Neel.CNRS.fr
E. Rebolini
Institut Laue-Langevin
71 Avenue des Martyrs,
38000 Grenoble,
France

**Email :** rebolini@ill.fr

J. Horswill
Institut Laue-Langevin
71 Avenue des Martyrs,
38000 Grenoble,
France

**Email :** horswill@ill.fr

## 10 Disp_solve

### 10.1 What is Disp_solve?

Disp_solve is a script that generates a number of atomic displacement cell files for a crystal compound. These displaced unit cell atoms result from enacting a range of uniform electric fields on a system. It takes input files from a CRYSTAL [5] simulation to generate the necessary matrices for this displacement calculation.

[5]Density functional quantum ab-initio program.

## 10.2 How to Use Disp_solve

### 10.2.1 Installing python 3 and necessary modules

A **python 3** compiler must be installed (see section 11.1.1 for information on how to install python 3 or an associated IDE [6]). If an error message occurs that mentions a missing module, execute the command

```
'pip install <module_name>'
```

in the terminal. The following python modules are required to compile the script:

- numpy
- sys
- itertools
- copy
- os.path
- tabulate
- ast
- pathlib
- json

Most if not all of these modules should be native to your python 3 package. You can check if you have these by executing the command:

```
'pip list | grep <module_name_you_want_to_check>'
```

or by running the script and interpreting the error for missing modules.

### 10.2.2 Required Files

The following files are required by the python script:

- (Compulsory) the CRYSTAL ouput file of a phonon calculation of your system. If this is your source for the Hessian and Born tensor, it must include IR intensities (Born charges) and Hessian matrix printed in the Cartesian basis.
- (Compulsory) the system.cell input file of the env code corresponding to the undistorted system.
- (Optional) the .DAT files output from the crystal calculation that contain the Fortran-complicit arrays for the Born tensor and Hessian matrix. These are used to generate these matrices when the CRYSTAL output file does not contain the necessary data.
- (Optional) the disp_solve command line input file. If this is absent, the script prompts will ask you for the input data.

[6] Integrated development environment

### 10.2.3 Executing the Script and Input File Format

Once you have made sure all of these files are present, you have two options:

- ‘`python3 disp_solve.py`’ (with no input file: the code will prompt you for the inputs).
- ‘`python3 disp_solve.py <input_file_path>`’ Passing input file as a command line argument.

The prompt option allows you to enter the information directly into the terminal. You will receive the following prompts:

- Unit cell generator parameter (direct,charge,auto).

  Enter `direct` if you want the unit cell irreducible atom information, coordinates and charge data to all come from initial cell file.

  Enter `charge` or `auto` if you want the coordinates and irreducible atom groups to come from crystal output file.

  `auto` automatically generates charge data from initial cell file, whereas `charge` allows for manual input.
- Relative or full path of Born tensor source.

  This can be the crystal output file, to be parsed for the Born charges, or the direct .DAT file. The source must be the same as the Hessian source.
- Relative or full path of Hessian matrix source.

  This can be the crystal output file, to be parsed for the HRED L matrix, or the direct .DAT file. The source must be the same as the Born source.
- Relative or full path of crystal output file:

  Input the name of the crystal output if it is local to the script directory, or the full path of the file if it is not.
- Relative or full path of initial cell file containing the unit cell and charge data:

  Same again, name if local, full path if non-local.
- start,stop,step separated by commas for $E_x$ ($E_y$, $E_z$ are done sequentially after) in units of kV/m.

  Simply input starting electric field value, final value and size of step as a tuple.
- If `charge` is chosen, float charge values will then be prompted

The alternative is writing an input text file that contains the above information in a certain format. Below is an example of an input file format:

```
born_file =
    ../ex_student_ymno3_data/Position/YMnO3/BORN_B1Pw_loto.DAT
hess_file = ../ex_student_ymno3_data/Position/YMnO3/HESSIEN.DAT
crystal_file =
    ../ex_student_ymno3_data/frequence.B1PW_PtBs.loto.out
cell_init = ../ymno3_loto.cell
ex = [0,0.5,0.1]
```

```
ey = [0,0.2,0.05]
ez = [0,1,0.2]
unit_source = direct
charge_dict = {"Y":3.0, "MN":3.0, "O1":-2.0, "O2":-2.0}
```

We have the following rules:

- Each line element is separated by a space i.e. there is a space either side of the '=' sign. Every variable must be on the same line as the associated piece of data.
- `crystal_file` is the variable assigned to the relative/full path directory of the crystal output file.
- `cell_init` is assigned to the relative/full path directory of the initial cell file.
- The `ex, ey, ez` lists must be in the format [start,stop,step] where start is the initial value, stop is the final value and step is the increment between values given in kV/m also.
- `unit_source` is the variable assigned to the unit cell generator parameter (charge,auto,direct).
- If `charge` is passed, a dictionary in python format must assign each atom to a charge value. The general format is {`"key":value`}. See section 2.3 for more examples.

Once this input file has been written, one can execute the script with this input file passed as an argument on the command line as above.

#### 10.2.4 Output files

- Output cell file is named after the initial cell file with an added (Ex, Ey, Ez) component label. If the files are called locally, output cell file grid will be generated and stored in a new folder inside the script directory that is named after the array parameters for the x, y, z components.
- If the inputs files are called from full path, the output cell files will be generated and stored in a new folder inside either the directory of the cell file or the output file (specified by the command line).

# 11 Env2Seward

## 11.1 Introduction

This code processes and formats the output from a program called ENV so it can be used in a calculation performed by a program called SEWARD.

## 11.2 Inputs Files and Data

There are two input files:

- Prefix.env.sew0 is a help file that contains the input data for SEWARD of the MOLCAS chain. It contains the data for the quantum fragment, the TIPS and the renormalised charges.
- The second is called prefix.env.psd, and contains the information on the atoms represented by TIPs.

The script requires the basis sets for two groups of atoms:

- The quantum fragment
- The ionic pseudopotential (TIPs)

Once these have been prompted and entered through the terminal command line, an output file ready to be fed into SEWARD will be generated. This will be called prefix.sew.in when using the prompt option, however, the filename is customisable if you use the input method in 11.4.2.

For command prompt data input, the files output from Env to be parsed by Env2Seward must be in the same directory as the python script. If you want the files to be called from a different relative or full path directory, it is recommended to employ the input method below.

## 11.3 Installing Python

- A **python 3** distribution such as 'Anaconda' or 'Active Python' can be downloaded, which provides the option of using integrated development environments (IDE) such as 'Spyder' for programming in the python language
- If you are using an IDE, open the file (typically an 'open file' option in the top left of the window, or you can drag it into the IDE text editor), hit run, and answer the prompts in the terminal
- If you want to install python directly, see `https://realpython.com/installing-python/`

## 11.4 Input Options

1. '`python3 env2seward.py`' (no input file: command line will prompt you for direct inputs; can use full or relative path for script). If you want to call the script from full or relative path, just replace the `env2seward.py` with it's full or relative path directory. The new file will appear in the same directory as the python script.
2. '`python3 env2seward.py ⟨input_file_path⟩`' (Input file is passed as a command line argument. Can be relative or full path.)

### 11.4.1 Command Prompt Input

For the first option you will be prompted for:

1. Name of the system prefix such as 'GdMn2O5_J1' (Coordinate input files must be local for this option). Must match the prefix of the Env output files.
2. Title and name of created file
3. Basis sets and library locations

Continue until all basis sets have been entered and the file will be written.

### 11.4.2 Reading Filenames, Basis Sets and Libraries from an Input File

For the second option you will need to write the manually-generated input file with the following format:

```
filename = chosen_filename
title = chosen_title
# Following files can be local path or full path, just replace
    name with full path directory
sew0_file = prefix.env.sew0
psd_file = prefix.env.psd
lib_frag = {"atom type":{"loc":"specified basis set
    library","key":"basis set for
atom type"},"atom type2":...}
lib_pseudo = {"atom type":{"loc":"specified basis set
    library","key":"basis set
for atom type"},"atom type2":...}
```

- Each line element is separated by a space
- Write the filenames and paths to the input files next to the corresponding variable (must be named sew0_file, psd_file e.t.c)
- Give the fragment library and pseudopotential library in the python dictionary format: `https://www.w3schools.com/python/python_dictionaries.asp`.

Here is an example for a GdMn2O5 input file, where the fragment dictionary (`lib_frag`) has no specified library but the TIP dictionary (`lib_pseudo`) basis sets are found in PSEUDO:

```
filename = example.sew.in
title = example
sew0_file = GdMn2O5_J1.env.sew0
psd_file = GdMn2O5_J1.env.psd
lib_frag =
    {'Mn':{'loc':'','key':'Mn.ano-rcc.Roos.21s15p10d6f4g2h.6s4p3d1f0g.'},
'O':{'loc':'','key':'O.ano-rcc.Roos.14s9p4d3f2g.4s3p1d0f'}}
lib_pseudo =
    {'Gd1':{'loc':'PSEUDO','key':'Gd.ECP.Marie.0s.0s.0e-Gd1-GdMn2O5.'},
'Gd2':{'loc':'PSEUDO','key':'Gd.ECP.Marie.0s.0s.0e-Gd2-GdMn2O5.'},
'Mn1':{'loc':'PSEUDO','key':'Mn.ECP.Marie.0s.0s.0e-Mn1-GdMn2O5.'},
'Mn2':{'loc':'PSEUDO','key':'Mn.ECP.Marie.0s.0s.0e-Mn2-GdMn2O5.'},
'O1':{'loc':'PSEUDO','key':'O.ECP.Marie.0s.0s.0e-O1-GdMn2O5.'},
'O2':{'loc':'PSEUDO','key':'O.ECP.Marie.0s.0s.0e-O2-GdMn2O5.'},
'O3':{'loc':'PSEUDO','key':'O.ECP.Marie.0s.0s.0e-O3-GdMn2O5.'},
'O4':{'loc':'PSEUDO','key':'O.ECP.Marie.0s.0s.0e-O4-GdMn2O5.'}}
```

If the setup has been performed correctly, the script will output 'File has been created' and the sew.in file will appear in the input file directory.

### 11.4.3 Jupyter Notebook Version

To install Jupyter Notebook, follow this guide: `https://jupyter.org/install`. For the creation of the `env2seward_notebook.ipynb` file, the Anaconda distribution was used to install Jupyter Notebook with the `conda` command. This is the recommended method (see `https://www.anaconda.com/products/individual` for distribution download).
Once you have installed Jupyter notebook you need to download and localise the following files into the same directory/folder:

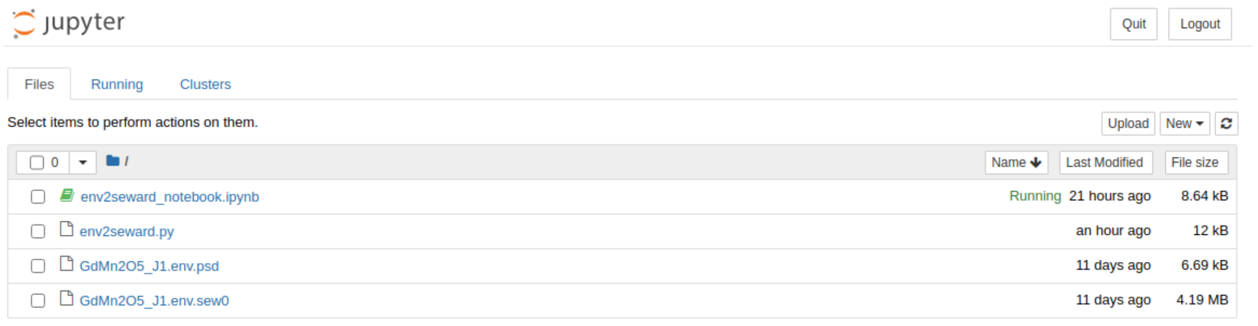
- `env2seward.py`
- `env2seward_notebook.ipynb`
- `prefix.env.sew0`
- `prefix.env.psd`

Next we need to open the Jupyter Notebook web app. We can do this by:

- Opening up a terminal in the directory containing the four files mentioned above.
- Entering '`jupyter notebook`' to initialise the web app.
- Your browser should open it in a new tab and you should see these four files in your notebook directory.
- Now open the `env2seward_notebook.ipynb` file from the web app dashboard menu.

You should see the menu shown in figure (1) and the notebook shown in figure (2). In the notebook you will see a page of Jupyter cells, some in markdown (a text formatting language) and some to run python3 code. If you want to learn more about how these work, see `https://www.dataquest.io/blog/jupyter-notebook-tutorial/`. In order to run these cells individually, click on them and hit `Ctrl+Enter` or press the run button (below the cell tab in the toolbar).

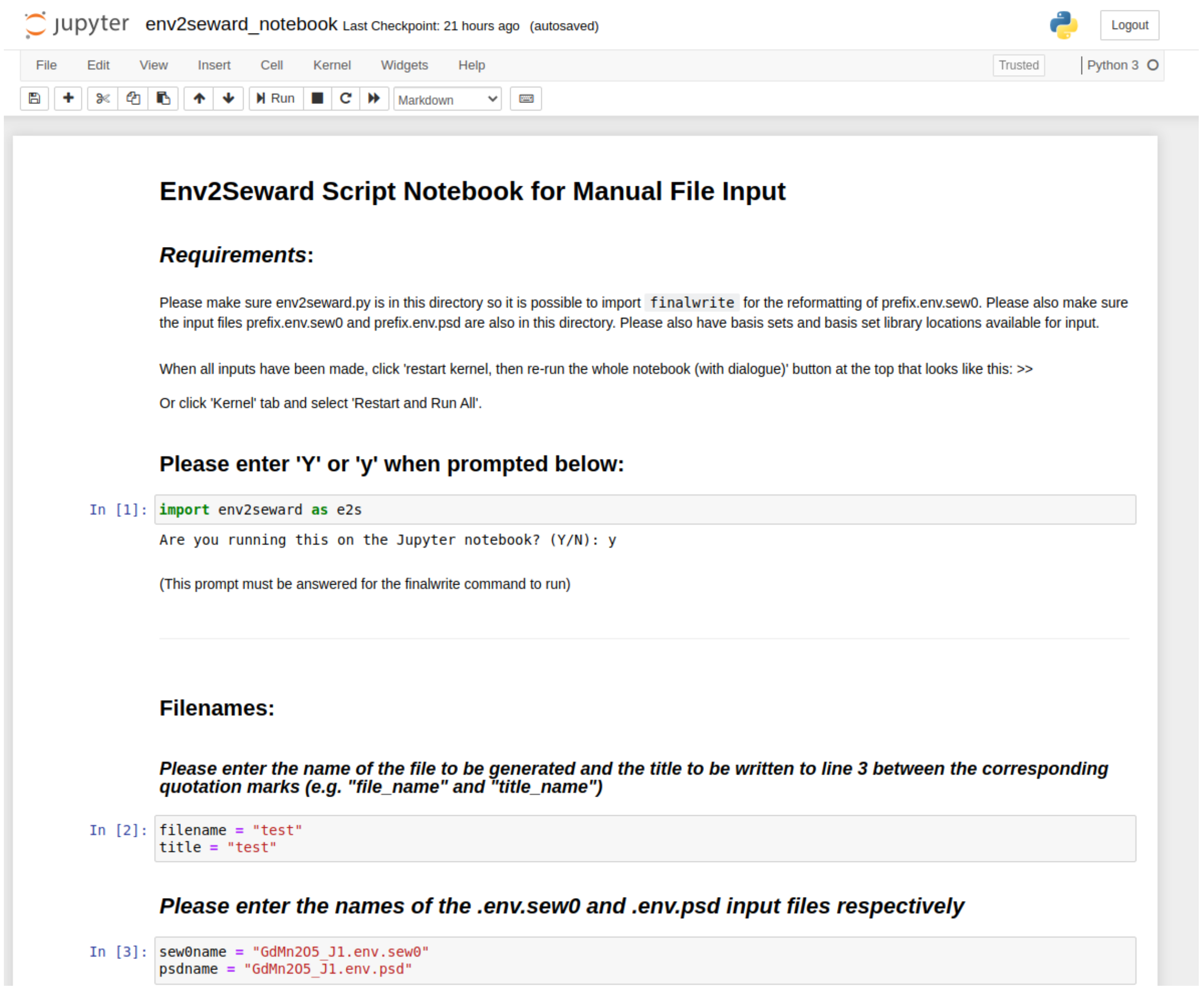
Please follow the markdown instructions in between the code cells to enter the correct inputs. If you would like to see the atom types that require basis set entry, follow the instructions under the subtitle 'Basis atoms (see instructions below on how to run this cell):'. The next steps are:



Select items to perform actions on them.

```
env2seward_notebook.ipynb   Running  21 hours ago   8.64 kB
env2seward.py                        an hour ago    12 kB
GdMn2O5_J1.env.psd                   11 days ago    6.69 kB
GdMn2O5_J1.env.sew0                  11 days ago    4.19 MB
```

Figure 4: Web app dashboard menu



**Env2Seward Script Notebook for Manual File Input**

***Requirements*:**

Please make sure env2seward.py is in this directory so it is possible to import `finalwrite` for the reformatting of prefix.env.sew0. Please also make sure the input files prefix.env.sew0 and prefix.env.psd are also in this directory. Please also have basis sets and basis set library locations available for input.

When all inputs have been made, click 'restart kernel, then re-run the whole notebook (with dialogue)' button at the top that looks like this: >>

Or click 'Kernel' tab and select 'Restart and Run All'.

**Please enter 'Y' or 'y' when prompted below:**

```
In [1]: import env2seward as e2s
        Are you running this on the Jupyter notebook? (Y/N): y
```

(This prompt must be answered for the finalwrite command to run)

**Filenames:**

***Please enter the name of the file to be generated and the title to be written to line 3 between the corresponding quotation marks (e.g. "file_name" and "title_name")***

```
In [2]: filename = "test"
        title = "test"
```

***Please enter the names of the .env.sew0 and .env.psd input files respectively***

```
In [3]: sew0name = "GdMn205_J1.env.sew0"
        psdname = "GdMn205_J1.env.psd"
```

Figure 5: env2seward_notebook when opened

- Make sure the inputs have all been entered according to the instructions (filename, title, sew0name, psdname, lib_frag, lib_pseudo)
- Select the button that says 'restart the kernel and re-run the whole notebook' that looks like a fast-forward icon.
- If you cannot find this, click the kernel icon (in English, it may be different in other languages), next to cell and widgets, and select 'Restart and Run All'.
- You may see a prompt in the first code cell (`import env2seward as e2s`). Please enter 'Y' if so.
- After this the last code cell (`e2s.finalwrite(filename,...)` will output 'File has been created' which means you now have a seward input file ready in the webapp directory. You can download this to a specific directory by going back to the dashboard menu, ticking the box next to '`chosen filename`' and clicking 'Download'.
- The output file will also naturally appear in the same directory as your `env2seward_notebook.ipynb` file.

## 12 Crys2Seward

Crys2seward is a python program that links the output from the CRYSTAL geometry optimisation to `disp_solve`, `xenv15` and `env2seward` to produce a grid of input files for the program SEWARD.

### 12.1 Installing python 3

A **python 3** distribution must be installed. Either Anaconda or direct installation `https://realpython.com/installing-python/` are arguably the best options.

### 12.2 Correct packages

The following packages must be installed with your distribution for the program to run:

- sys
- os
- shutil
- pathlib
- env2seward
- disp_solve

Install packages you don't have with `pip install <module_name>` or check that you have it with `pip list | grep <module_name_you_want_to_check>`, unless they are packages available only in this repository such as env2seward and disp_solve.

In that case you will need to download both the script and the `__init__.py` files from the git repository, and reference their path in `crys2seward.py` when declaring used packages (unless they are located in the same relative path as the repository, as it is by default, then no changes are required).

## 12.3 Input files

The program requires the following inputs:

- CRYSTAL output file of a phonon calculation of your system including IR intensities (Born charges) and Hessian matrix printed in fractional coordinates
- The initial system.cell file of the ENV code corresponding to the undistorted system
- The envin file for the env15 program `xenv15` after executing the necessary `make` command in the Env15 folder (see the local tools manual written by M. B. Lepetit for how to install Env15).
- The prefix.c2s.in input file containing all the paths pointing to the aforementioned files, as well as the inputs for `disp_solve` and `env2seward`
- (Optional) The .DAT files for the Born and Hessian matrices, used if the CRYSTAL output .loto.out file does not contain the necessary data to construct one or both of these matrices.

## 12.4 Executing the script

To execute the script, type:

```
python3 <path_to_crys2seward.py> <path_to_c2s.in_input_file>
```

into the command line in the same working directory as the xenv15 program and it's input files (envin and env.out). `xenv15` only works in the current working directory, i.e. it cannot be referenced from another path.

For example, on this git repository the relative paths are:

```
python3 ../../../python_scripts/crys2seward/crys2seward.py ../ymno3_d1.c2s.in
```

The main difficulty of executing the script is writing a correct input file. See figure (6) for an example.

### 12.4.1 Rules for c2s.in:

- envin and envout correspond to the absolute path of the prefix.envin, prefix.env.out files

```
#########crystal2seward input########################
envin = ymno3_d1.envin
envout = ymno3_d1.env.out
xenv_sew0 = ymno3_d1.sew0
xenv_psd = ymno3_d1.psd
#########disp_solve input############################
born_file = ../ex_student_ymno3_data/Position/YMnO3/BORN_B1Pw_loto.DAT
hess_file = ../ex_student_ymno3_data/Position/YMnO3/HESSIEN.DAT
crystal_file = ../ex_student_ymno3_data/frequence.B1PW.loto.out
cell_init = ../ymno3.cell
ex = [0,.15,.03]
ey = [0,.15,.03]
ez = [0,.15,.03]
unit_source = auto
charge_dict = {'Y':3.0, 'MN':3.0, 'O1':-2.0, 'O2':-2.0}
#########env2seward input###########################
filename = ymno3_d1.sew.in
title = <insert_title_here>
sew0_file = ymno3_d1.0.0_0.0_0.0.env.sew0
psd_file = ymno3_d1.0.0_0.0_0.0.env.psd
lib_frag = {'O': {'key': 'O_frag_basis', 'loc': ''},
'Y': {'key': 'Y_frag_basis', 'loc': ''}, 'MN': {'key': 'MN_frag_basis', 'loc': ''}}
lib_pseudo = {'MN': {'key': 'MN_pseudo_basis', 'loc': 'MN_pseudo_library'},
'O1': {'key': 'O1_pseudo_basis', 'loc': 'O1_pseudo_library'},
'O2': {'key': 'O2_pseudo_basis', 'loc': 'O2_pseudo_library'},
'O3': {'key': 'O3_pseudo_basis', 'loc': 'O3_pseudo_library'},
'O4': {'key': 'O4_pseudo_basis', 'loc': 'O4_pseudo_library'},
'Y': {'key': 'Y_pseudo_basis', 'loc': 'Y_pseudo_library'},
'Y1': {'key': 'Y1_pseudo_basis', 'loc': 'Y1_pseudo_library'},
'Y2': {'key': 'Y2_pseudo_basis', 'loc': 'Y2_pseudo_library'}}
```

Figure 6

- xenv_sew0 and xenv_psd correspond to the default names of the xenv15 output files for the system
- The disp_solve and env2seward input rules can be found in their README's and their sections in this software manual.

## 12.5 Output

Each grid of files is stored in an appropriately named folder. The sew0, psd and sew.in files are generated in separate folders but in the same current working directory with the xenv15 files as discussed before. The disp_solve cell files are generated in the same way, but stored in the same directory as the initial cell file or the CRYSTAL output file if their relative or full paths are specified. They are **not** generated in the local directory unless the disp_solve input file paths are not specified. Every grid folder name features the start, stop, step for Ex, Ey, Ez chosen in the c2s.in file in the disp_solve input section.

# References

[1] PW Anderson. Limits on the energy of the antiferromagnetic ground state. *Physical Review*, 83(6):1260, 1951.

[2] Ibério de PR Moreira, Francesc Illas, Carmen J Calzado, Javier F Sanz, Jean-Paul Malrieu, Nadia Ben Amor, and Daniel Maynau. Local character of magnetic coupling in ionic solids. *Physical Review B*, 59(10):R6593, 1999.

[3] Roberto Dovesi, VR Saunders, C Roetti, R Orlando, CM Zicovich-Wilson, F Pascale, B Civalleri, K Doll, NM Harrison, IJ Bush, et al. Crystal17. 2017.

[4] VM García, M Reguero, and R Caballol. Application of the iterative difference-dedicated configuration interaction method to the determination of excitation energies in some benchmark systems: Be, ch+, bh and ch2. *Theoretical Chemistry Accounts*, 98(1):50–56, 1997.

[5] Alain Gellé and Marie-Bernadette Lepetit. Fast calculation of the electrostatic potential in ionic crystals by direct summation method. *The Journal of chemical physics*, 128(24):244716, 2008.

[6] Alain Gellé, Julien Varignon, and M-B Lepetit. Accurate evaluation of magnetic coupling between atoms with numerous open shells: An ab initio method. *EPL (Europhysics Letters)*, 88(3):37003, 2009.

[7] T Goto, T Kimura, G Lawes, AP Ramirez, and Y Tokura. Ferroelectricity and giant magnetocapacitance in perovskite rare-earth manganites. *Physical review letters*, 92(25):257201, 2004.

[8] N. Hur, S. Park, P. A. Sharma, J. S. Ahn, S. Guha, and S.-W. Cheong. Electric polarization reversal and memory in a multiferroic material induced by magnetic fields. *Nature*, 429(6990):392–395, May 2004. `doi:10.1038/nature02572`.

[9] Walter Kohn and Lu Jeu Sham. Self-consistent equations including exchange and correlation effects. *Physical review*, 140(4A):A1133, 1965.

[10] Ryogo Kubo. The spin-wave theory of antiferromagnetics. *Physical Review*, 87(4):568, 1952.

[11] Lev Landau. The theory of phase transitions. *Nature*, 138(3498):840–841, 1936.

[12] Marie-Bernadette Lepetit. How to compute the magneto-electric tensor from ab-initio calculations? *Theoretical Chemistry Accounts*, 135(4):91, 2016.

[13] Takehiko Oguchi. Theory of spin-wave interactions in ferro-and antiferromagnetism. *Physical Review*, 117(1):117, 1960.

[14] Warren E Pickett. Electronic structure of the high-temperature oxide superconductors. *Reviews of Modern Physics*, 61(2):433, 1989.

[15] Elisa Rebolini. *Range-separated density-functional theory for molecular excitation energies*. PhD thesis, 2014.

[16] Keith Refson, Paul R Tulip, and Stewart J Clark. Variational density-functional perturbation theory for dielectrics and lattice dynamics. *Physical Review B*, 73(15):155114, 2006.

[17] JJ Sakurai. Modern quantum mechanics (revised edition), tuan san fu. *Reading: Addison-Wesley*, 1994.

[18] C David Sherrill. An introduction to hartree-fock molecular orbital theory. *School of Chemistry and Biochemistry Georgia Institute of Technology*, 2000.

[19] J Varignon, S Petit, A Gellé, and MB Lepetit. An ab initio study of magneto-electric coupling of ymno3. *Journal of Physics: Condensed Matter*, 25(49):496004, 2013.

[20] NW Winter, RM Pitzer, and DK Temple. Theoretical study of a cu+ ion impurity in a naf host. *The Journal of Chemical Physics*, 86(6):3549–3556, 1987.