Supporting information for: Effects of Concentration, Crystal Structure, Magnetism, and Electronic Structure Method on First-Principles Oxygen Vacancy Formation Energy Trends in Perovskites

Matthew T. Curnan[†,‡] and John R. Kitchin[*,†,¶]

National Energy Technology Laboratory-Regional University Alliance (NETL-RUA), Pittsburgh, Pennsylvania 15236

E-mail: jkitchin@andrew.cmu.edu

[*] To whom correspondence should be addressed

[†] National Energy Technology Laboratory-Regional University Alliance (NETL-RUA), Pittsburgh, Pennsylvania 15236

[‡] Department of Materials Science and Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213

[¶] Department of Chemical Engineering, Carnegie Mellon University, 5000 Forbes, Ave, Pittsburgh, PA 15213

February 3, 2016

Contents

1 Introduction

This supporting information document for the article, ”Effects of Concentration, Crystal Structure, Magnetism, and Electronic Structure Method on First-Principles Oxygen Vacancy Formation Energy Trends in Perovskites”, provides all of the information needed to reproduce all of the results presented in this article, in addition to explaining several of the preliminary tests and analyses performed to justify the insights and assumptions made in the article itself. Both this supporting information section and the article explained by it were prepared using the org-mode mark-up language, which facilitates the formatting, drafting, organization, and quick PDF conversion of manuscripts prepared in LaTeX.[S1] The original source of this document can be found here: 📌.

Each section of this supporting information document provides additional information regarding how the results contained within a particular division of the article were achieved, illustrating how the calculations yielding the results were completed, detailing how results were aggregated and visualized as plotted data, and justifying several of the implicit assumptions made in using particular results to verify drawn conclusions. Minimally, each section of this document contains an INCAR, KPOINTS, and POSCAR file to allow the replication of each different type of calculation performed in each subsection of the Results and Discussion section of the article, which was accomplished using version 5.2.12 of VASP and the PAW-PBE pseudopotential set in this article. In addition, each section will also contain the Python[S2] script applied to generate each plot within each section of the article, accompanied by the numerical values of the oxygen vacancy formation energies depicted on each plot.

Furthermore, several subsections of this article directly or indirectly allude to a Supporting Information document to substantiate generalizations, elucidate results, or illustrate methodology more clearly, all of which are addressed in the subsections listed below. For example, the Birch-Murnaghan equation of state is used to find the ground state energies that ultimately enable the calculation of oxygen vacancy formation energies in this article, though the standalone code needed to calculate these ground state energies and plot them is included only in certain subsections of this document. In certain cases that require comparisons of total DFT energy – such as section 3.5.4 – or assessing changes in equilibrium cell volume, such as section 3.7.5, several raw numerical results or plots produced by the equation of state are included to further explain various aspects of methodology that could not be adequately covered in the article itself. Several of these sections will also feature additional input files needed to generate necessary supporting information.

All of the results of calculations and input parameters responsible for these calculations are stored in a database consisting of relations that are largely consistent (excluding one set of relations) with either just Boyce-Codd Normal Form (BCNF)[S3] or both BCNF and 4th Normal Form (4NF).[S4] The information needed to generate and sort all input files and results of calculations will be stored in several files of the .csv file format, which can be imported into a wide variety of relational database management systems (RDBMS) such as MySQL[S5] or SQLite[S6] to appropriately process information. In the following section, the relational schema categorizing the data will be detailed, illustrating how the information can be recalled. Pursuant to the recollection of information needed for plot generation, the beginning of each subsection of this document, which each correspond to a subsection of the Results and Discussion section of the article, will contain relational algebraic expressions and matching SQL queries suitable for generating all of the plots contained in that Results and Discussion subsection from the database itself.

2 Relational Schema for Database

The relational hierarchy of the database detailed below can be illustrated using the class diagram generation feature of the PlantUML Java tool,[S7] representing each relation (R) as an object in the diagram, each attribute as an element listed within an object or relation, and each key as a morphism or link between two objects or relations.

In the following sections, we describe each feature of the structure illustrated above in detail. Assembly of the relational schema dictating the structure of the database presented here proceeds from three initial relations or tables, which are further decomposed into subsections to organize data in an efficient manner inspired by the design principles of BCNF and 4NF while providing all of the necessary energetic and volumetric information needed to reproduce and verify presented results. One of these initial relations refers to the raw output data of completed calculations. Relations connecting raw data to calculated energies and volumes portrayed in the article or in this document have been preemptively decomposed into stoichiometric (NoVac) and defect-containing (1Vac) tables. Therefore, each RawData

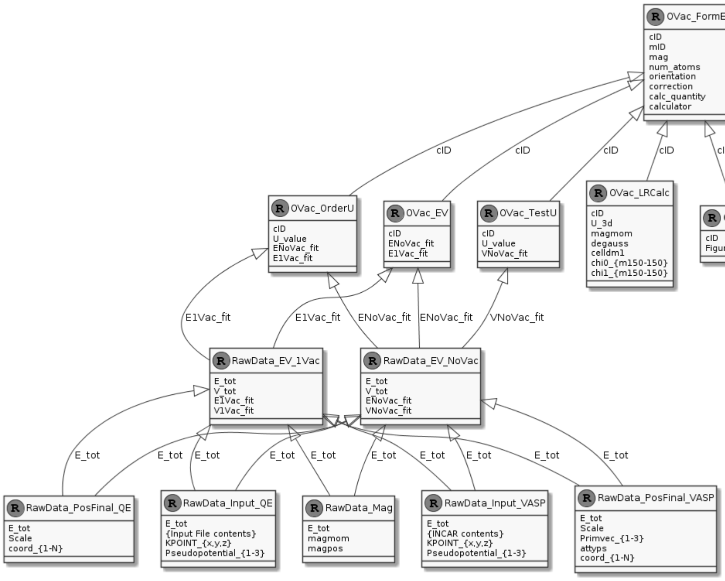

Figure S1: Visualization of the relational hierarchy of the database after division into the subrelations detailed above.

relation (NoVac and 1Vac) was initially organized as follows:

2.1 RawData(E_tot, V_tot, E_fit, V_fit, $magmom$, $magpos$)

The attributes defined in each RawData relation are defined as:

2.1.1 *E_tot*

For a given input volume, this is the outputted total energy of a DFT calculation. VASP calculated energies are calculated in units of eV, whereas Quantum Espresso (QE) calculated energies are calculated in units of Ry.

2.1.2 *V_tot*

The inputted volume corresponding to *E_tot*. VASP calculated volumes are calculated in units of Å^3, whereas QE calculated volumes are calculated in units of a.u.^{3} (Bohr^3).

2.1.3 *E_fit*

For a set of *E_tot* values, a particular value of energy labelled E_fit is calculated by Birch-Murnaghan equation of state (EOS) fit. A single tuple value of E_fit corresponds to range of DFT energies or *E_tot* values. VASP calculated energies are calculated in units of eV, whereas QE calculated energies are calculated in units of Ry.

2.1.4 *V_fit*

For a set of *V_tot* values, a particular value of volume labelled V_fit is calculated by Birch-Murnaghan equation of state (EOS) fit. A single tuple value of V_fit corresponds to range of DFT volumes or *V_tot* values. VASP calculated volumes are calculated in units of Å^3, whereas QE calculated volumes are calculated in units of a.u.^{3} (Bohr^3).

2.1.5 magmom

In a VASP magnetic calculation, this term is the magnetic moment magnitude resolved on an individual magnetic atom after the conclusion of a calculation. In QE, this refers to the total magnetization resolved upon convergence of a calculation.

2.1.6 magpos

The position of a magnetic atom in a magnetic calculation after the calculation has ended, described in reduced coordinates (i.e.: x, y, or z coordinates are in the domain [0,1]) separated by semicolons.

2.1.7 Decomposition and Generation of RawData

In this relation, the combined attributes of magmom and magpos would serve as keys for E_tot in the absence of other attributes, as all magnetic moment and position combinations corresponding to a single calculation match to a single total energy in both VASP and QE calculations. Thus, E_tot observes a functional dependency to these magmom and magpos combinations and the following decompositions are made in accordance with Boyce-Codd Normal Form (BCNF), rendering the subrelation RawData_Mag. Data is stored in RawData_Mag.csv :

$$RawData\_Mag(magmom, magpos) \rightarrow RawData\_Mag(E\_tot) \tag{1}$$

There are no further functional dependencies in the following table, thus those decompositions allow the table to be compatible with BCNF. However, a multivalued dependency exists, as V_tot, E_fit, and V_fit can fully determine any value of E_tot in all VASP calculations and all QE calculations not involving linear response. Thus, further decomposition of the table will occur in accordance with 4th Normal Form (4NF), rendering the subrelations RawData_EV_NoVac and RawData_EV_1Vac. Data is stored in RawData_EV_NoVac.csv and RawData_EV_1Vac.csv , respectively:

$$RawData\_EV(E\_tot) \Rightarrow RawData\_EV(V\_tot, E\_fit, V\_fit) \tag{2}$$

In the case of linear response QE calculations, full determination of a quantity of interest requires different information and will be shown later. Information concerning the structures inputted into calculations is entered into the Materials initial relation, which is detailed below.

2.2 Materials(sym, struct, f_unit_cell, morph, sID, mID, {at1, at2, . . . , atN})

The attributes defined in each Materials relation are defined as:

2.2.1 sym

The sym attribute refers to the symmetries observed by a studied system. Although this attribute can be used to classify molecules by their molecular geometry, it is primarily used to classify periodic structures with the crystallographic point group symmetries they observe.

2.2.2 struct

The struct attribute can be used to differentiate molecules and aperiodic materials from periodic structures. However, in the context of periodic, crystallographic structures, this attribute refers to the Bravais lattice of a system.

2.2.3 f_unit_cell

The formula of the unit cell representing the structure, found by scaling the relative proportions of a structure's empirical formula by the total number of atoms in the simulated structure. In this notation, letters are alternated with numbers to specify the quantities of particular atom types contained in the structure. The presence of parentheses indicates doping or a solid-state mixture at a particular atomic site rather than an unmixed compound. The parentheses surround a particular set of atomic species that would normally represent a single atomic species in a pure compound, indicating the number of each atoms in the set are contained at an atomic site. For example, the tetragonally distorted perovskite A4B4C12 can be doped at its second atomic site to give A4(B1C3)D12, containing the same number of atoms at the second atomic site as the unmixed species. However, in the doped compound, three of one species and one of another are present (as opposed to four of one species) at the second atomic site.

2.2.4 morph

Morph can refer to any solely structural feature of a system that distinguishes it from systems studied in a shared source, though this attribute excludes consideration of features that are fully described by the attributes struct and f_unit_cell. In the context of crystallographic structures, this attribute can refer to polymorphic structures, such as cubic and orthorhombic, of a shared composition and stoichiometry such as ABO_3. The attribute can be further extended to distinguish between systems of shared composition and stoichiometry that differ in Strukturbericht classification, types of structures that are geometrically distorted to have different symmetries (e.g.: perovskites of varying unit cell size), or any other structural feature not categorized by struct and f_unit_cell that distinguishes structures not linked by a common cID value. In other words, defect structures that share a common cID value in order to calculate a single quantity of interest in a source (such as structures with no oxygen vacancies and one oxygen vacancy) would not require distinct tuple value classifications under morph.

2.2.5 sID

The structure ID is an integer number used to uniquely classify any possible structure (periodic lattice, molecule, etc.) recorded in this database. This form of classification was devised for this database in order to allow convenient representation of arrangements of atoms. Note that stoichiometric (NoVac) and defect (1Vac) structures share the same sID, as they are originally derived from the same parent material.

2.2.6 mID

The motif ID is an integer number used to uniquely classify any possible pattern of atoms (composition) recorded in the database. When used in conjunction with sID, this ID is capable of uniquely identifying any possible periodic crystal structure.

2.2.7 {at1, at2, . . . , atN}

Number of distinct atomic species classified. This attribute is N dimensional, where N is the number of distinct atomic species in the compound with the largest number of constituent atomic species in the database at any time.

2.2.8 Decomposition of Materials

The attributes contained in Materials are decomposed by first considering all functional dependencies between attributes, then all multivalued dependencies. In this relation, sym would serve as a key for struct in the absence of other attributes, as all crystallographic space groups match to single crystal structures. Thus, struct observes a functional dependency on sym and the two should be decomposed into a new relation (with respect to the original relation) in accordance with Boyce-Codd Normal Form (BCNF), rendering the subrelation Symmetry:

$$Symmetry(sym) \rightarrow Symmetry(struct) \tag{3}$$

There are no further functional dependencies in the following table, thus this decomposition allows the table to be compatible with BCNF. However, a multivalued dependency exists, as struct, f_unit_cell, and morph can fully determine any structure that would be placed in the database. For this reason, a key value, sID, was introduced to this relation, as it allows the rest of the variables now (i.e.: not in the Symmetry decomposed subrelation) to be decomposed into further subrelations. Thus, in accordance with 4th Normal Form (4NF), the subrelation Structure is yielded:

$$Structure(sID) \Rightarrow Structure(struct, f\_unit\_cell, morph) \tag{4}$$

As shown in f_unit_cell, many compounds included in the database contain fewer atomic species than the compound with the largest number of species. As a result, a redundancy of NULL tuple values is present in compounds with fewer atom types. Given that mID can be shared as a key across multiple tables, a more efficient way to construct this table and remove redundancy due to NULL tuple values is to decompose the remaining relation into N relations (N is the maximum number of atomic species represented by any compound in the database), which will each contain 1 atomic species linked to a particular mID. This creates N subrelations of the form Composition$N$:

$$Composition1(mID) \Rightarrow Composition1(sID, at1, stoich1) \tag{5}$$

$$Composition2(mID) \Rightarrow Composition2(sID, at2, stoich2) \tag{6}$$

$$Composition3(mID) \Rightarrow Composition3(sID, at3, stoich3) \tag{7}$$

$\vdots$

$$CompositionN(mID) \Rightarrow CompositionN(sID, atN, stoichN) \tag{8}$$

These subrelations are stored in .csv files named Symmetry.csv, Structure.csv, Composition1.csv, Composition2.csv, and Composition3.csv, respectively. The third and final initial relation refers to the part of the data used to categorize redox reaction results directly, additionally serving as a bridge between the RawData and Materials initial relations. CalculateRedox contains this data over many different variations of the reaction parameters, which correspond to its many attributes.

2.3 CalculateRedox(mID, mag, orientation, num_atoms, cID, correction, calculator, calc_quantity, figure, E_fit, V_fit, U_value, U_3d, degauss, chi0_#, chi1_#, celldm1)

The attributes defined in each CalculateRedox relation are defined as:

2.3.1 mID

The motif ID of a material tested in a calculation (same as before), which is used to link to the Structure and Composition initial relations.

2.3.2 mag

Indicates whether the calculation performed is magnetic or not (NM, or non-magnetic). Types of magnetism represented in this study include PM (paramagnetic), FM (ferromagnetic), and AFM (anti-ferromagnetic). AFM magnetism can be further qualified via the orientation of its magnetic moments (i.e.: AAFM, CAFM, GAFM), which is done in the case of fully periodic supercells with 8 or more magnetic atoms (i.e.: $2 \times 2 \times 2$ cells).

2.3.3 orientation

Can indicate several characteristics of a calculation. Firstly, this attribute can show whether a calculation is performed in the bulk of a structure or on its surface. Furthermore, several values of this attribute can correspond to surface calculations that may otherwise be difficult to distinguish from one another, such as surfaces oriented along different Miller indices. In the context of defect chemistry, values of this attribute can be further qualified by specifying the site at which a defect is present (e.g.: 4*c* vs. 8*d* oxygen vacancies in *Pbnm* LaBO$_3$).

2.3.4 num_atoms

The total number of atoms in the simulated calculation, which also infers oxygen vacancy concentration and the number of unit cells in a particular supercell in conjunction with attributes in other relations (e.g.: f_unit_cell). In this relation, this quantity refers to the number of atoms in the stoichiometric, defect-free cell.

2.3.5 cID

The calculation ID of a particular calculation presented in a collection of results (e.g.: a figure, table, etc.), corresponding to a single data point on a plot in this article. Though similar to sID in form and function, cID values link raw data (generally contained in multiple relations) to produce final results rather than referring to a single tuple or entry in only one relation. Furthermore, cID values assist in the classification of results based on the publications in which they appeared, following one of two forms of classification that both generally follow the format {`unique integer ID of calculation`}_{`publication classifier`}. Prior to publication, the publication classifier of cIDs found in a source is the first letter of the first five words of the title of that source, which for this article – Effects of Concentration, Crystal Structure, Magnetism, and Electronic Structure Method on First-Principles Oxygen Vacancy Formation Energy Trends in Perovskites – is EOCCS. Upon publication, the publication classifier becomes the Digital Object Identifier (DOI) attributed to this article, namely jp507957n.

2.3.6 correction

Indicates whether a correction for the electron-electron interaction issues mentioned previously was used in a calculation or not. Tuple values such as 'U_3d' or 'U_4f' indicate the addition of a Hubbard U parameter to a pertinent orbital for interaction correction, while other values can indicate that no correction ('None') was added, a different functional was used (e.g.: HSE06), or another type of correction was used.

2.3.7 calculator

The molecular simulation program or calculator used to produce a result associated with a particular cID. In this article, calculations can take on tuple values of 'VASP' or 'QE' (Quantum Espresso) for this attribute.

2.3.8 calc_quantity

The calculated quantity of interest associated with a cID. In the case of VASP calculations, this is always an oxygen vacancy formation energy. For QE calculations, this can be either a Hubbard U parameter derived via linear response (in eV) or an equilibrium cell volume in cubic atomic units (Bohr^3).

2.3.9 Figure

If a calculation associated with a cID was used in a plot in the article, this attribute will indicate the plot(s) in which the calculation was originally placed. When a calculation is present in multiple plots, the tuple values associated with each of those plots will be separated by semicolons. The tuple values themselves are determined by the default names of the figures outputted by the codes (see the ”Plot Generation” subsections of this document) associated with particular data points, excluding file extensions (e.g.: Asitevary, TestU_3dB_ALa).

2.3.10 U_value

In evaluating the effects of increasing U parameter magnitude on relative redox energy ordering formed by trends containing several redox energies, calculations featuring incrementally increased constant values of U were performed on several systems featured in energetic trends. This attribute designates the single U values attached to each system.

2.3.11 ENoVac_fit

Serves as a link to keys in relations of the form RawData_NoVac holding energetic and volumetric data for stoichiometric calculations (VASP only). In particular, ENoVac_fit holds fitted energy data.

2.3.12 E1Vac_fit

Serves as a link to keys in relations of the form RawData_1Vac holding energetic and volumetric data for defect calculations (VASP only). In particular, E1Vac_fit holds fitted energy data.

2.3.13 VNoVac_fit

Serves as a link to keys in relations of the form RawData_NoVac holding energetic and volumetric data for stoichiometric calculations (QE only). In particular, VNoVac_fit holds fitted volume data.

2.3.14 U_3d

The value of the linear response calculated Hubbard U value on the 3*d* orbital of a tested perovskite *B*-site.

2.3.15 degauss

The value of the Gaussian spreading applied to a QE calculation, corresponding to its matching QE input parameter.[S8]

2.3.16 chi0_#

The initial 3*d* orbital trace (initial response) of the orbital occupancy of a *B*-site at a given level of perturbation (specified by #). The value of perturbation (#) is between -0.15 and 0.15 eV, and is taken in intervals of 0.05 eV, yielding attribute values of chi0_m150, chi0_m100, chi0_m050, chi0_000, chi0_050, chi0_100, and chi0_150.

2.3.17 chi1_#

The final 3*d* orbital trace (final response) of the orbital occupancy of a *B*-site at a given level of perturbation (specified by #). The value of perturbation (#) is between 0.15 and 0.15 eV and is taken in intervals of 0.05 eV, yielding attribute values of chi1_m150, chi1_m100, chi1_m050, chi1_000, chi1_050, chi1_100, and chi1_150.

2.3.18 celldm1

The value of the edge length of a cubic cell, corresponding to its matching QE input parameter, celldm(1).[S8] It is used to differentiate between strained and unstrained results in linear response calculations.

2.3.19 Decomposition of CalculateRedox

There are no functional dependencies in the following table for VASP calculations (excluding ENoVac_fit and E1Vac_fit, which will be handled later). However, for QE calculations, functional dependencies exist depending on the quantity of interest (calc_quantity) calculated. For QE calculations that involve calculation of linear response U values (U_3d), a calculation can be completely determined by the value of its resolved U parameter, its total magnetization after SCF cycle convergence (magmom, only applied to the perovskite B-site), the magnitude of its Gaussian spreading, and the values of the initial and final responses of its perturbed B-site atom. Therefore, decomposition of the above table will occur in accordance with BCNF via the following operation, rendering the subrelation OVac_LRCalc:

$$OVac\_LRCalc(cID) \rightarrow OVac\_LRCalc(U\_3d, magmom, degauss, chi0\_\#, chi1\_\#) \quad (9)$$

Three multivalued dependencies exist, as cID can be uniquely determined by the attributes sID, mag, num_atoms, orientation, correction, and calc_quantity. In addition, cID can also be determined by its fitted energies corresponding to its no vacancy and one vacancy structures, given that cID is matched to sID in the first relation. In the case of incremented U values, the second multivalued dependency must be split into a separate relation to avoid redundancy in calculations not featuring a Hubbard U correction. For QE calculations that involve calculation of equilibrium cell volumes (VNoVac_fit) at different values of U, a third multivalued dependency similar to the second is formed. Finally, a subset of cIDs can be mapped to the figures containing their tuple values. Thus, decomposition of the table will occur in accordance with 4th Normal Form (4NF) via the following operations, rendering the subrelations OVac_FormE, OVac_EV, OVac_OrderU, OVac_TestU, and OVac_Figure:

$$\begin{aligned}&OVac\_FormE(cID) \rightarrow OVac\_FormE(mID, mag, num\_atoms,\\&orientation, correction, calc\_quantity,\\&calculator)\end{aligned} \tag{10}$$

$$OVac\_EV(cID) \rightarrow OVac\_EV(ENoVac\_fit, E1Vac\_fit) \tag{11}$$

$$OVac\_OrderU(cID) \rightarrow OVac\_OrderU(U\_value, ENoVac\_fit, E1Vac\_fit) \tag{12}$$

$$OVac\_TestU(cID) \rightarrow OVac\_TestU(U\_value, VNoVac\_fit) \tag{13}$$

$$OVac\_Figure(cID) \rightarrow OVac\_Figure(Figure) \tag{14}$$

These subrelations are stored in OVac_FormE.csv, OVac_EV.csv, OVac_OrderU.csv, OVac_TestU.csv, and OVac_Figure.csv, respectively. Note that all of the calculations that do not involve values of Hubbard U or figures in the article have energies and volumes listed in the OVac_EV relation, while those that do have single values of U attached to any orbital have energetic and volumetric data listed in relations OVac_OrderU and OVac_TestU. In all of the above relations, NULL values are represented by a '0' value.

2.4 Input Data and Output Structures

Though not organized into BCNF or 4NF, information corresponding to the input file data and output file structures of each VASP and QE calculation is organized into relations RawData_Input_VASP, RawData_Input_QE, RawData_PosFinal_VASP, and RawData_PosFinal_QE.

These relations are stored in RawData_Input_VASP.csv, RawData_Input_QE.csv, RawData_PosFinal_VASP.csv, and RawData_PosFinal_QE.csv. In the tables characterizing these relations, NULL values are represented by the character string '–'. These relations contain attributes and possess characteristics detailed below:

2.4.1 RawData_Input_VASP

The attributes contained within this table consist of all possible INCAR contents from all VASP calculations completed in the article, the title strings associated with each POTCAR used in each calculation (found by performing the command 'grep TITEL' on each POTCAR file used in each calculation), each k-point value (corresponding to the x, y, and z axes) listed in pertinent KPOINTS files, and the converged total energies associated with each calculation. The title strings for used POTCAR files are stored under attributes Pseudopotential_1, Pseudopotential_2, and Pseudopotential_3, with spacing in the title strings substituted by semicolons. Each k-point value for each axis is stored under attributes KPOINT_x, KPOINT_y, and KPOINT_z. The total energies serve as keys linking the input data to relations (i.e.: RawData_EV_NoVac and RawData_EV_1Vac) that connect raw calculation data to calculated results portrayed in the article. The union of INCAR contents of all calculations completed in the article creates the following set of INCAR tags, which serve as labels for attributes in the table:

1. ISTART, ICHARG

2. ENCUT

3. ISIF

4. IBRION

5. EDIFF, EDIFFG

6. NSW

7. NELM, NELMDL, NELMIN, MAXMIX
8. ISPIN
9. ISMEAR, SIGMA
10. ISYM, SYMPREC
11. MAGMOM_number_1, MAGMOM_number_2, MAGMOM_number_3, MAGMOM_number_4, MAGMOM_number_5, MAGMOM_number_6, MAGMOM_number_7, MAGMOM_number_8
12. MAGMOM_maginit_1, MAGMOM_maginit_2, MAGMOM_maginit_3, MAGMOM_maginit_4, MAGMOM_maginit_5, MAGMOM_maginit_6, MAGMOM_maginit_7, MAGMOM_maginit_8
13. VOSKOWN
14. NUPDOWN
15. LDAU, LDAUTYPE, LDAUPRINT
16. LMAXMIX, LASPH, LORBIT
17. LDAUJ_1, LDAUJ_2, LDAUJ_3, LDAUL_1, LDAUL_2, LDAUL_3, LDAUU_1, LDAUU_2, LDAUU_3
18. NBANDS
19. NPAR

Note, for every attribute containing the label MAGMOM, the tuple value associated with that attribute is actually a portion of a MAGMOM command in a given calculation. When converting values in the relation above into commands to be used in VASP calculations, note that the attributes above correspond to the MAGMOM command accordingly:

```
MAGMOM = MAGMOM_number_1*MAGMOM_maginit_1 MAGMOM_number_2*MAGMOM_maginit_2 ...
```

2.4.2 RawData_Input_QE

The attributes contained within this table consist of all possible input file contents from all QE calculations completed in the article (excluding structural parameters listed in relation RawData_PosFinal_QE), the file names associated with each pseudopotential used in each calculation (found under the ATOMIC_SPECIES card), each k-point value (corresponding to the x, y, and z axes) and matching k-point grid off-sets (a, b, and c), and the converged total energies associated with each calculation. The title strings for used POTCAR files are stored under attributes Pseudopotential_1, Pseudopotential_2, and Pseudopotential_3, with spacing in the title strings substituted by semicolons. Each k-point value for each axis is stored under attributes KPOINT_x, KPOINT_y, and KPOINT_z. The total energies serve as keys linking the input data to relations (i.e.: RawData_EV_NoVac and RawData_EV_1Vac) that connect raw calculation data to calculated results portrayed in the article. The union of QE input file contents of all calculations completed in the article creates the following set of input parameters, which serve as labels for attributes in the table:

1. calculation, verbosity, restart_mode, disk_io, pseudo_dir, outdir, title, prefix, wf_collect

2. ibrav, nat, ntyp, nspin, ecutwfc, ecutrho, starting_magnetization1, starting_magnetization2, occupations, smearing, degauss, nosym

3. U_projection_type, lda_plus_u, lda_plus_u_kind, Hubbard_U1, Hubbard_U2, Hubbard_U3, Hubbard_U4

4. electron_maxstep, conv_thr, diagonalization, diago_thr_init, diago_full_acc, startingpot, startingwfc, mixing_mode, mixing_beta, mixing_ndim

5. etot_conv_thr, forc_conv_thr, nstep, upscale, ion_dynamics

6. KPOINTS_x, KPOINTS_y, KPOINTS_z, KPOINTS_a, KPOINTS_b, KPOINTS_c

7. Pseudopotential_1, Pseudopotential_2, Pseudopotential_3

2.4.3 RawData_PosFinal_QE

The attributes contained within this relation consist of all input file contents needed to represent the final structures associated with each QE calculation, including all atomic coordinates listed under the ATOMIC_POSITIONS card (attributes of form coord_N, where N is an integer between 1 and 40), the magnitude of the number used to scale the cubic systems tested in QE (this attribute is labeled celldm1, as it is equivalent to the cubic lattice constants of these systems), and the converged total energies associated with each calculation. Each value within each tuple of each attribute that is generally separated from other values within the tuple via spacing is instead separated by semicolons.

2.4.4 RawData_PosFinal_VASP

The attributes contained within this relation consist of all CONTCAR contents needed to represent the final structures associated with each VASP calculation, include each primitive vector row from each calculation (attributes Primvec_1, Primvec_2, and Primvec_3), the number and types of atoms in a calculation that corresponding to the order of pseudopotentials listed in POTCAR and the coordinates listed in POSCAR or CONTCAR (attribute attyps), the magnitude of the number used to scale the primitive vectors (attribute scale), each set of x, y, and z atomic coordinates (attributes of form coord_N, where N is an integer between 1 and 40), and the converged total energies associated with each calculation. Each value within each tuple of each attribute that is generally separated from other values within the tuple via spacing is instead separated by semicolons.

2.5 Reproducing the Database

The .csv files accompanying this article and its supporting information file can be imported into any RDBMS platform to reproduce results or perform other operations on aggregated data. However, in order to facilitate the easy integration of .csv files into org-mode and enable full reproducibility of all plots from initial data within this contained document, this article will assemble all .csv files into a single .sqlite file, which can be read into Python scripts within the ”Plot Generation” subsections of this document to generate all plots contained within the article. Several tables not associated with plots in the article can also be reproduced using this file. Upon being equipped with proper extensions, Mozilla Firefox[S9] can also import the .sqlite file, perform queries on its contained data, and provide a graphic-user interface suitable for browsing and examining the data.

The database we created can be downloaded here. EOCCS_data.sqlite: 📌.

In order to generate a single .sqlite file containing all .csv file data needed to reproduce all plots listed in the following section, execute the Python scripts listed below in the order that they are presented. Note that the .csv files and this document should be located in the current working directory prior to executing these scripts, which upload tables that correspond to aforementioned RDBMS subrelations one at a time:

2.5.1 Structure table

```
import sqlite3
import os

# We start from scratch. Delete the database if it already exists.
if os.path.exists('EOCCS_data.sqlite'):
    os.remove('EOCCS_data.sqlite')

db = sqlite3.connect('EOCCS_data.sqlite')

db.execute('''create table Structure(sID INTEGER PRIMARY KEY, struct TEXT, f_unit_cell TEXT, morph TEXT)''')

with open('Structure.csv') as f:
    lines = f.readlines()

for line in lines[1:]:
    fields = [x.strip() for x in line.split(',')]
    db.execute('''insert into Structure(sID,struct,f_unit_cell,morph)
                  VALUES(?,?,?,?)''', fields)

db.commit()
db.close()
```

2.5.2 Symmetry Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table Symmetry(sym TEXT, struct TEXT)’’’)

with open(’Symmetry.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into Symmetry(sym, struct)
                  VALUES(?,?)’’’, fields)

db.commit()
db.close()
```

2.5.3 Composition1 Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table Composition1(mID INTEGER, sID INTEGER, at1 TEXT, stoich1 INTEGER)’’’)

with open(’Composition1.csv’) as f:
    lines = f.readlines()

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into Composition1(mID,sID,at1,stoich1)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.4 Composition2 Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table Composition2(mID INTEGER, sID INTEGER, at2 TEXT, stoich2 INTEGER)’’’)

with open(’Composition2.csv’) as f:
    lines = f.readlines()

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into Composition2(mID,sID,at2,stoich2)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.5 Composition3 Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table Composition3(mID INTEGER, sID INTEGER, at3 TEXT, stoich3 INTEGER)’’’)

with open(’Composition3.csv’) as f:
    lines = f.readlines()

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into Composition3(mID,sID,at3,stoich3)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.6 RawData_EV_NoVac Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table RawData_EV_NoVac(E_tot FLOAT, V_tot FLOAT, VNoVac_fit FLOAT, ENoVac_fit FLOAT)’’’)

with open(’RawData_EV_NoVac.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into RawData_EV_NoVac(E_tot,V_tot,VNoVac_fit,ENoVac_fit)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.7 RawData_EV_1Vac Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table RawData_EV_1Vac(E_tot FLOAT, V_tot FLOAT, E1Vac_fit FLOAT, V1Vac_fit FLOAT)’’’)

with open(’RawData_EV_1Vac.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into RawData_EV_1Vac(E_tot,V_tot,E1Vac_fit,V1Vac_fit)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.8 RawData_Mag Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table RawData_Mag(magmom,magpos,E_tot)’’’)

with open(’RawData_Mag.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into RawData_Mag(magmom, magpos, E_tot)
                  VALUES(?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.9 OVac_EV Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table OVac_EV(cID TEXT, ENoVac_fit FLOAT, E1Vac_fit FLOAT)’’’)

with open(’OVac_EV.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into OVac_EV(cID, ENoVac_fit, E1Vac_fit)
                  VALUES(?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.10 OVac_FormE Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table OVac_FormE(cID TEXT, mID INTEGER,
mag TEXT, num_atoms INTEGER, orientation TEXT, correction TEXT,
calc_quantity FLOAT, calculator TEXT)’’’)

with open(’OVac_FormE.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into OVac_FormE(cID, mID, mag, num_atoms,
orientation, correction, calc_quantity, calculator)
                  VALUES(?,?,?,?,?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.11 OVac_OrderU Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table OVac_OrderU(cID TEXT, U_value FLOAT, E1Vac_fit FLOAT, ENoVac_fit FLOAT)’’’)

with open(’OVac_OrderU.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into OVac_OrderU(cID, U_value, E1Vac_fit, ENoVac_fit)
                  VALUES(?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.12 OVac_LRCalc Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table OVac_LRCalc(cID TEXT, U_3d FLOAT, magmom INTEGER,
degauss FLOAT, celldm1 FLOAT, chi0_m150 FLOAT, chi0_m100 FLOAT,
chi0_m050 FLOAT, chi0_000 FLOAT, chi0_050 FLOAT, chi0_100 FLOAT,
chi0_150 FLOAT, chi1_m150 FLOAT, chi1_m100 FLOAT, chi1_m050 FLOAT,
chi1_000 FLOAT, chi1_050 FLOAT, chi1_100 FLOAT, chi1_150 FLOAT)’’’)

with open(’OVac_LRCalc.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into OVac_LRCalc(cID,  U_3d, magmom, degauss,
celldm1, chi0_m150, chi0_m100, chi0_m050, chi0_000, chi0_050, chi0_100,
chi0_150, chi1_m150, chi1_m100, chi1_m050, chi1_000, chi1_050, chi1_100,
chi1_150)
VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.13 OVac_TestU Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table OVac_TestU(cID TEXT, U_value FLOAT, VNoVac_fit FLOAT)’’’)

with open(’OVac_TestU.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into OVac_TestU(cID, U_value, VNoVac_fit)
                  VALUES(?,?,?)’’’, fields)

db.commit()
db.close()
```

2.5.14 RawData_Input_VASP Table

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

db.execute(’’’create table RawData_Input_VASP(NELMDL TEXT, LDAUJ_3 TEXT,
LDAUJ_1 TEXT, SYMPREC TEXT, LDAUL_1 TEXT, NELM TEXT, LDAUL_3 TEXT,
VOSKOWN TEXT, IBRION TEXT, LDAUTYPE TEXT, LDAUU_1 TEXT, LDAUU_3 TEXT,
LDAUU_2 TEXT, SIGMA TEXT, LDAUPRINT TEXT, MAGMOM_maginit_1 TEXT,
LASPH TEXT, MAGMOM_maginit_3 TEXT, MAGMOM_maginit_2 TEXT,
MAGMOM_maginit_5 TEXT, MAGMOM_maginit_4 TEXT, MAGMOM_maginit_7 TEXT,
MAGMOM_maginit_6 TEXT, LDAUJ_2 TEXT, MAXMIX TEXT, ISIF TEXT, ICHARG TEXT,
Pseudopotential_1 TEXT, Pseudopotential_2 TEXT, Pseudopotential_3 TEXT,
EDIFF TEXT, LDAUL_2 TEXT, ENCUT TEXT, NUPDOWN TEXT, NELMIN TEXT,
ISYM TEXT, NBANDS TEXT, ISMEAR TEXT, NPAR TEXT, ISTART TEXT,
MAGMOM_number_4 TEXT, MAGMOM_number_5 TEXT, MAGMOM_number_6 TEXT,
MAGMOM_number_7 TEXT, MAGMOM_number_1 TEXT, MAGMOM_number_2 TEXT,
MAGMOM_number_3 TEXT, LDAU TEXT, LMAXMIX TEXT, KPOINT_z TEXT, EDIFFG TEXT,
KPOINT_x TEXT, KPOINT_y TEXT, ISPIN TEXT, LORBIT TEXT, NSW TEXT,
E_tot FLOAT)’’’)

with open(’RawData_Input_VASP.csv’) as f:
    lines = f.readlines()[0].split(’\r’)

for line in lines[1:]:
    fields = [x.strip() for x in line.split(’,’)]
    db.execute(’’’insert into RawData_Input_VASP(NELMDL, LDAUJ_3, LDAUJ_1,
SYMPREC, LDAUL_1, NELM, LDAUL_3, VOSKOWN, IBRION, LDAUTYPE, LDAUU_1,
LDAUU_3, LDAUU_2, SIGMA, LDAUPRINT, MAGMOM_maginit_1, LASPH,
MAGMOM_maginit_3, MAGMOM_maginit_2, MAGMOM_maginit_5, MAGMOM_maginit_4,
MAGMOM_maginit_7, MAGMOM_maginit_6, LDAUJ_2, MAXMIX, ISIF, ICHARG,
Pseudopotential_1, Pseudopotential_2, Pseudopotential_3, EDIFF, LDAUL_2,
ENCUT, NUPDOWN, NELMIN, ISYM, NBANDS, ISMEAR, NPAR, ISTART, MAGMOM_number_4,
MAGMOM_number_5, MAGMOM_number_6, MAGMOM_number_7, MAGMOM_number_1,
MAGMOM_number_2, MAGMOM_number_3, LDAU, LMAXMIX, KPOINT_z, EDIFFG, KPOINT_x,
KPOINT_y, ISPIN, LORBIT, NSW, E_tot)
                  VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,
    ?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)''', fields)

db.commit()
db.close()
```

2.5.15 RawData_Input_QE Table

```
import sqlite3

db = sqlite3.connect('EOCCS_data.sqlite')

db.execute('''create table RawData_Input_QE(Hubbard_U3 TEXT, nspin TEXT,
conv_thr TEXT, occupations TEXT, calculation TEXT,
starting_magnetization1 TEXT, lda_plus_u_kind TEXT, upscale TEXT,
KPOINTS_x TEXT, forc_conv_thr TEXT, mixing_beta TEXT, nstep TEXT,
electron_maxstep TEXT, ecutwfc TEXT, disk_io TEXT, diago_thr_init TEXT,
title TEXT, KPOINTS_y TEXT, ibrav TEXT, Pseudopotential_1 TEXT,
 Pseudopotential_2 TEXT, Pseudopotential_3 TEXT, diagonalization TEXT,
 restart_mode TEXT, Hubbard_U1 TEXT, ion_dynamics TEXT, nat TEXT,
Hubbard_U2 TEXT, starting_magnetization2 TEXT, pseudo_dir TEXT,
 mixing_ndim TEXT, mixing_mode TEXT, smearing TEXT, etot_conv_thr TEXT,
 startingwfc TEXT, outdir TEXT, wf_collect TEXT, KPOINTS_c TEXT,
 KPOINTS_z TEXT, prefix TEXT, lda_plus_u TEXT, nosym TEXT, degauss TEXT,
 KPOINTS_b TEXT, Hubbard_U4 TEXT, KPOINTS_a TEXT, U_projection_type TEXT,
 ntyp TEXT, diago_full_acc TEXT, verbosity TEXT, startingpot TEXT,
ecutrho TEXT, E_tot FLOAT)''')

with open('RawData_Input_QE.csv') as f:
    lines = f.readlines()[0].split('\r')

for line in lines[1:]:
    fields = [x.strip() for x in line.split(',')]
    db.execute('''insert into RawData_Input_QE(Hubbard_U3, nspin, conv_thr,
 occupations, calculation, starting_magnetization1, lda_plus_u_kind, upscale,
 KPOINTS_x, forc_conv_thr, mixing_beta, nstep, electron_maxstep, ecutwfc,
 disk_io, diago_thr_init, title, KPOINTS_y, ibrav, Pseudopotential_1,
 Pseudopotential_2, Pseudopotential_3, diagonalization, restart_mode,
 Hubbard_U1, ion_dynamics, nat, Hubbard_U2, starting_magnetization2,
 pseudo_dir, mixing_ndim, mixing_mode, smearing, etot_conv_thr, startingwfc,
 outdir, wf_collect, KPOINTS_c, KPOINTS_z, prefix, lda_plus_u, nosym, degauss,
 KPOINTS_b, Hubbard_U4, KPOINTS_a, U_projection_type, ntyp, diago_full_acc,
 verbosity, startingpot, ecutrho, E_tot)

                  VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,
?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)''', fields)

db.commit()
db.close()
```

2.5.16 RawData_PosFinal_VASP Table

```
import sqlite3

db = sqlite3.connect('EOCCS_data.sqlite')

db.execute('''create table RawData_PosFinal_VASP(coord_33 TEXT, coord_32 TEXT,
 coord_31 TEXT, coord_30 TEXT, coord_37 TEXT, coord_36 TEXT, coord_35 TEXT,
 coord_34 TEXT, Primvec_2 TEXT, coord_39 TEXT, coord_38 TEXT, Primvec_1 TEXT,
 attyps TEXT, coord_9 TEXT, coord_8 TEXT, coord_5 TEXT, coord_4 TEXT,
coord_7 TEXT, coord_6 TEXT, coord_1 TEXT, Primvec_3 TEXT, coord_3 TEXT,
 coord_2 TEXT, coord_40 TEXT, Scale TEXT, coord_20 TEXT, coord_21 TEXT,
 coord_22 TEXT, coord_23 TEXT, coord_24 TEXT, coord_25 TEXT, coord_26 TEXT,
 coord_27 TEXT, coord_28 TEXT, coord_29 TEXT, coord_19 TEXT, coord_18 TEXT,
 coord_11 TEXT, coord_10 TEXT, coord_13 TEXT, coord_12 TEXT, coord_15 TEXT,
 coord_14 TEXT, coord_17 TEXT, coord_16 TEXT, E_tot FLOAT)''')

with open('RawData_PosFinal_VASP.csv') as f:
    lines = f.readlines()[0].split('\r')

for line in lines[1:]:
    fields = [x.strip() for x in line.split(',')]
    db.execute('''insert into RawData_PosFinal_VASP(coord_33, coord_32,
 coord_31, coord_30, coord_37, coord_36, coord_35, coord_34, Primvec_2,
 coord_39, coord_38, Primvec_1, attyps, coord_9, coord_8, coord_5, coord_4,
 coord_7, coord_6, coord_1, Primvec_3, coord_3, coord_2, coord_40, Scale,
 coord_20, coord_21, coord_22, coord_23, coord_24, coord_25, coord_26,
 coord_27, coord_28, coord_29, coord_19, coord_18, coord_11, coord_10,
 coord_13, coord_12, coord_15, coord_14, coord_17, coord_16, E_tot)
                  VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,
?,?,?,?,?,?,?,?,?,?,?,?)''', fields)

db.commit()
db.close()
```

2.5.17 RawData_PosFinal_QE Table

```
import sqlite3

db = sqlite3.connect('EOCCS_data.sqlite')

db.execute('''create table RawData_PosFinal_QE(coord_33 TEXT, coord_32 TEXT,
 coord_31 TEXT, coord_30 TEXT, coord_37 TEXT, coord_36 TEXT, coord_35 TEXT,
 coord_34 TEXT, coord_39 TEXT, coord_38 TEXT, coord_9 TEXT, coord_8 TEXT,
 coord_5 TEXT, coord_4 TEXT, coord_7 TEXT, coord_6 TEXT, coord_1 TEXT,
coord_3 TEXT, coord_2 TEXT, coord_40 TEXT, coord_20 TEXT, coord_21 TEXT,
 coord_22 TEXT, coord_23 TEXT, coord_24 TEXT, coord_25 TEXT, coord_26 TEXT,
 coord_27 TEXT, coord_28 TEXT, coord_29 TEXT, celldm1 TEXT, coord_19 TEXT,
 coord_18 TEXT, coord_11 TEXT, coord_10 TEXT, coord_13 TEXT, coord_12 TEXT,
 coord_15 TEXT, coord_14 TEXT, coord_17 TEXT, coord_16 TEXT, E_tot FLOAT)''')

with open('RawData_PosFinal_QE.csv') as f:
    lines = f.readlines()[0].split('\r')

for line in lines[1:]:
    fields = [x.strip() for x in line.split(',')]
    db.execute('''insert into RawData_PosFinal_QE(coord_33, coord_32,
 coord_31, coord_30, coord_37, coord_36, coord_35, coord_34, coord_39,
 coord_38, coord_9, coord_8, coord_5, coord_4, coord_7, coord_6, coord_1,
 coord_3, coord_2, coord_40, coord_20, coord_21, coord_22, coord_23, coord_24,
 coord_25, coord_26, coord_27, coord_28, coord_29, celldm1, coord_19,
 coord_18, coord_11, coord_10, coord_13, coord_12, coord_15, coord_14,
 coord_17, coord_16, E_tot)
                  VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,
?,?,?,?,?,?,?,?,?)''', fields)

db.commit()
db.close()
```

2.6 Reproducing Calculation Results

The input needed to reproduce any calculation, which includes converged atomic position data, as well as input parameters derived from ”.in” files in QE and INCAR, KPOINTS, and POTCAR tags in VASP, are queried from the database in the example below. The example below first retrieves the INCAR tags, KPOINTS mesh, and POTCAR titles needed to reproduce a PM, Hubbard U ($3d$ orbital) corrected, cubic unit cell $LaFeO_3$ calculation with one vacancy induced. The single set of input parameters produced in this manner is to be applied to each set of POSCAR file contents generated in the second query below, which retrieves each line used in each POSCAR file. Each POSCAR file corresponds to a constant volume (ISIF = 4) simulation needed to complete an equation of state calculation for the $LaFeO_3$ system detailed above.

2.6.1 Example: INCAR, KPOINTS, POTCAR Retrieval

```
import sqlite3

db = sqlite3.connect(’EOCCS_data.sqlite’)

NULLchar = ’--’
datapts_dict = {}

for row in db.execute(’’’
select input.NELMDL, input.LDAUJ_3, input.LDAUJ_1, input.SYMPREC,
input.LDAUL_1, input.NELM, input.LDAUL_3, input.VOSKOWN, input.IBRION,
input.LDAUTYPE, input.LDAUU_1, input.LDAUU_3, input.LDAUU_2, input.SIGMA,
input.LDAUPRINT, input.MAGMOM_maginit_1, input.LASPH, input.MAGMOM_maginit_3,
input.MAGMOM_maginit_2, input.MAGMOM_maginit_5, input.MAGMOM_maginit_4,
input.MAGMOM_maginit_7, input.MAGMOM_maginit_6, input.LDAUJ_2, input.MAXMIX,
input.ISIF, input.ICHARG, input.Pseudopotential_1, input.Pseudopotential_2,
input.Pseudopotential_3, input.EDIFF, input.LDAUL_2, input.ENCUT,
input.NUPDOWN, input.NELMIN, input.ISYM, input.NBANDS, input.ISMEAR,
input.NPAR, input.ISTART, input.MAGMOM_number_4, input.MAGMOM_number_5,
input.MAGMOM_number_6, input.MAGMOM_number_7, input.MAGMOM_number_1,
input.MAGMOM_number_2, input.MAGMOM_number_3, input.LDAU, input.LMAXMIX,
input.KPOINT_z, input.EDIFFG, input.KPOINT_x, input.KPOINT_y, input.ISPIN,
input.LORBIT, input.NSW, input.E_tot from Structure as s
inner join Symmetry as sy on sy.struct=s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ov.cID=ovu.cID
inner join RawData_EV_1Vac as rdev on ovu.E1Vac_fit=rdev.E1Vac_fit
inner join RawData_Input_VASP as input on rdev.E_tot=input.E_tot
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and c2.at2=’Fe’
and ov.num_atoms=5
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and ovu.U_value=’3.5’
;’’’):
    datapts_list = []

    (a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,
     aa,ab,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,ap,aq,
     ar,at,au,av,aw,ax,ay,ba,bb,bc,bd,be,bf,bg,bh) = row

    series =  (a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,
               aa,ab,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,
               ap,aq,ar,at,au,av,aw,ax,ay,
               ba,bb,bc,bd,be,bf,bg,bh)

    for z in series:
        datapts_list.append(z)

    datapts_dict[bh] = datapts_list
    del datapts_dict[bh][-1]

Input_check = {}
Input_match = {}

att_list = [ ’NELMDL’, ’LDAUJ_3’, ’LDAUJ_1’, ’SYMPREC’, ’LDAUL_1’, ’NELM’,
             ’LDAUL_3’, ’VOSKOWN’, ’IBRION’, ’LDAUTYPE’, ’LDAUU_1’,
             ’LDAUU_3’, ’LDAUU_2’, ’SIGMA’, ’LDAUPRINT’, ’MAGMOM_maginit_1’,
             ’LASPH’, ’MAGMOM_maginit_3’, ’MAGMOM_maginit_2’,
             ’MAGMOM_maginit_5’, ’MAGMOM_maginit_4’, ’MAGMOM_maginit_7’,
             ’MAGMOM_maginit_6’, ’LDAUJ_2’, ’MAXMIX’, ’ISIF’, ’ICHARG’,
             ’Pseudopotential_1’, ’Pseudopotential_2’, ’Pseudopotential_3’,
             ’EDIFF’, ’LDAUL_2’, ’ENCUT’, ’NUPDOWN’, ’NELMIN’, ’ISYM’,
             ’NBANDS’, ’ISMEAR’, ’NPAR’, ’ISTART’, ’MAGMOM_number_4’,
             ’MAGMOM_number_5’, ’MAGMOM_number_6’, ’MAGMOM_number_7’,
             ’MAGMOM_number_1’, ’MAGMOM_number_2’, ’MAGMOM_number_3’,
             ’LDAU’, ’LMAXMIX’, ’KPOINT_z’, ’EDIFFG’, ’KPOINT_x’,
             ’KPOINT_y’, ’ISPIN’, ’LORBIT’, ’NSW’ ]

for i,j in enumerate( datapts_dict.keys() ):
    if datapts_dict[j] not in Input_check.values():
        list_match = []
        list_check = []

        for k,l in enumerate( att_list ):
            list_check.append( datapts_dict[j][k] )

            if datapts_dict[j][k] != NULLchar:
                list_match.append( (att_list[k], datapts_dict[j][k]) )

        Input_check[i] = list_check
        Input_match[i] = list_match

SortCAR = { "ISTART": 0, "ICHARG": 1, "ENCUT": 2, "ISMEAR": 3,
            "SIGMA": 4, "ISPIN": 5, "IBRION": 6, "EDIFF": 7,
            "EDIFFG": 8, "ISYM": 9, "NELMIN": 10, "NELMDL": 11,
            "MAXMIX": 12, "NELM": 13, "NSW": 14,  "LDAU": 10,
            "LDAUTYPE": 11, "LDAUPRINT": 12, "LASPH": 13,
            "LMAXMIX": 14, "LDAUL_1": 15, "LDAUL_2": 16,
            "LDAUL_3": 17, "LDAUJ_1": 18, "LDAUJ_2": 19,
            "LDAUJ_3": 20, "LDAUU_1": 21, "LDAUU_2": 22,
            "LDAUU_3": 23, "KPOINT_x": 24, "KPOINT_y": 25,
            "KPOINT_z": 26, "Pseudopotential_1": 27,
            "Pseudopotential_2": 28, "Pseudopotential_3": 29 }

Input_INCKPTPOT = sorted(Input_match[0], key=lambda tag: SortCAR[tag[0]])

num_KPTPOT = 6
```

```
num_INCAR  = len(Input_INCKPTPOT) - num_KPTPOT

for i in range( num_INCAR ):
    print Input_INCKPTPOT[i][0],'=',Input_INCKPTPOT[i][1]

print '--'

for i in range( num_KPTPOT ):
    print Input_INCKPTPOT[num_INCAR+i][0],Input_INCKPTPOT[num_INCAR+i][1]
```

```
ISTART = 0

ICHARG = 2

ENCUT = 600

ISMEAR = 0

SIGMA = 0.05

ISPIN = 2

IBRION = 1

EDIFF = 5.00E-06

EDIFFG = -0.005

ISYM = 0

NELMIN = 5

LDAU = .TRUE.

NELMDL = -10

LDAUTYPE = 2

LDAUPRINT = 1

MAXMIX = -100

NELM = 200

LASPH = .TRUE.

LMAXMIX = 4

NSW = 100
```

```
LDAUL_1 = -1
LDAUL_2 = 2
LDAUL_3 = -1
LDAUJ_1 = 0
LDAUJ_2 = 0
LDAUJ_3 = 0
LDAUU_1 = 0
LDAUU_2 = 3.5
LDAUU_3 = 0
--
KPOINT_x 8
KPOINT_y 8
KPOINT_z 8
Pseudopotential_1 PAW_PBE:La;06Sep2000
Pseudopotential_2 PAW_PBE:Fe;06Sep2000
Pseudopotential_3 PAW_PBE:O;08Apr2002
```

The script below generates the ordered CONTCAR files used to generate the fitted energy of the system described above, which is derived from the Birch-Murnaghan equation of state fitting procedure and is used to generate plotted data associated with cID values. When available in the database, these CONTCAR input coordinates form structurally relaxed systems that can be outputted to POSCAR files.

2.6.2 Example: CONTCAR Retrieval

```python
import sqlite3
import re

db = sqlite3.connect('EOCCS_data.sqlite')

```

```
NULLchar = ’--’
datapts_dict = {}

for row in db.execute(’’’
select pos.coord_33, pos.coord_32, pos.coord_31, pos.coord_30, pos.coord_37,
pos.coord_36, pos.coord_35, pos.coord_34, pos.Primvec_2, pos.coord_39,
pos.coord_38, pos.Primvec_1, pos.attyps, pos.coord_9, pos.coord_8,
pos.coord_5, pos.coord_4, pos.coord_7, pos.coord_6, pos.coord_1,
pos.Primvec_3, pos.coord_3, pos.coord_2, pos.coord_40, pos.coord_20,
pos.coord_21, pos.coord_22, pos.coord_23, pos.coord_24, pos.coord_25,
pos.coord_26, pos.coord_27, pos.coord_28, pos.coord_29, pos.coord_19,
pos.coord_18, pos.coord_11, pos.coord_10, pos.coord_13, pos.coord_12,
pos.coord_15, pos.coord_14, pos.coord_17, pos.coord_16, pos.Scale, pos.E_tot
from Structure as s
inner join Symmetry as sy on sy.struct=s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ov.cID=ovu.cID
inner join RawData_EV_1Vac as rdev on ovu.E1Vac_fit=rdev.E1Vac_fit
inner join RawData_PosFinal_VASP as pos on rdev.E_tot=pos.E_tot
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and c2.at2=’Fe’
and ov.num_atoms=5
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and ovu.U_value=’3.5’
;’’’):
    datapts_list = []
    (a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,
     aa,ab,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,ap,aq,
     ar,at,au,av) = row
    series = ( a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,
               aa,ab,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,ap,aq,
               ar,at,au,av)

    for z in series:
        datapts_list.append(z)

    datapts_dict[av] = datapts_list
    del datapts_dict[av][-1]

Input_check = {}
Input_match = {}

att_list = [ ’coord_33’, ’coord_32’, ’coord_31’, ’coord_30’, ’coord_37’,
             ’coord_36’, ’coord_35’, ’coord_34’, ’Primvec_2’, ’coord_39’,
             ’coord_38’, ’Primvec_1’, ’attyps’, ’coord_9’, ’coord_8’,
             ’coord_5’, ’coord_4’, ’coord_7’, ’coord_6’, ’coord_1’,
             ’Primvec_3’, ’coord_3’, ’coord_2’, ’coord_40’, ’coord_20’,
             ’coord_21’, ’coord_22’, ’coord_23’, ’coord_24’, ’coord_25’,
             ’coord_26’, ’coord_27’, ’coord_28’, ’coord_29’, ’coord_19’,
             ’coord_18’, ’coord_11’, ’coord_10’, ’coord_13’, ’coord_12’,
             ’coord_15’, ’coord_14’, ’coord_17’, ’coord_16’, ’Scale’ ]

SortCAR = { "Scale": 0, "Primvec_1": 1, "Primvec_2": 2, "Primvec_3": 3,
"attyps": 4, "coord_1": 5, "coord_2": 6, "coord_3": 7, "coord_4": 8 }

for i,j in enumerate( datapts_dict.keys() ):
    list_match = []

    for k,l in enumerate( att_list ):
        if datapts_dict[j][k] != NULLchar:
            list_match.append( (att_list[k], datapts_dict[j][k]) )

    Input_match[datapts_dict[j][-1]] = list_match
    Input_match[datapts_dict[j][-1]].sort(key=lambda val: SortCAR[val[0]])

Input_POSCAR = sorted(Input_match.iteritems(), key=lambda (k,v): k)

Scales_POSCAR = len( Input_match.keys() )
Lines_POSCAR =  len( SortCAR )

for i in range( Scales_POSCAR ):
    print ’Title’

    for j in range( Lines_POSCAR ):
        linePOSCAR = re.sub(’;’, ’ ’, Input_POSCAR[i][1][j][1].rstrip())
        print linePOSCAR

    print ’--’
```

```
Title
3.83
1.0000000000000000 0.0000000000000000 0.0000000000000000
0.0000000000000000 1.0000000000000000 0.0000000000000000
0.0000000000000000 0.0000000000000000 1.0000000000000000
1 1 2
0.0026542948147870 0.0024629265466301 0.0026949615024616
0.5025181569632975 0.4976225471919549 0.5026662889877253
0.5025151794751981 0.0023829663340659 0.5025457847489473
0.0025223687467258 0.4975733456887479 0.5025319647608643
--
Title
3.84
1.0000000000000000 0.0000000000000000 0.0000000000000000
0.0000000000000000 1.0000000000000000 0.0000000000000000
0.0000000000000000 0.0000000000000000 1.0000000000000000
1 1 2
0.0026126405407573 0.0024006589240853 0.0027429160833867
0.5026006039214556 0.4975543718103462 0.5025677390656994
0.5024320195144374 0.0024395083059363 0.5025692798949066
0.0025647360233575 0.4976357954196818 0.5025590649560078
--
Title
3.85
```

```
1.0000000000000000 0.0000000000000000 0.0000000000000000

0.0000000000000000 1.0000000000000000 0.0000000000000000

0.0000000000000000 0.0000000000000000 1.0000000000000000

1 1 2

0.0026155975143523 0.0023865699879277 0.0031910181382412

0.5025389069877826 0.4976237622171453 0.5024846407068199

0.5025324748917634 0.0023896376839385 0.5023886834674245

0.0025230206061103 0.4975024454547277 0.5023746576875124

--

Title

3.86

1.0000000000000000 0.0000000000000000 0.0000000000000000

0.0000000000000000 1.0000000000000000 0.0000000000000000

0.0000000000000000 0.0000000000000000 1.0000000000000000

1 1 2

0.0025779413037294 0.0023422411895241 0.0033893156939426

0.5025953477873107 0.4975548081322764 0.5026056150177890

0.5024583675549223 0.0024123204040285 0.5022509082850218

0.0025783433540445 0.4975497534612827 0.5021931610032438

--

Title

3.87

1.0000000000000000 0.0000000000000000 0.0000000000000000

0.0000000000000000 1.0000000000000000 0.0000000000000000

0.0000000000000000 0.0000000000000000 1.0000000000000000

1 1 2

0.0026087773363121 0.0023010562689921 0.0033290890869272
```

```
0.5026104406570032 0.4975321247033073 0.5025946774250570

0.5024073535203309 0.0024451371344026 0.5022895216025576

0.0025834284863604 0.4975640687000938 0.5022257118854563

--

Title

3.88

1.0000000000000000 0.0000000000000000 0.0000000000000000

0.0000000000000000 1.0000000000000000 0.0000000000000000

0.0000000000000000 0.0000000000000000 1.0000000000000000

1 1 2

0.0026189552704468 0.0023280364763455 0.0039900663331869

0.5025963333407674 0.4975835287846180 0.5024772904566539

0.5024276441359415 0.0024339841018292 0.5020514961393880

0.0025670672528535 0.4975284917935635 0.5019201470707696

--

Title

3.89

1.0000000000000000 0.0000000000000000 0.0000000000000000

0.0000000000000000 1.0000000000000000 0.0000000000000000

0.0000000000000000 0.0000000000000000 1.0000000000000000

1 1 2

0.0027163119171203 0.0023015486318435 0.0035157390300608

0.5025570078004150 0.4975809171897236 0.5024438296793904

0.5024024993688605 0.0024370760351591 0.5022444383496529

0.0025341809136147 0.4975077074772848 0.5022349929408925

--
```

3 Results and Discussion

Data for the plot generated in this section – which depicts the oxygen vacancy formation energies of cubic unit cell systems including all studied B-sites, A-sites of multiple valence shell configurations (A = La, Sr, Ba, Ca, K, Na) and all simplifications detailed in the main sections of the article already made – can currently be aggregated via an RBDMS command adhering to the relational algebraic expression below:

$$\Pi_{at1,at2,calc\_quantity}$$
$$\sigma[(morph =' perovskite')]$$
$$\sigma \wedge [(at1 =' La') \vee (at1 =' Sr') \vee (at1 =' Ba')$$
$$\sigma \wedge [(at1 =' Ca') \vee (at1 =' K') \vee (at1 =' Na')]$$
$$\sigma \wedge [(num\_atoms = 5)]$$
$$\sigma \wedge [(correction =' none')]$$
$$\sigma \wedge [(mag =' none')]$$
$$\sigma \wedge [(orientation =' bulk')]$$
$$[Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$$

3.1 Sample Input Files

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies presented in the introductory section of the results section are shown below and labeled accordingly for both no vacancy ($AB\mathrm{O}_3$) and one vacancy ($AB\mathrm{O}_{3\text{-}\delta}$) unit cell structures. The same KPOINTS file is used in both cases. These particular sample input files were used in LaMnO_3 structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a lattice constant of 3.78 Å. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 1: INCAR No Vacancy

```
System = LaMnO3 Perovskite Trial No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = 5E-05

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100
```

Listing 2: INCAR 1 Vacancy

```
System = LaMnO3 Perovskite Trial 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = -0.005

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100
```

Listing 3: KPOINTS

```
6x6x6
0
Monkhorst
6 6 6
0 0 0
```

Listing 4: POSCAR No Vacancy

```
3.78 No Vacancy
3.78
1 0 0
0 1 0
0 0 1
1 1 3
direct
0.000000 0.000000 0.000000
0.500000 0.500000 0.500000
0.500000 0.500000 0.000000
0.500000 0.000000 0.500000
0.000000 0.500000 0.500000
```

Listing 5: POSCAR 1 Vacancy

```
3.78 1 Vacancy
3.78
1 0 0
0 1 0
0 0 1
1 1 2
direct
0.000010 0.000030 0.010400
0.500030 0.490080 0.500010
0.500100 0.000050 0.500020
0.010070 0.500190 0.500009
```

3.2 Plot Generation

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the plot presented in the Introductory subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded. Note that data with A = Ca exists, though this data is omitted from the query below and the plot generated from it for the purpose of preserving the visual clarity of the plot.

```python
# caption: Introductory Results Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

```

```
db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, Evac, Asites, Bsites = [], [], [], [], []

datapts_dict = {}

for row in db.execute(’’’
select c1.at1,c2.at2,ov.calc_quantity from Structure as s
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where s.morph=’perovskite’
and (c1.at1=’La’ or c1.at1=’Sr’ or c1.at1=’Ba’ or c1.at1=’K’ or c1.at1=’Na’)
and ov.num_atoms=5
and ov.correction=’none’
and ov.mag=’none’
and ov.orientation=’bulk’
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        Asites += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Sc": 0, "Ti": 1, "V": 2, "Cr": 3, "Mn": 4, "Fe": 5, "Co": 6, "Ni": 7, "Cu": 8}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

```

```
num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(Asites):
    matchlist = []
    for k in range(num_Bsites):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

x = [i for i in range( num_Bsites )]
ax = plt.gca()

print EBsiteMatch[’La’]
print EBsiteMatch[’Sr’]
print EBsiteMatch[’Ba’]
print EBsiteMatch[’K’]
print EBsiteMatch[’Na’]

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[’La’], ’bo-’, label = r’A = La’)
plt.plot(x, EBsiteMatch[’Sr’], ’ro-’, label = r’A = Sr’)
plt.plot(x, EBsiteMatch[’Ba’], ’go-’, label = r’A = Ba’)
plt.plot(x, EBsiteMatch[’K’], ’mo-’, label = r’A = K’)
plt.plot(x, EBsiteMatch[’Na’], ’co-’, label = r’A = Na’)

plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (-0.75, 6.25) )
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(bbox_to_anchor=(0.54, 0.37), prop={’size’:10})
plt.gcf().subplots_adjust(left=0.15)

for ext in [’png’, ’pdf’, ’eps’]:
    plt.savefig(’Asitevary.’ + ext)
plt.clf()
```

```
[6.074514929, 5.797935974, 5.35355224, 4.801317205, 4.265554955,
3.861766349, 3.42457409, 3.156455571, 3.068941827]
[3.601333605, 5.785254048, 5.151312452, 4.389435479, 3.502640925,
2.553923165, 1.849540578, 1.16803283, 0.738695201]

[3.690252674, 5.697070759, 5.075382454, 4.33889267, 3.471830727,

2.584238763, 1.886579332, 1.204811377, 0.657881926]

[1.214572067, 3.182810141, 3.979048969, 3.170132392, 2.229003588,

1.224145835, 0.291977356, -0.222482442, -0.531128386]

[1.00878734, 3.010745669, 3.871963721, 3.03949161, 2.063041824,

1.035203502, 0.140807927, -0.323244702, -0.644836512]
```

Table S1: Oxygen vacancy formation energies (eV) of non-magnetic cubic unit cell $\boldsymbol{AB}O_3$ systems without U correction.

A-site/B-site	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
La	6.07	5.80	5.35	4.80	4.27	3.86	3.42	3.16	3.07
Sr	3.60	5.79	5.15	4.39	3.50	2.55	1.85	1.17	0.74
Ba	3.69	5.70	5.08	4.34	3.47	2.58	1.89	1.20	0.66
Ca	3.35	5.62	4.98	4.23	3.35	2.40	1.72	1.06	0.68
K	1.21	3.18	3.98	3.17	2.23	1.22	0.29	-0.22	-0.53
Na	1.01	3.01	3.87	3.04	2.06	1.04	0.14	-0.32	-0.64

For the above calculations, formation energy values are stored in lists for a given A-site that are ordered by B-site, where B = Sc, Ti, V, Cr, Mn, Fe, Co, Ni, Cu.

3.3 Negative Values of Oxygen Vacancy Formation Energy

In several cases, especially those observed in the plot contained within the Introductory subsection of the Results and Discussion section, negative values of ΔE_{vac} were occasionally achieved, such as oxygen vacancy formation in $KCuO_3$. As mentioned previously, some of the systems represented in featured energetic trends are thermodynamically unstable, therefore their negative vacancy formation energies infer the greater stability of a generally more reduced system containing the same cations (e.g.: $KCuO_2$).[S11] However, the negativity or positivity of ΔE_{vac} values achieved in this study should not be used exclusively to infer the absolute thermodynamic stability of any of the systems characterized by these values under any set of experimental conditions, as the energetic trends derived are only intended to illustrate the relative redox reactivity of $AB\mathrm{O}_3$ systems sharing particular cation compositions. The systematic error of O_2 overbinding in the calculation of the DFT energy of the triplet state of the diatomic O_2 molecule ($\mathrm{E}_{O_2,DFT}$) causes calculated ΔE_{vac} values to be inaccurate by at least a fixed quantity that can be corrected via addition of a constant ($\mathrm{E}_{O_2,corr}$). Previous studies have estimated this correction to be $\mathrm{E}_{O_2,corr}$ = -1.36 eV per O_2 molecule, using the following equation to correct the value of the O_2 energy used in the expression for ΔE_{vac} (E_{O_2}):[S12]

$$E_{O_2} = E_{O_2,DFT} - E_{O_2,corr} \tag{15}$$

The DFT energy of O_2 is calculated to be -9.861 eV in a single k-point calculation using a O=O bond distance of 1.21 Å,[S13] a simulation box size with an energetically converged edge length of 12 Å, and the following INCAR and POSCAR input files:

Listing 6: INCAR O_2

```
System = Diatomic O2
ISTART = 0 ; ICHARG = 2
ENCUT = 400

ISMEAR = 0
SIGMA = 0.01
ISPIN = 2

IBRION = 2
EDIFF = 5E-05
EDIFFG = -0.03

NSW = 25
MAXMIX = 80
```

Listing 7: POSCAR O_2

```
12
1
12 0 0
0 12 0
0 0 12
2
selective
cartesian
0 0 0 F F T
0 0 1.21 F F T
```

Application of the calculated DFT energy of the O_2 molecule in its triply degenerate ground state and the known correction to the DFT energy to the expression for finding ΔE_{vac} of a particular system such as $KCuO_3$ (ΔE_{vac} = -0.531 eV for the corrected quantity) reveals the uncorrected oxygen vacancy formation energy for that system:

$$\Delta E_{vac}(\text{eV}) = E_{ABO_{3-\delta}} + \frac{1}{2}E_{O_2} - E_{ABO_3} \tag{16}$$

$$\Delta E_{vac}(\text{eV}) = E_{ABO_{3-\delta}} + \frac{1}{2}(E_{O_2,DFT} - E_{O_2,corr}) - E_{ABO_3} \tag{17}$$

$$-0.531\text{eV} = E_{ABO_{3-\delta}} - E_{ABO_3} + \frac{1}{2}(-9.861 - (-1.36))\text{eV} \tag{18}$$

$$-1.211\text{eV} = E_{ABO_{3-\delta}} - E_{ABO_3} - (-9.861)\text{eV} \tag{19}$$

An estimate for the actual quantity by which E_{O_2} should be corrected in accordance with experiment is revealed by equating the DFT energy of the O_2 molecule to the bond dissociation energy of O_2 ($E_{O=O}$) via the DFT energy of a single oxygen atom (E_O). E_O is calculated to be -1.895 eV using a single *k*-point calculation employing INCAR parameters comparable to those used in the calculation of diatomic O_2, as well as a simulation box size of 11 Å featuring broken box symmetry (see below):[S14]

Listing 8: POSCAR O atom

```
O atom
1
10.9 0.00 0.00
0.00 11.0 0.00
0.00 0.00 11.1
1
cart
0 0 0
```

Application of this result to the expression for E$_{O=O}$ reveals a DFT estimate of the bond dissociation energy of O_2 of 6.071 eV, as is shown below:

$$E_{O=O} = 2E_O - E_{O_2} = (2 \times -1.895) - (-9.861)eV = 6.071eV \tag{20}$$

Given an experimentally resolved E$_{O=O}$ value of 5.17 eV[S14] and assuming there is no error in the calculation of the DFT energy of the oxygen atom, a correction of 5.17 - 6.07 eV = -0.90 eV – rather than -1.36 eV – per O_2 molecule should be instituted in accordance with the experimentally calculated oxygen molecule bond dissociation energy. However, the value of E$_O$ can be either overestimated or underestimated by DFT. Nevertheless, both possible values for the correction of the DFT energy of O_2 (E$_{O_2,corr}$), namely -0.90 eV and -1.36 eV, lead to systematic corrections of ΔE_{vac} values of -0.45 eV and -0.68 eV, respectively. These two corrective values serve to illustrate the magnitude of how negative a value of ΔE_{vac} should be to conclude whether a particular ABO$_3$ system is thermodynamically unstable at its current level of oxidation or whether errors in the calculation of the DFT total energy of E$_{O_2}$ prevent such a conclusion from being drawn.

3.4 Vacancy Concentration section

Data for the A = La plot generated in this section, which depicts the oxygen vacancy formation energies of supercell systems including all studied B-sites and all simplifications detailed in the main sections of the article already made (except for those involving vacancy concentration), can currently be aggregated via an RBDMS command adhering to the relational algebraic expression below. In order to produce the corresponding A = Sr plot, simply substitute the value of the attribute at1 with 'Sr':

$$\Pi_{num\_atoms,at1,at2,calc\_quantity}$$

$$\sigma[(struct =' Pm - 3m')]$$

$$\sigma \wedge [(morph =' perovskite')]$$

$$\sigma \wedge [(at1 =' La')]$$

$$\sigma \wedge [(num\_atoms = 5) \vee (num\_atoms = 10) \vee (num\_atoms = 20) \vee (num\_atoms = 40)]$$

$$\sigma \wedge [(correction =' none')]$$

$$\sigma \wedge [(mag =' none')]$$

$$\sigma \wedge [(orientation =' bulk')]$$

$$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$$

3.4.1 Sample Input Files

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies presented in the Vacancy Concentration subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy ($AB\text{O}_3$) and one vacancy ($AB\text{O}_{3\text{-}\delta}$) structures. The same KPOINTS file is used in both cases. These particular sample input files were used in LaMnO_3 structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a lattice constant of 3.81 Å for a $2 \times 2 \times 2$ supercell. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 9: INCAR No Vacancy

```
System = LaMnO3 Perovskite Trial No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4
```

Listing 10: INCAR 1 Vacancy

```
System = LaMnO3 Perovskite Trial 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4
```

Listing 11: KPOINTS

```
3x3x3
0
Monkhorst-Pack
3 3 3
0 0 0
```

```
# caption: POSCAR No Vacancy
3.81 No Vacancy
3.81
2 0 0
0 2 0
```

```
0 0 2
8 8 24
Direct
0.0000        0.0000        0.0000
0.5000        0.0000        0.0000
0.0000        0.5000        0.0000
0.0000        0.0000        0.5000
0.5000        0.5000        0.0000
0.5000        0.0000        0.5000
0.0000        0.5000        0.5000
0.5000        0.5000        0.5000

0.2500        0.2500        0.2500
0.7500        0.2500        0.2500
0.2500        0.7500        0.2500
0.2500        0.2500        0.7500
0.7500        0.7500        0.2500
0.7500        0.2500        0.7500
0.2500        0.7500        0.7500
0.7500        0.7500        0.7500

0.2500        0.2500        0.0000
0.7500        0.2500        0.0000
0.2500        0.7500        0.0000
0.7500        0.7500        0.0000
0.2500        0.2500        0.5000
0.7500        0.2500        0.5000
0.2500        0.7500        0.5000
0.7500        0.7500        0.5000

0.2500        0.0000        0.2500
0.7500        0.0000        0.2500
0.2500        0.0000        0.7500
0.7500        0.0000        0.7500
0.2500        0.5000        0.2500
0.7500        0.5000        0.2500
0.2500        0.5000        0.7500
0.7500        0.5000        0.7500

0.0000        0.2500        0.2500
0.0000        0.7500        0.2500
```

```
0.0000  0.2500  0.7500
0.0000        0.7500        0.7500
0.5000        0.2500        0.2500
0.5000        0.7500        0.2500
0.5000        0.2500        0.7500
0.5000        0.7500        0.7500
```

```
#caption: POSCAR 1 Vacancy
3.81 1 Vacancy
3.81
2 0 0
0 2 0
0 0 2
8  8  23
Direct
  0.0028265114275790  0.0104233554733421  0.0110085339920051
  0.5054026762799446  0.0128319139049892  0.0125698078369502
  0.0027174111864015  0.4979528554550048  0.0101889840287193
  0.0058216778666534  0.0092510677397705  0.4988258231635488
  0.5053415320196020  0.4957648854738518  0.0122080696561636
  0.5035105333722354  0.0112895718684222  0.4974365877465119
  0.0056480609006659  0.4996272850540678  0.4992037925592512
  0.5033671858950036  0.4971906789691610  0.4977652412455728
  0.2496012608209845  0.2548884893194709  0.2539424658214479
  0.7590977788878327  0.2543818038975971  0.2573433925797472
  0.2612674221463638  0.7541267715503881  0.2528335303047630
  0.2605960069654803  0.2539525123533591  0.7553206628038575
  0.7476216085948875  0.7543189750700229  0.2537336527421551
  0.7481084949614215  0.2547919244655782  0.7529453382997588
  0.2328064405551903  0.7549056003155086  0.7559718482668955
  0.7757649624500126  0.7545467308968442  0.7551065063893403
  0.2538035657969821  0.2533550439674284  0.0047924796350295
  0.7547299270043952  0.2543439406014752  0.0051523546768602
  0.2688458015453682  0.7556597358661657 -0.0007145027694399
  0.7412300164036213  0.7531488468062254  0.9998022706075843
  0.2521151371124858  0.2555906859356042  0.5056911400120223
  0.7560381247436906  0.2533486878589326  0.5058418025048257
  0.2689937315365204  0.7531854565593730  0.5115210250885481
  0.7389765317829116  0.7541878250421191  0.5118274207152592
  0.2526919579233243  0.0039141098203333  0.2559667800537996
  0.7560759278520972  0.0036258549262379  0.2555151804419796
  0.2693045998929147  0.9984526012168904  0.7543706756711538
  0.7378418342147335  0.9974997697635374  0.7561548703672261
  0.2539638184715716  0.5047885981404534  0.2536841063600958
  0.7547624024171730  0.5044361443189462  0.2565104338727288
  0.2670293817306553  0.5102930233888991  0.7568170773197279
  0.7419730886303225  0.5095255163600100  0.7550847132297088
  0.0042662072728371  0.2550338476356143  0.2548201592885823
  0.0044941233912626  0.7534542301416585  0.2552825090243301
  0.0039630421977291  0.2519850884924801  0.7557884192781279
  0.0043154573029069  0.7564365923199758  0.7545806935259097
  0.5042599365881950  0.2538602066100866  0.2561639679948878
  0.5044506169012053  0.7547056894222466  0.2545825335172249
  0.5041945649567910  0.2553094149979843  0.7542665671471742
```

3.4.2 Plot Generation

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the plots presented in the Vacancy Concentration subsection of the Results and Discussion section, then generate the plots themselves. The code written below was used to generate the plot involving $LaBO_3$ sytems, while the $SrBO_3$ plot was generated via a highly similar code. The actual numerical values of the oxygen vacancy formation energy used to generate the $LaBO_3$ plot are written at the conclusion of the code as tabulated, rounded results achieved with code output. They are followed by matching $SrBO_3$ tabulated values, the plot of which can be reproduced by changing the value of the variable A-site to ’Sr’ and the query line ”and c1.at1=’La’” to ”and c1.at1=’Sr’”.

```python
# caption: Vacancy Concentration Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Asite = ’La’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, Evac, atomnum, Bsites = [], [], [], [], []
```

```
datapts_dict = {}

for row in db.execute(’’’
select ov.num_atoms,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and (ov.num_atoms=5 or ov.num_atoms=10 or ov.num_atoms=20 or ov.num_atoms=40)
and ov.correction=’none’
and ov.mag=’none’
and ov.orientation=’bulk’
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        atomnum += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Sc": 0, "Ti": 1, "V": 2, "Cr": 3, "Mn": 4, "Fe": 5, "Co": 6, "Ni": 7, "Cu": 8}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
```

```
EBsiteMatch = {}

for i,j in enumerate(atomnum):
    matchlist = []
    for k in range(num_Bsites):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

x = [i for i in range( num_Bsites )]
ax = plt.gca()

print EBsiteMatch[5]
print EBsiteMatch[10]
print EBsiteMatch[20]
print EBsiteMatch[40]

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[5], 'bo-', label = r'1 $\times$ 1 $\times$ 1')
plt.plot(x, EBsiteMatch[10], 'ro-', label = r'1 $\times$ 1 $\times$ 2')
plt.plot(x, EBsiteMatch[20], 'go-', label = r'2 $\times$ 2 $\times$ 1')
plt.plot(x, EBsiteMatch[40], 'ko-', label = r'2 $\times$ 2 $\times$ 2')
plt.xlabel( 'B-site' )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (1.5, 6.5) )
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'upper right', prop={'size':10})
plt.text(0.08, 0.85, 'A = {0}'.format(Asite),
         style = 'oblique', color = 'white',
         fontsize = 16,
         transform = ax.transAxes,
         bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})

plt.gcf().subplots_adjust(left=0.15)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('ConcDep_' + Asite + '.' + ext)
plt.clf()
```

```
[6.074514929, 5.797935974, 5.35355224, 4.801317205, 4.265554955,
3.861766349, 3.42457409, 3.156455571, 3.068941827]

[5.886123299, 5.781887748, 5.634927624, 4.972912383, 4.334676078,

3.776676998, 3.203190457, 2.626749499, 1.885192589]

[5.322880582, 6.065841428, 5.599833737, 4.938266108, 4.476644714,

3.838638166, 3.029766446, 2.493002509, 1.627650485]

[6.080706073, 6.22995336, 5.398069469, 4.906962468, 4.677997905,

3.967727678, 3.000129742, 2.624675621, 1.692585206]
```

Table S2: Oxygen vacancy formation energies (eV) of non-magnetic cubic supercell La*B*O$_3$ systems without *U* correction.

Supercell	**Sc**	**Ti**	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**	**Cu**
$1 \times 1 \times 1$	6.07	5.80	5.35	4.80	4.27	3.86	3.42	3.16	3.07
$1 \times 1 \times 2$	5.89	5.78	5.63	4.97	4.33	3.78	3.20	2.63	1.89
$2 \times 2 \times 1$	5.32	6.07	5.60	4.94	4.48	3.84	3.03	2.49	1.63
$2 \times 2 \times 2$	6.08	6.23	5.40	4.91	4.68	3.97	3.00	2.62	1.69

Table S3: Oxygen vacancy formation energies (eV) of non-magnetic cubic supercell Sr*B*O$_3$ systems without *U* correction.

Supercell	**Sc**	**Ti**	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**	**Cu**
$1 \times 1 \times 1$	3.60	5.79	5.15	4.39	3.50	2.55	1.85	1.17	0.74
$1 \times 1 \times 2$	0.19	5.77	5.30	4.50	3.41	2.25	1.22	0.39	0.36
$2 \times 2 \times 1$	-0.21	5.73	5.26	4.37	3.08	1.92	0.76	0.05	-0.04
$2 \times 2 \times 2$	-0.17	5.56	5.26	4.29	2.92	1.78	0.52	0.15	-0.25

For the above calculations, formation energy values are stored in lists for a given *A*-site that are ordered by *B*-site, where B = Sc, Ti, V, Cr, Mn, Fe, Co, Ni, Cu.

3.5 Crystal Structure section

Data for the rhombohedral vs. cubic plot generated in this section, which depicts the oxygen vacancy formation energies of unit cell systems including all studied *B*-sites, La and Sr *A*-sites, and all simplifications detailed in the main sections of the article already made (except for those involving crystallographic symmetry), can currently be aggregated via an RBDMS command adhering to the relational algebraic expression below:

$$\Pi_{struct,at1,at2,calc\_quantity}$$

$$\sigma[(struct =' Pm - 3m') \vee (struct =' R - 3c')]$$

$$\sigma \wedge [(morph =' perovskite')]$$

$$\sigma \wedge [(at1 =' La') \vee (at1 =' Sr')]$$

$$\sigma \wedge [(num\_atoms = 10)]$$

$$\sigma \wedge [(correction =' none')]$$

$$\sigma \wedge [(mag =' none')]$$

$$\sigma \wedge [(orientation =' bulk')]$$

$$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$$

The corresponding orthorhombic vs. cubic plot generated in this section for oxygen vacancies induced at the $8d$ symmetry site can be aggregated via commands that correspond to the expression below (generation of a plot considering oxygen vacancies at the $4c$ site simply requires substituting the orientation tag 'bulk_8d' with 'bulk_4c'):

$$\Pi_{struct,at1,at2,calc\_quantity}$$

$$\sigma[(struct =' Pm - 3m') \vee (struct =' Pbnm')]$$

$$\sigma \wedge [(morph =' perovskite')]$$

$$\sigma \wedge [(at1 =' La') \vee (at1 =' Sr')]$$

$$\sigma \wedge [(num\_atoms = 20)]$$

$$\sigma \wedge [(correction =' none')]$$

$$\sigma \wedge [(mag =' none')]$$

$$\sigma \wedge [(orientation =' bulk') \vee (orientation =' bulk\_8d')]$$

$$\sigma \wedge [(at2 \neq' Sc')]$$

$$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$$

3.5.1 Sample Input Files

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies presented in the rhombohedral plot within the Crystal Structure subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy ($AB\mathrm{O}_3$) and one vacancy ($AB\mathrm{O}_{3\text{-}\delta}$) structures. The same KPOINTS file is used in both cases. These particular sample input files were used in LaCoO_3 structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a rhombohedral lattice constant of 5.40 Å. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 12: INCAR No Vacancy

```
System = LaCoO3 Rhombohedral Perovskite Trial, No Vacancy
 ISTART = 0 ; ICHARG = 2
 ENCUT = 550.000000
 SIGMA = 0.010000
 EDIFF = 5.00e-06
 EDIFFG = -1.00e-02
 ISYM = 1
 SYMPREC = 1E-5
 MAXMIX = -60
 ISMEAR = 0
 NELMIN = 8
 NELM = 80
 NSW = 100
 IBRION = 1
 ISIF = 4
```

Listing 13: INCAR 1 Vacancy

```
System = LaCoO3 Rhombohedral Perovskite Trial, 1 Vacancy
 ISTART = 0 ; ICHARG = 2
 ENCUT = 550.000000
 SIGMA = 0.010000
 EDIFF = 5.00e-06
 EDIFFG = -1.00e-02
 ISYM = 0
 SYMPREC = 1E-5
 MAXMIX = -60
 ISMEAR = 0
 NELMIN = 8
 NELM = 80
 NSW = 100
 IBRION = 1
 ISIF = 4
```

Listing 14: KPOINTS

```
6x6x6
0
Monkhorst-Pack
6 6 6
0 0 0
```

Listing 15: POSCAR No Vacancy

```
5.40, Rhombohedral Perovskite Trial, No Vacancy
5.40
0.506018  -0.292150  0.811538
0.000000   0.584300  0.811538
-0.506018 -0.292150  0.811538
2 2 6
direct
0.250000 0.250000 0.250000
0.750000 0.750000 0.750000

0.000000 0.000000 0.000000
0.500000 0.500000 0.500000

0.19820  0.30180  0.75000
0.75000  0.19820  0.30180
0.30180  0.75000  0.19820

-0.19820 0.69820  0.25000
0.25000  -0.19820  0.69820
0.69820  0.25000  -0.19820
```

Listing 16: POSCAR 1 Vacancy

```
5.40, Rhombohedral Perovskite Trial, 1 Vacancy
5.40
0.506018  -0.292150  0.811538
0.000000   0.584300  0.811538
-0.506018 -0.292150  0.811538
2 2 5
direct
0.280003 0.270020 0.240007
0.750010 0.750004 0.760050

0.000060 0.010030 0.000005
0.510007 0.470009 0.480008

0.178201  0.291806  0.780007
0.750002  0.198205  0.301809
0.301803  0.760004  0.198208

-0.188203 0.698209  0.250006
0.688205  0.280002  -0.228204
```

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies presented in the orthorhombic plot within the Crystal Structure subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy (ABO_3) and one vacancy ($ABO_{3\text{-}\delta}$ oxygen vacancy at 8d oxygen site) structures. The same KPOINTS file is used in both cases. These particular sample input files were used in $LaMnO_3$ structural relaxation calculations run at a constant cell volume

(ISIF = 4) corresponding to orthorhombic lattice constants of 5.42, 5.43, and 7.66 Å. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 17: INCAR No Vacancy

```
System = LaMnO3 Orthorhombic Perovskite Trial, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = 5E-05

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100
```

Listing 18: INCAR 1 Vacancy

```
System = LaMnO3 Orthorhombic Perovskite Trial, 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100
```

Listing 19: KPOINTS

```
6x6x6
0
Monkhorst-Pack
6 6 6
0 0 0
```

Listing 20: POSCAR No Vacancy

```
5.42, Orthorhombic Unit Cell, No Vacancy
5.42
1.0000 0.0000 0.0000
0.0000 1.0018 0.0000
0.0000 0.0000 1.4167
4 4 12
direct
-0.00614  0.02840  0.25000
 0.49386  0.47160  0.75000
 0.00614 -0.02840  0.75000
 0.50614  0.52840  0.25000
 0.00000  0.50000  0.00000
 0.50000  0.00000  0.00000
 0.50000  0.00000  0.50000
 0.00000  0.50000  0.50000
 0.07310  0.48750  0.25000
 0.57310  0.01250  0.75000
-0.07310 -0.48750  0.75000
 0.42690  0.98750  0.25000
-0.28090  0.28150  0.03940
 0.21910  0.21850 -0.03940
 0.28090 -0.28150  0.53940
 0.78090  0.78150  0.46060
 0.28090 -0.28150 -0.03940
 0.78090  0.78150  0.03940
-0.28090  0.28150  0.46060
 0.21910  0.21850  0.53940
```

Listing 21: POSCAR 1 Vacancy

```
5.42, Orthorhombic Unit        Cell, 1 Vacancy
5.42
     1.0021521119499524    0.0000000000000000    0.0000000000000000
     0.0000000000000000    0.9998347709054796    0.0000000000000000
     0.0000000000000000    0.0000000000000000    1.4164362707305829
   4   4  11
Direct
  0.9953540481232052  0.0264783696511581  0.2500000000000000
  0.4953540481232058  0.4735216303488415  0.7500000000000000
  0.0046459518767942  0.9735216303488421  0.7500000000000000
  0.5046459518767948  0.5264783696511579  0.2500000000000000
  0.0000000000000000  0.5000000000000000  0.0000000000000000
  0.5000000000000000  0.0000000000000000  0.0000000000000000
  0.5000000000000000  0.0000000000000000  0.5000000000000000
  0.0000000000000000  0.5000000000000000  0.5000000000000000
  0.0643910210574352  0.4934293146910956  0.2500000000000000
  0.5643910210574350  0.0065706853089043  0.7500000000000000
  0.9356089789425650  0.5065706853089043  0.7500000000000000
  0.4356089789425650  0.9934293146910957  0.2500000000000000
  0.7211697898924809  0.2785113175574914  0.0341409499337401
  0.2211697898924811  0.2214886824425086  0.9658590500662597
  0.2788302101075192  0.7214886824425087  0.5341409499337403
  0.7788302101075191  0.7785113175574913  0.4658590500662599
  0.2788302101075192  0.7214886824425087  0.9658590500662597
  0.7788302101075191  0.7785113175574913  0.0341409499337401
  0.7211697898924809  0.2785113175574914  0.4658590500662599
```

3.5.2 Plot Generation

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the plot of rhombohedral and cubic energetic trends presented in the Crystal Structure subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```
# caption: Crystal Structure (Rhombohedral) Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, Evac, gsym, Asites, Bsites = [], [], [], [], [], []

datapts_dict = {}

for row in db.execute(’’’
select sy.sym,c1.at1,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where (sy.sym=’Pm-3m’ or sy.sym=’R-3c’)
and s.morph=’perovskite’
and (c1.at1=’La’ or c1.at1=’Sr’)
and ov.num_atoms=10
and ov.correction=’none’
and ov.mag=’none’
and ov.orientation = ’bulk’
;’’’):
    datapts_list = []
    a,b,c,d = row
    datapts_list.append( (a,b,c,d) )

    if a not in label_1:
        gsym += [a,b]

    if c not in label_2:
        Bsites += [c]

    label_1 += [a,b]
    label_2 += [c]
    Evac += [d]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Sc": 0, "Ti": 1, "V": 2, "Cr": 3, "Mn": 4, "Fe": 5, "Co": 6, "Ni": 7, "Cu": 8}

for i, j, k, l in datapts_dict:
    SortElist[i,j].append( (k,l) )
    SortElist[i,j].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    for k in range(num_Bsites):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

x = [i for i in range( num_Bsites )]
ax = plt.gca()

print EBsiteMatch[’R-3c’,’La’]
print EBsiteMatch[’R-3c’,’Sr’]
print EBsiteMatch[’Pm-3m’,’La’]
print EBsiteMatch[’Pm-3m’,’Sr’]

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[’R-3c’,’La’], ’bo-’, label = r’R$\overline{3}$c, A = La’)
plt.plot(x, EBsiteMatch[’R-3c’,’Sr’], ’ro-’, label = r’R$\overline{3}$c, A = Sr’)
plt.plot(x, EBsiteMatch[’Pm-3m’,’La’], ’go-’, label = r’Pm$\overline{3}$m, A = La’)
plt.plot(x, EBsiteMatch[’Pm-3m’,’Sr’], ’ko-’, label = r’Pm$\overline{3}$m, A = Sr’)
plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (0, 7) )
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(loc = ’upper right’, prop={’size’:7})
plt.gcf().subplots_adjust(left=0.15)

for ext in [’png’, ’pdf’, ’eps’]:
    plt.savefig(’AltGeom_R3c.’ + ext)
plt.clf()
```

```
[6.87561945, 6.23142688, 5.680036337, 5.175607852, 4.92238919,
4.499011868, 3.926056274, 3.321207363, 2.533913304]
[0.752951441, 6.217992865, 5.754238174, 4.737194954, 3.684990902,
2.777595673, 1.939853134, 1.120294422, 0.703553637]
[5.886123299, 5.781887748, 5.634927624, 4.972912383, 4.334676078,
3.776676998, 3.203190457, 2.626749499, 1.885192589]
[0.18857621, 5.772711096, 5.302436653, 4.495060139, 3.410589387,
2.249253612, 1.21904744, 0.393146856, 0.35967468]
```

Table S4: Oxygen vacancy formation energies (eV) of non-magnetic rhombohedral unit cell and cubic $1 \times 1 \times 2$ supercell $\boldsymbol{AB}$O$_3$ systems without U correction.

A-site, Symmetry	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
La, $R\bar{3}c$	6.87	6.23	5.68	5.18	4.92	4.50	3.93	3.32	2.53
Sr, $R\bar{3}c$	0.75	6.22	5.75	4.74	3.68	2.78	1.94	1.12	0.70
La, $Pm\bar{3}m$	5.89	5.78	5.63	4.97	4.33	3.78	3.20	2.63	1.89
Sr, $Pm\bar{3}m$	0.19	5.77	5.30	4.49	3.41	2.25	1.22	0.39	0.36

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the plot of orthorhombic (oxygen vacancy at 8d site) and cubic energetic trends presented in the Crystal Structure subsection of the Results and Discussion section, then generate the plot itself. In order to achieve the orthorhombic plot corresponding to 4c oxygen vacancies, change the Osite variable from ’8d’ to ’4c’ and the query line

”(ov.orientation=’bulk_8d’ or ov.orientation=’bulk’)” to ”and (ov.orientation=’bulk_4c’ or ov.orientation=’bulk’)”. Despite the presence of B = Sc data for the cubic trend, formation energies with B = Sc are omitted (see query line ”(c2.at2 != ’Sc’)”) due to the absence of corresponding orthorhombic data. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```python
# caption: Crystal Structure (Orthorhombic) Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Osite=’8d’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, Evac, gsym, Asites, Bsites = [], [], [], [], [], []

datapts_dict = {}

for row in db.execute(’’’
select sy.sym,c1.at1,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where (sy.sym=’Pm-3m’ or sy.sym=’Pbnm’)
and s.morph=’perovskite’
and (c1.at1=’La’ or c1.at1=’Sr’)
and ov.num_atoms=20
and ov.correction=’none’
and ov.mag=’none’
and (ov.orientation=’bulk_8d’ or ov.orientation=’bulk’)
and (c2.at2 != ’Sc’)
;’’’):
    datapts_list = []
    a,b,c,d = row
    datapts_list.append( (a,b,c,d) )

    if a not in label_1:
        gsym += [a,b]

    if c not in label_2:
        Bsites += [c]

    label_1 += [a,b]
    label_2 += [c]
    Evac += [d]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k, l in datapts_dict:
    SortElist[i,j].append( (k,l) )
    SortElist[i,j].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate( SortElist.keys() ):
    matchlist = []
    for k in range(num_Bsites):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[’Pbnm’,’La’]
print EBsiteMatch[’Pbnm’,’Sr’]
print EBsiteMatch[’Pm-3m’,’La’]
print EBsiteMatch[’Pm-3m’,’Sr’]

x = [i for i in range( num_Bsites )]
ax = plt.gca()

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[’Pbnm’,’La’], ’bo-’, label = r’Pbnm, A = La’)
plt.plot(x, EBsiteMatch[’Pbnm’,’Sr’], ’ro-’, label = r’Pbnm, A = Sr’)
plt.plot(x, EBsiteMatch[’Pm-3m’,’La’], ’go-’, label = r’Pm$\overline{3}$m, A = La’)
plt.plot(x, EBsiteMatch['Pm-3m','Sr'], 'ko-', label = r'Pm$\overline{3}$m, A = Sr')
plt.xlabel( 'B-site' )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (-0.5, 7) )
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'lower left', prop={'size':8.5})
plt.gcf().subplots_adjust(left=0.15)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('AltGeom_Pbnm_' + Osite + '.' + ext)
plt.clf()
```

```
[6.657303133, 6.032310759, 5.534656372, 5.344250874, 4.864247743, 3.92129643, 3.38289480

[5.956639453, 5.555568128, 4.48016203, 3.285420546, 2.026542302, 1.323855466, 0.54410631

[6.065841428, 5.599833737, 4.938266108, 4.476644714, 3.838638166, 3.029766446, 2.4930025

[5.725392622, 5.264959878, 4.370702766, 3.081100537, 1.921811777, 0.763258974, 0.0466553
```

Table S5: Oxygen vacancy formation energies (eV) of non-magnetic orthorhombic unit cell (8d site oxygen vacancy) and cubic $2 \times 2 \times 1$ supercell ABO_3 systems without U correction.

A-site, Symmetry	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
La, *Pbnm*	6.66	6.03	5.53	5.34	4.86	3.92	3.38	2.21
Sr, *Pbnm*	5.96	5.56	4.48	3.29	2.03	1.32	0.54	0.12
La, $Pm\overline{3}m$	6.07	5.60	4.94	4.48	3.84	3.03	2.49	1.63
Sr, $Pm\overline{3}m$	5.73	5.26	4.37	3.08	1.92	0.76	0.05	-0.04

Table S6: Oxygen vacancy formation energies (eV) of non-magnetic orthorhombic unit cell (4c site oxygen vacancy) and cubic $2 \times 2 \times 1$ supercell ABO_3 systems without U correction.

A-site, Symmetry	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
La, *Pbnm*	6.60	6.12	5.51	5.33	4.86	3.90	3.41	2.08
Sr, *Pbnm*	6.16	5.68	4.77	3.69	2.61	1.55	0.98	0.33
La, $Pm\overline{3}m$	6.07	5.60	4.94	4.48	3.84	3.03	2.49	1.63
Sr, $Pm\overline{3}m$	5.73	5.26	4.37	3.08	1.92	0.76	0.05	-0.04

3.5.3 Evaluation of Alternate Orthorhombic Oxygen Vacancy Sites

In order to verify whether imposing orthorhombic crystal structure on $LaBO_3$ and $SrBO_3$ systems has an effect on the ΔE_{vac} ordering of the energetic trends containing them, the possible presence of oxygen vacancies at either of two non-equivalent oxygen sites in the 20 atom *Pbnm* unit cell had to be considered. Input POSCAR files depicting the atomic sites, namely the $8d$ and $4c$ sites,[S15] at which oxygen vacancies were induced are listed below. Visualizations that assist in distinguishing between the formation of oxygen vacancies at both sites were completed using the Visualization for Electronic and Structural Analysis (VESTA) software package and are shown below.[S16]

Listing 22: POSCAR (Orthorhombic) for $8d$ Oxygen Vacancy

```
5.42
   5.42000000000000
     1.0061427167511148    0.0006170676645362    0.0020125332336380
     0.0006090824664552    0.9985690275017297    0.0013879512646019
     0.0028530810594471    0.0019704721520323    1.4126156050045480
   La   Mn   O
   4   4  11
Direct
  0.9917317387768867  0.0205208743772843  0.2433228805689292
  0.5043630254205310  0.4831365430237454  0.7557731311458977
 -0.0014774834849442  0.9728990518202730  0.7483887405796068
  0.5062172310636456  0.5270284760479345  0.2529387223243661
  0.0041010785000933  0.4949424366596999 -0.0011647454614288
  0.4945210244924635  0.0045065263057128  0.0003147488250800
  0.4973975935204362  0.0075313971777559  0.4995075068359662
  0.0024904985751258  0.4978238320064411  0.4993764228511517
  0.0646905909389330  0.4868692683541715  0.2511629418784200
  0.5516630966846233  0.0111763651719799  0.7495154044506090
  0.9540482384950654  0.4860456351055150  0.7482658633636579
  0.4339143337331991  1.0077395573671484  0.2528663913760595
  0.7169385099529016  0.2831720848879423  0.0331352623943422
  0.2186568485040386  0.2171711148483619  0.9618264184820277
  0.2701815552911276  0.7309674442891022  0.5322314100111262
  0.7780927968072772  0.7764013009513805  0.4682364250193704
  0.2802966062285159  0.7195170142213120  0.9694481024633256
  0.7804012315193639  0.7806018156556616  0.0272208542140661
  0.7306016950882388  0.2704605792860760  0.4734925687436799
```

Listing 23: POSCAR (Orthorhombic) for 4*c* Oxygen Vacancy

```
5.42
   5.42000000000000
     1.0011697746514023   -0.0002774134987534   -0.0000004230490151
    -0.0002756962865619    0.9998252178381174   -0.0000086650868859
    -0.0000011943503772   -0.0000131205046009    1.4178397185675229
   La   Mn   O
   4   4  11
Direct
  0.9934276032738927  0.0239850617910080  0.2500015109497449
  0.4951990558727700  0.4757270044234014  0.7499958101541282
  0.0053448293358048  0.9769062832985971  0.7499902107274901
  0.4979553013399876  0.5230973648964663  0.2500045030462645
 -0.0035689660457601  0.5015246219439983  0.0034445216148664
  0.5024718819283209 -0.0005851939551223  0.0044992050805053
  0.5024780787413782 -0.0005831140186897  0.4955018084016123
 -0.0035705600715401  0.5015234669356647  0.4965556220122812
  0.5667813998464192  0.0110297501476764  0.7500042108378107
  0.9367097055589946  0.5088905395727972  0.7499997922961417
  0.4339572408596958  0.9945979143719714  0.2500031142329704
  0.7272373513203784  0.2768374071530404  0.0384063501194849
  0.2232401961095022  0.2252178315929756  0.9758954489421976
  0.2769772992847755  0.7211443526799377  0.5235428261784424
  0.7767944632137155  0.7720312241134276  0.4632606091613164
  0.2769562105322219  0.7211632241419230  0.9764484647439526
  0.7767696429091596  0.7720023746995677  0.0367463626956836
  0.7272247773712063  0.2768562628528823  0.4616043423746826
  0.2232234675616440  0.2252043086673784  0.5240952864304261
```

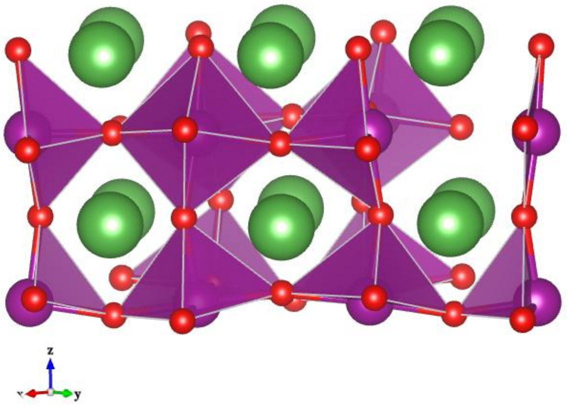

Figure S2: Visualization of a 20 atom $\sqrt{2} \times \sqrt{2} \times 2$ *Pbnm* unit cell partially periodically repeated along the (110) plane to depict the presence of an oxygen vacancy at the 4*c* site. Including periodically repeated atoms and the single oxygen vacancy, this visualization contains 59 atoms.

The ΔE_{vac} values of $LaBO_3$ and $SrBO_3$ systems corresponding to oxygen vacancies induced at 8*d* and 4*c* sites are shown in tables located in the previous section, while plots directly comparing them to analogous 20 atom 2 × 2 × 1 cubic systems are shown below.

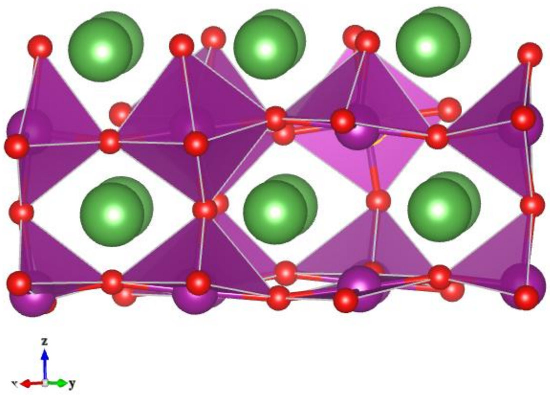

Figure S3: Visualization of a 20 atom $\sqrt{2} \times \sqrt{2} \times 2$ *Pbnm* unit cell partially periodically repeated along the (110) plane to depict the presence of an oxygen vacancy at the 8*d* site. Including periodically repeated atoms and the single oxygen vacancy, this visualization contains 59 atoms.

Though differences in ΔE_{vac} values between matching cases with oxygen vacancies at different sites can be significant (e.g.: *Pbnm*, A = Sr energetic trend lines compared between induced oxygen vacancies at 4*c* and 8*d* sites) and on the order of $\Delta(\Delta E_{vac}) = 0.6$ eV, these differences in oxygen vacancy formation energy are fairly uniform and do not affect conclusions drawn in the Results and Discussion section concerning ΔE_{vac} ordering or the relative favorability of vacancy formation in orthorhombic or cubic systems.

3.5.4 Evaluation of Alternate Rhombohedral Coordinate Sets

The rhombohedral angle and set of atomic coordinates provided as input values (within POSCAR) to all studied perovskites in the Crystal Structure subsection of the Results and Discussion section models the rhombohedral ($R\overline{3}c$) crystal structure of $LaCoO_3$ at 293 K.[S17] In order to study the possible effects of providing different initial guesses to the atomic structure of studied rhombohedral systems, an alternate set of structural data values consisting of a different rhombohedral angle[S18] and atomic coordinates[S19] corresponding to $LaCrO_3$ under similar experimental conditions was also applied to calculations of oxygen vacancy formation energy in $LaCrO_3$ and $LaMnO_3$. Though other sets of structural data exist and can be used

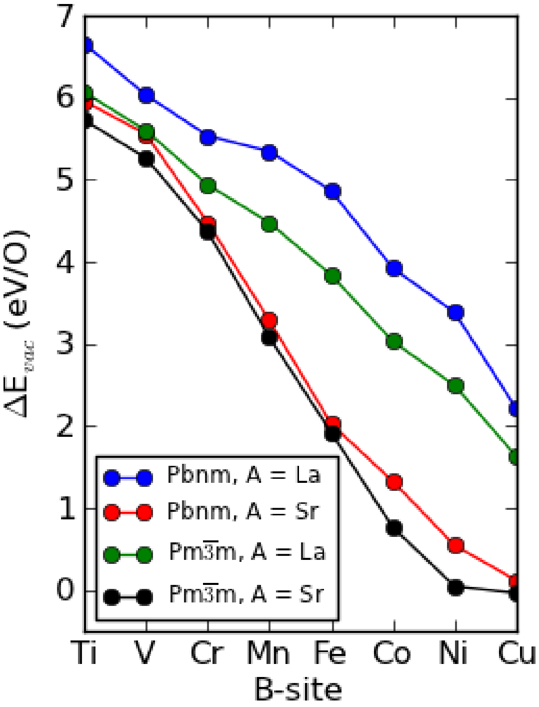

Figure S4: Oxygen vacancy formation energies in 20 atom orthorhombic ($Pbnm$ unit cell) and cubic ($Pm\bar{3}m$ supercell) perovskite crystal structures. Oxygen vacancies are formed at the $8d$ site in the above plot.

to illustrate the effects of providing different initial structural guesses to the systems studied,[S20] the purpose of applying different initial guesses to several test cases (e.g.: $LaCrO_3$ and $LaMnO_3$) was to illustrate that structural relaxations in DFT from different initial guesses of atomic structure give the same final energetic trend results within an energetic tolerance. By proving the equivalency of the results achieved by both initial structural guesses, the generalization of initial guesses of atomic structural data from one material to many materials can be justified. In the ”standard” atomic structural data set employed across all studied perovskites,[S17] a rhombohedral angle of 60.8° was applied to sets of constant volume (ISIF = 4) structural minimizations centered around the ground state equilibrium cell volumes of particular systems. In the ”alternate” set of structural data,[S18,S19] a rhombohedral angle of 60.4°[S18] was applied to a different set of atomic coordinates.[S19] From these data sets, three different POSCAR files constituting three different calculations were generated

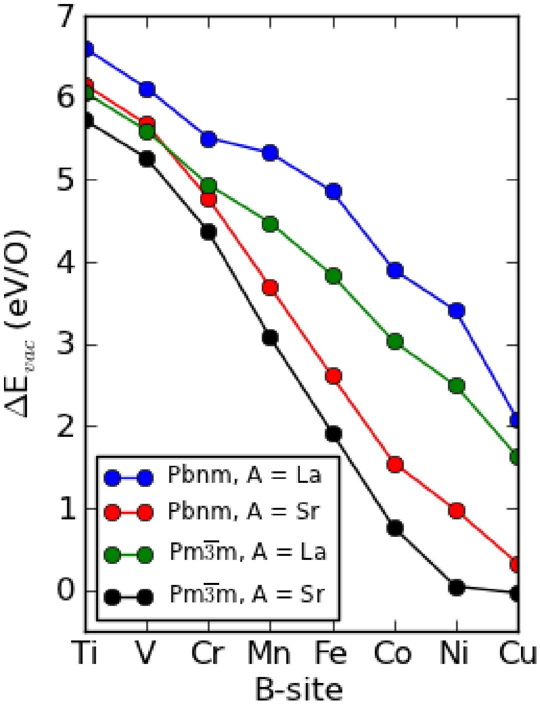

Figure S5: Oxygen vacancy formation energies in 20 atom orthorhombic (*Pbnm* unit cell) and cubic ($Pm\bar{3}m$ supercell) perovskite crystal structures. Oxygen vacancies are formed at the 4*c* site in the above plot.

to evaluate the effects of changing rhombohedral angle and atomic coordinate initial guesses on energetic data, namely calculations in which the ”standard” data set, ”alternate” atomic coordinates and a ”standard” rhombohedral angle, or the ”alternate” data set were applied. These calculations were applied to various no vacancy and one vacancy systems as is shown below in appropriately titled POSCAR files and captioned Birch-Murnaghan[S10] fits of total energy ranges to equilibrium cell volume:

$LaCrO_3$ and $LaMnO_3$, standard coordinate set and rhombohedral angle (see Figures S6 & S7 for no vacancy and Figure S8 for one vacancy systems):

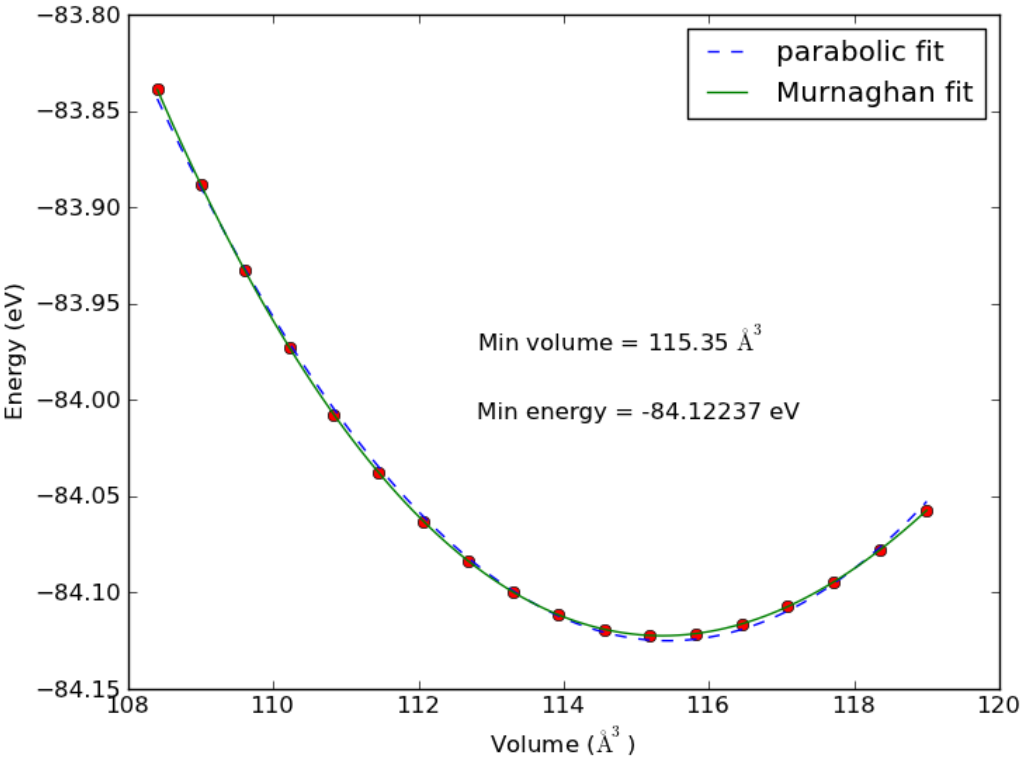

Figure S6: $LaCrO_3$ without oxygen vacancies, initialized with the ”standard” atomic coordinates and rhombohedral angle.

Listing 24: POSCAR with Standard Rhombohedral Angle and Atomic Coordinates

```
5.49
5.49
0.494827   -0.285688   0.820688
0            0.571377   0.820688
-0.494827  -0.285688   0.820688
2 2 6
direct
 0.25      0.25      0.25
 0.75      0.75      0.75
 0         0         0
 0.5       0.5       0.5
 0.30589   0.19411   0.75
 0.75      0.30589   0.19411
 0.19411   0.75      0.30589
-0.30589   0.80589   0.25
 0.25     -0.30589   0.80589
 0.80589   0.25     -0.30589
```

$LaCrO_3$, alternate coordinate set and standard rhombohedral angle (see Figure S9 for the no vacancy system):

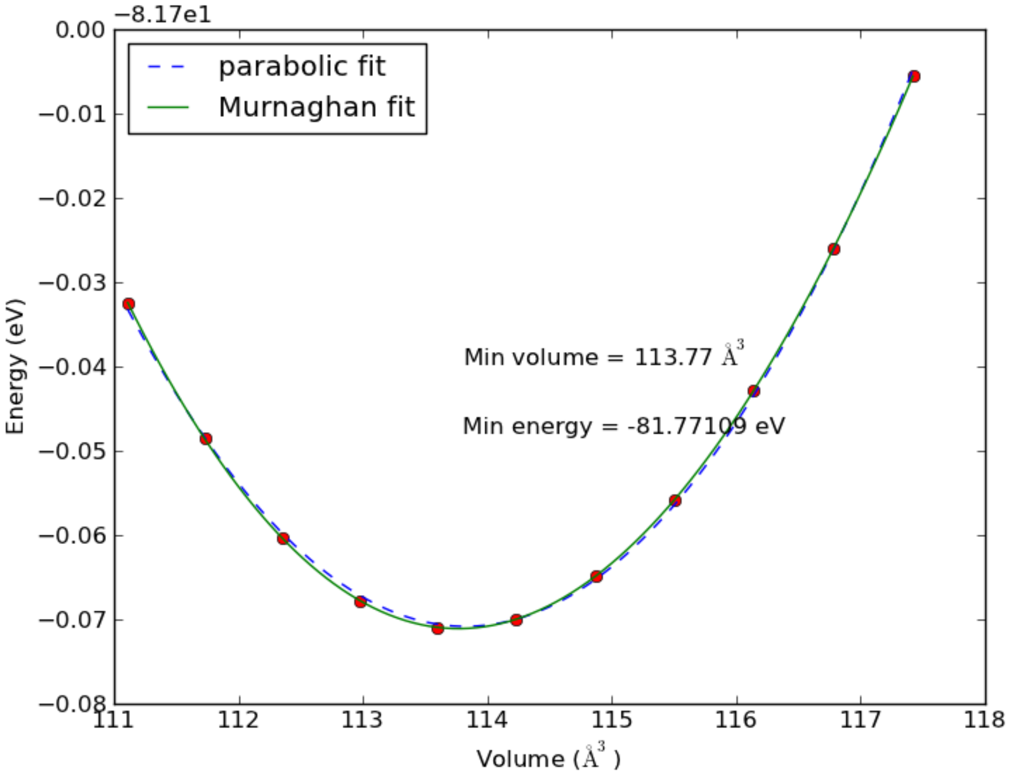

Figure S7: $LaMnO_3$ without oxygen vacancies, initialized with the ”standard” atomic coordinates and rhombohedral angle.

Listing 25: POSCAR with Standard Rhombohedral Angle and Alternate Atomic Coordinates

```
5.49
5.49
0.494827   -0.285688   0.820688
0           0.571377   0.820688
-0.494827  -0.285688   0.820688
2 2 6
direct
 0.25     0.25     0.25
 0.75     0.75     0.75
 0        0        0
 0.5      0.5      0.5
 0.1982   0.3018   0.75
 0.75     0.1982   0.3018
 0.3018   0.75     0.1982
-0.1982   0.6982   0.25
 0.25    -0.1982   0.6982
 0.6982   0.25    -0.1982
```

$LaCrO_3$, alternate coordinate set and rhombohedral angle (see Figures S10 and S11 for no vacancy and Figure S12 for one vacancy systems):

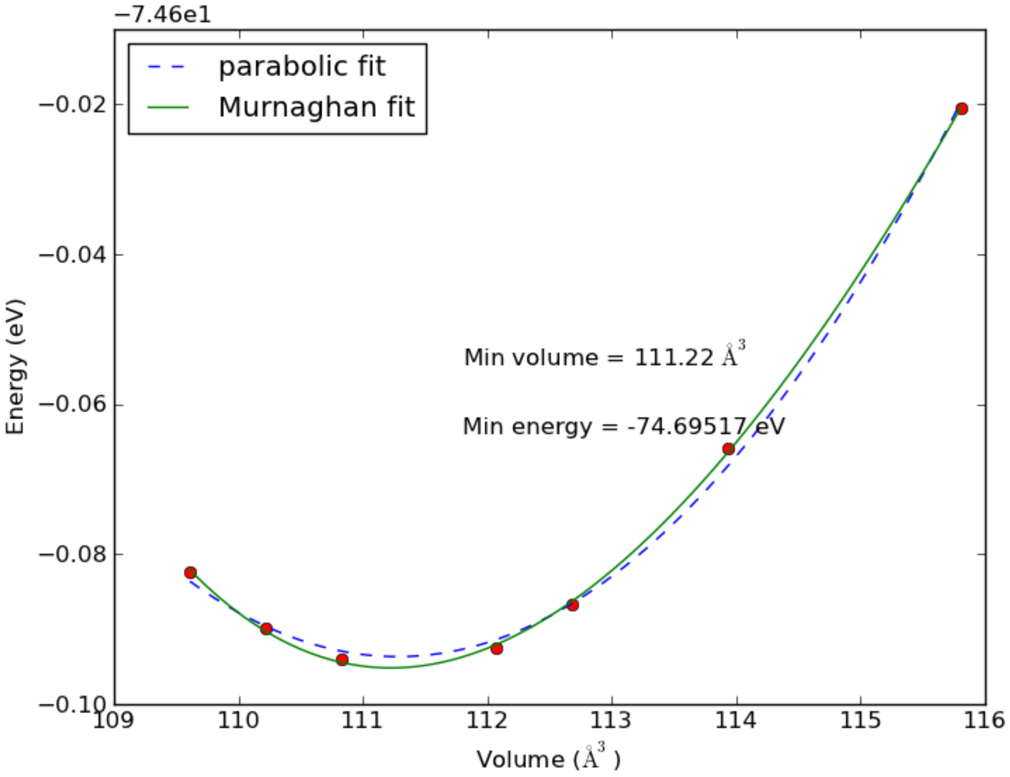

Figure S8: $LaCrO_3$ with one oxygen vacancy, initialized with the ”standard” atomic coordinates and rhombohedral angle.

Listing 26: POSCAR with Alternate Rhombohedral Angle and Atomic Coordinates

```
5.43
5.43
0.506018   -0.29215    0.811538
0           0.5843     0.811538
-0.506018  -0.29215    0.811538
2 2 6
direct
 0.25       0.25     0.25
 0.75       0.75     0.75
 0          0        0
 0.5        0.5      0.5
 0.1982     0.3018   0.75
 0.75       0.1982   0.3018
 0.3018     0.75     0.1982
-0.1982     0.6982   0.25
 0.25      -0.1982   0.6982
 0.6982     0.25    -0.1982
```

The standalone Birch-Murnaghan equation of state calculator and plotting code used to generate these fits is reproduced below:

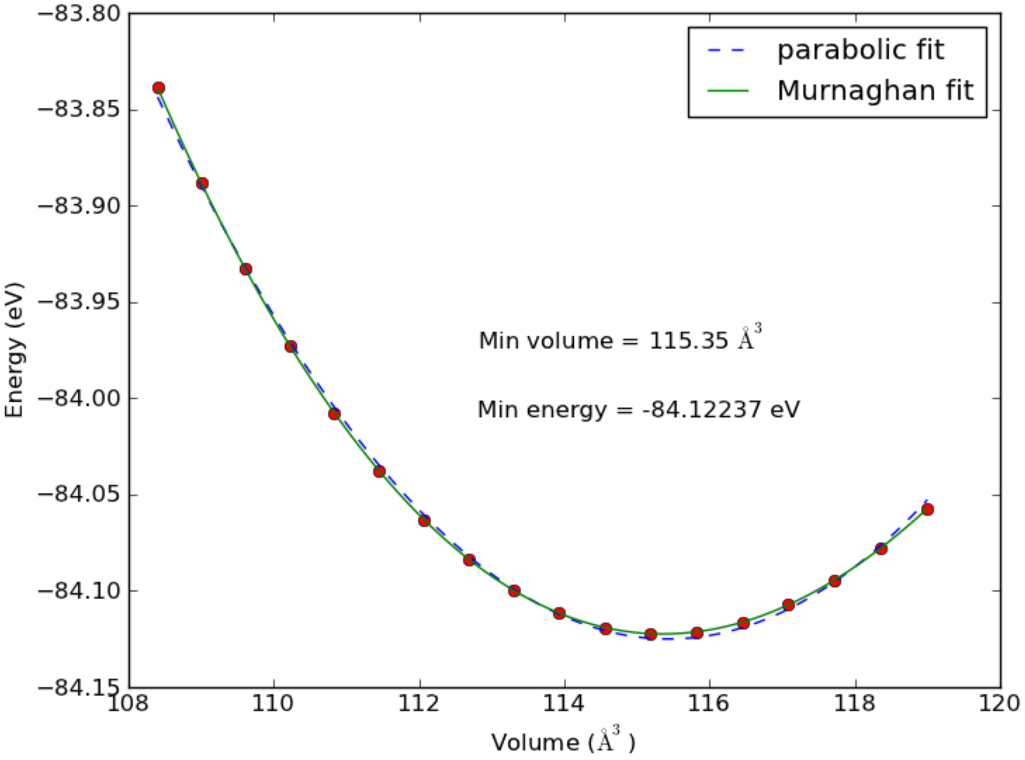

Figure S9: $LaCrO_3$ without oxygen vacancies, initialized with the ”alternate” atomic coordinates and ”standard” rhombohedral angle.

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator
from scipy.optimize import leastsq
import os
import fnmatch

Bsite = 'Mn'
struct = 'NoVac'
coord = 'matchCo'

Eval = "Esummary"
Vval = "Vsummary"
E_list = []
V_list = []

conv_V = 0.529177249      # convert Bohr to Angstrom (QE only)
```

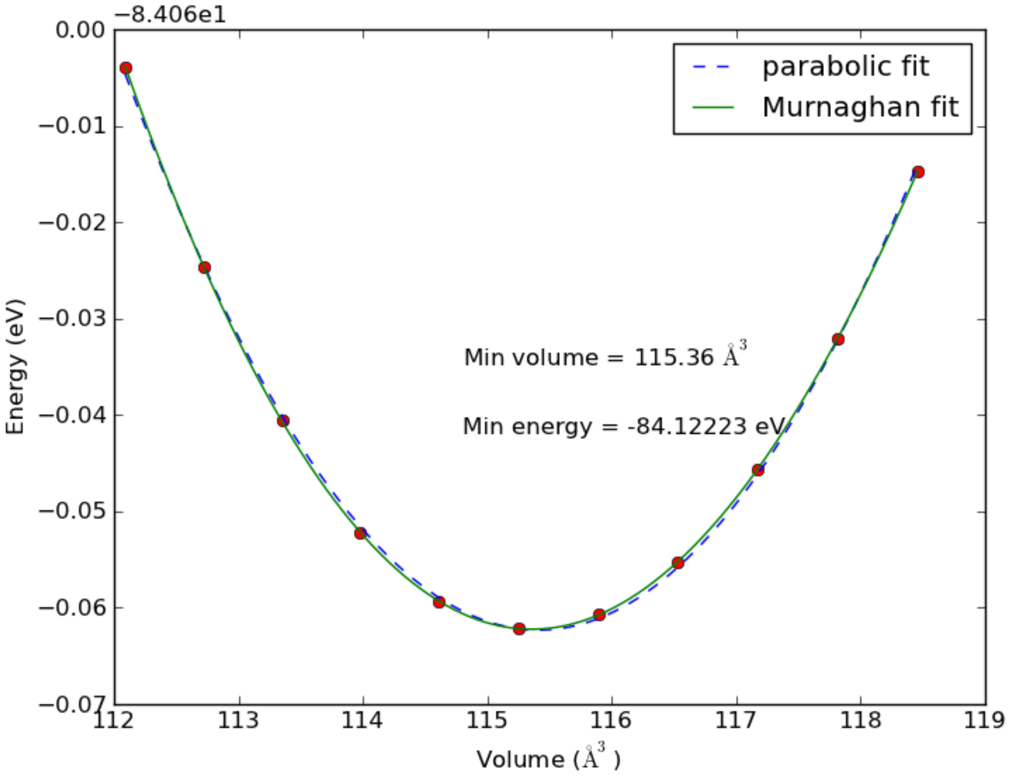

Figure S10: $LaCrO_3$ without oxygen vacancies, initialized with the ”alternate” atomic coordinates and rhombohedral angle.

```

with open(Eval, ’r’) as e:
    for line in e:
        E_str = line.strip()
        E_value = float(E_str)
        E_list.append(E_value)

with open(Vval, ’r’) as v:
    for line in v:
        V_str = line.strip()
        V_value = float(V_str)
        V_list.append(V_value)

E_array = np.array(E_list)
V_array = np.array(V_list)
Vol_fit = np.linspace(min(V_array),max(V_array),100)

```

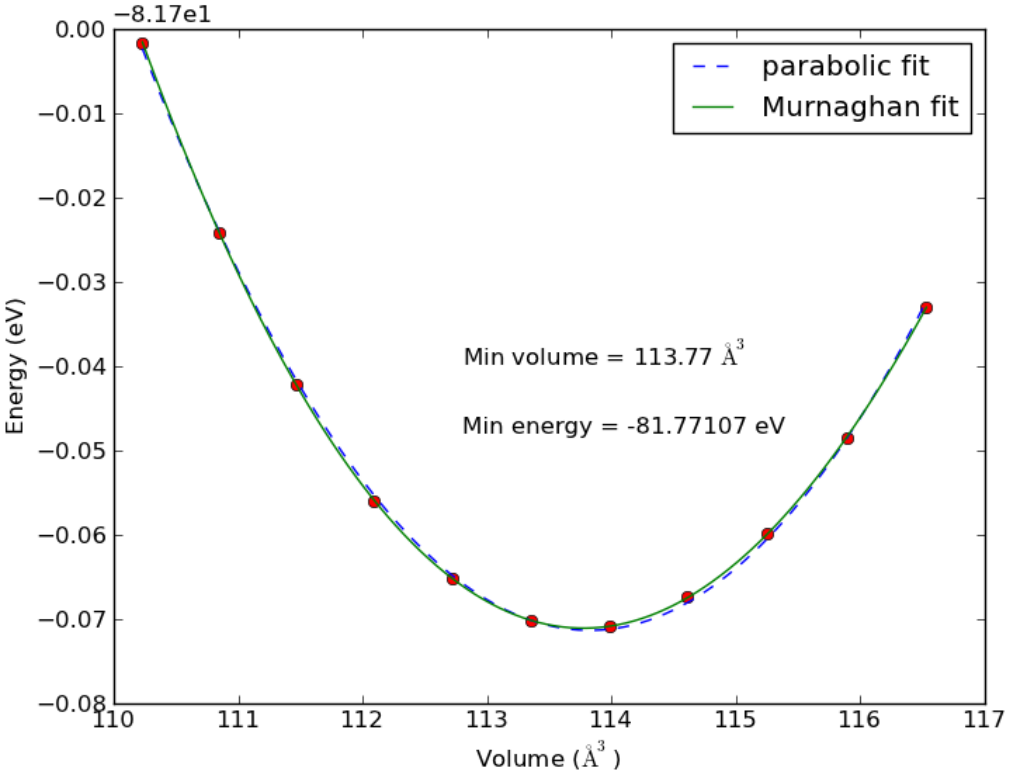

Figure S11: $LaMnO_3$ without oxygen vacancies, initialized with the ”alternate” atomic coordinates and rhombohedral angle.

```
a_quadfit,b_quadfit,c_quadfit = np.polyfit(V_array,E_array,2)

v0_init = -b_quadfit/(2*a_quadfit)
e0_init = a_quadfit*v0_init**2 + b_quadfit*v0_init + c_quadfit
b0_init = 2*a_quadfit*v0_init
bP_init = 4.00

def Murnaghan(parameters,vol):
    E0 = parameters[0]
    B0 = parameters[1]
    BP = parameters[2]
    V0 = parameters[3]

    E = E0 + B0*vol/BP*(((V0/vol)**BP)/(BP-1)+1) - V0*B0/(BP-1.)

    return E

```

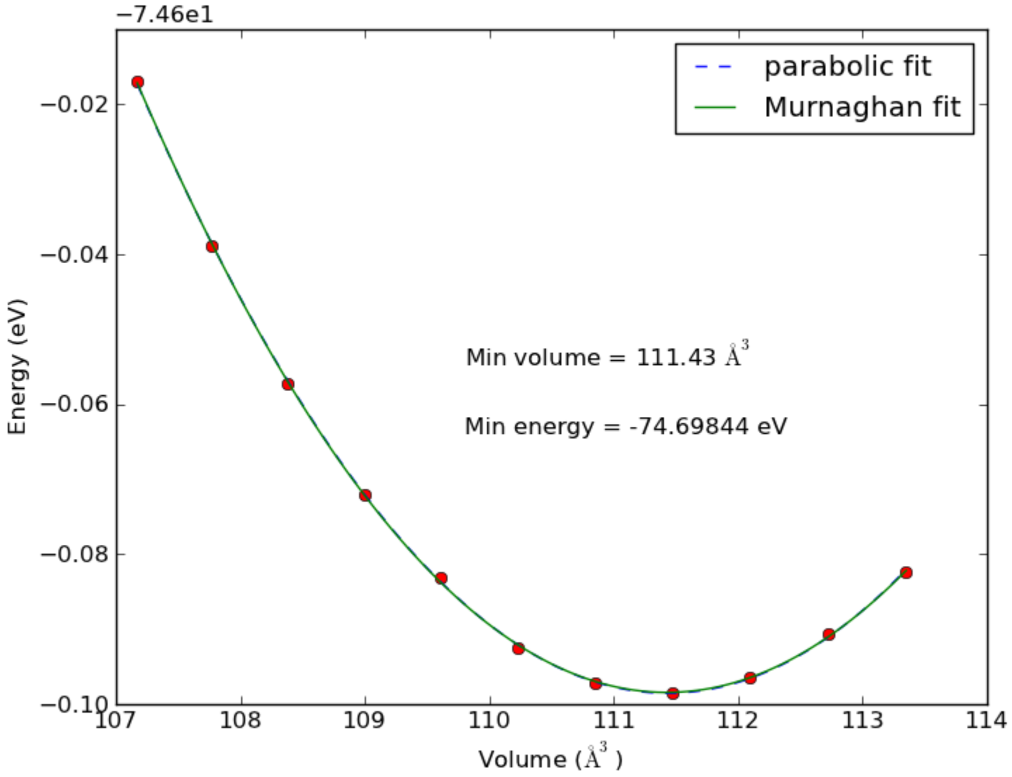

Figure S12: LaCrO$_3$ with one oxygen vacancy, initialized with the ”alternate” atomic coordinates and rhombohedral angle.

```
def objective(pars,y,x):
    err =  y - Murnaghan(pars,x)
    return err

x0 = [e0_init, b0_init, bP_init, v0_init]
murnpars, ier = leastsq(objective, x0, args=(E_array,V_array)) # this is from scipy

plt.plot(V_array,E_array,’ro’)
plt.plot(Vol_fit, a_quadfit*Vol_fit**2 + b_quadfit*Vol_fit + c_quadfit,’--’,label=’parabolic fit’)
plt.plot(Vol_fit, Murnaghan(murnpars,Vol_fit), label=’Murnaghan fit’)
plt.xlabel(’Volume ($\AAˆ3$)’)
plt.ylabel(’Energy (eV)’)
plt.legend(loc=’best’)

ax = plt.gca()

Vol_min = conv_V*conv_V*conv_V*murnpars[3]  # convert ground state volume from Bohrˆ3 to AAˆ3
```

```
Lat_min = conv_V*murnpars[3]**(1.0/3.0)      # convert ground state cubic lattice constant from Bohr to AA

print Vol_min
print Lat_min

plt.text(0.4, 0.5, 'Min volume = {0:1.2f} Bohr$^3$ = {1:1.2f} $\AA^3$'.format(murnpars[3],
                                                                                   murnpars[3]*conv_V*conv_V*conv_V),
         transform = ax.transAxes)
plt.text(0.4, 0.4, 'Min energy = {0:1.2f} Ry'.format(murnpars[0],
                                                     murnpars[0]),
         transform = ax.transAxes)
for ext in ['png', 'pdf', 'eps']:
    plt.savefig('BMurnFit_La' + Bsite + 'O3_' + struct + '_QE.' + ext)
plt.clf()
```

The primitive lattice vectors in the POSCAR files above, which characterize the initial guesses for the lattice constants and rhombohedral angle, can be calculated using various methods,[S8,S21] one of which is transcribed in the form of the matrix below:[S8]

$$\begin{pmatrix} x & -y & z \\ 0 & 2y & z \\ -x & -y & z \end{pmatrix}$$

The terms inserted into the matrix above can be calculated from the following equations, where α represents the rhombohedral angle of the unit cell:

$$x = \sqrt{\frac{1 - cos(\alpha)}{2}} \tag{21}$$

$$y = \sqrt{\frac{1 - cos(\alpha)}{6}} \tag{22}$$

$$z = \sqrt{\frac{1 + (2 \times cos(\alpha))}{3}} \tag{23}$$

The metric tensor for the rhombohedral unit cell can also be used to generate the primitive vectors above:[S21]

$$\begin{pmatrix} 1 & \cos(\alpha) & \cos(\alpha) \\ \cos(\alpha) & 1 & \cos(\alpha) \\ \cos(\alpha) & \cos(\alpha) & 1 \end{pmatrix}$$

Despite differences in the initial input atomic coordinate data applied to either $LaCrO_3$ and $LaMnO_3$ systems, structural relaxation reveals that the ground state DFT energies of each no vacancy structure are equivalent within a 0.001 eV tolerance, while the one vacancy case of $LaCrO_3$ has similar ground states energies regardless of initial structure within a 0.003 eV tolerance. Therefore, the same initial structural data can be employed across sets of different A-site and B-site cations to produce the energetic trends shown in this study.

3.6 Magnetism section

Data for the FM LaBO$_3$ plot generated in this section, which depicts the oxygen vacancy formation energies of FM systems with various supercell sizes and crystallographic symmetries, includes all studied B-sites and implements all other simplifications detailed in the main sections of the article. This data can currently be aggregated via an RBDMS command consistent with the relational algebraic expression below:

$$\Pi_{struct,num\_atoms,at1,at2,calc\_quantity}$$

$$\sigma[(struct =' Pm - 3m') \vee (struct =' R - 3c') \vee (struct =' Pbnm')]$$

$$\sigma \wedge [(morph =' perovskite')]$$

$$\sigma \wedge [(at1 =' La')]$$

$$\sigma \wedge [(num\_atoms = 5) \vee (num\_atoms = 10) \vee (num\_atoms = 20)]$$

$$\sigma \wedge [(correction =' none')]$$

$$\sigma \wedge [(mag =' FM')]$$

$$\sigma \wedge [(orientation =' bulk')]$$

$$\sigma \wedge [(calculator =' VASP')]$$

$$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$$

Data for the FM SrBO$_3$ plot generated in this section, which depicts the oxygen vacancy formation energies of FM systems with various supercell sizes and crystallographic symmetries, includes all studied B-sites and implements all other simplifications detailed in the main sections of the article. This data can currently be aggregated via an RBDMS command consistent with the relational algebraic expression below:

$\Pi_{struct,num\_atoms,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m') \vee (struct =' R - 3c') \vee (struct =' Pbnm')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' Sr')]$

$\sigma \wedge [(num\_atoms = 5) \vee (num\_atoms = 10) \vee (num\_atoms = 20) \vee (num\_atoms = 40)]$

$\sigma \wedge [(correction =' none')]$

$\sigma \wedge [(mag =' FM')]$

$\sigma \wedge [(orientation =' bulk')]$

$\sigma \wedge [(calculator =' VASP')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE]$

Data for the AFM LaBO$_3$ plot generated in this section, which depicts the oxygen vacancy formation energies of AFM systems with various supercell sizes, includes all studied B-sites and implements all other simplifications detailed in the main sections of the article. This data can currently be aggregated via an RBDMS command consistent with the relational algebraic expression below:

$\Pi_{num\_atoms,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' La')]$

$\sigma \wedge [(num\_atoms = 10) \vee (num\_atoms = 20) \vee (num\_atoms = 40)]$

$\sigma \wedge [(correction =' none')]$

$\sigma \wedge [(mag =' AFM') \vee (mag =' AAFM') \vee (mag =' CAFM') \vee (mag =' GAFM')]$

$\sigma \wedge [(orientation =' bulk')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE \bowtie OVac\_EV]$

Data for the AFM SrBO$_3$ plot generated in this section, which depicts the oxygen vacancy formation energies of AFM systems with various supercell sizes, includes all studied B-sites and implements all other simplifications detailed in the main sections of the article. This data can currently be aggregated via an RBDMS command consistent with the relational algebraic expression below:

$\Pi_{num\_atoms,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' Sr')]$

$\sigma \wedge [(num\_atoms = 10) \vee (num\_atoms = 20)]$

$\sigma \wedge [(correction =' none')]$

$\sigma \wedge [(mag =' AFM')]$

$\sigma \wedge [(orientation =' bulk')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE \bowtie OVac\_EV]$

3.6.1 Sample Input Files

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies relating to the ferromagnetic (FM) magnetism plots presented in the Magnetism subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy ($AB\mathrm{O}_3$) and one vacancy ($AB\mathrm{O}_{3\text{-}\delta}$) structures. The same KPOINTS file is used in both cases. These particular sample input files were used in LaMnO_3 structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a unit cell lattice constant of 3.81 Å. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 27: INCAR FM, No Vacancy

```
System = LaMnO3 Perovskite Trial, FM, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

ISPIN = 2
LORBIT = 11
MAGMOM = 1*0.0000 1*4.0000 3*0.0000
```

Listing 28: INCAR FM, 1 Vacancy

```
System = LaMnO3 Perovskite Trial, FM, 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

ISPIN = 2
LORBIT = 11
MAGMOM = 1*0.0000 1*4.0000 2*0.0000
```

Listing 29: KPOINTS

```
6x6x6
0
Monkhorst-Pack
6 6 6
0 0 0
```

Listing 30: POSCAR, No Vacancy

```
3.81, No Vacancy
   3.81000000000000
     1.0000000000000000    0.0000000000000000    0.0000000000000000
     0.0000000000000000    1.0000000000000000    0.0000000000000000
     0.0000000000000000    0.0000000000000000    1.0000000000000000
   La   Mn   O
     1     1     3
Direct
  0.0000000000000000  0.0000000000000000  0.0000000000000000
  0.5000000000000000  0.5000000000000000  0.5000000000000000
  0.5000000000000000  0.5000000000000000  0.0000000000000000
  0.5000000000000000  0.0000000000000000  0.5000000000000000
  0.0000000000000000  0.5000000000000000  0.5000000000000000
```

Listing 31: POSCAR, 1 Vacancy

```
3.81, 1 Vacancy
   3.81000000000000
     1.0000000000000000    0.0000000000000000    0.0000000000000000
     0.0000000000000000    1.0000000000000000    0.0000000000000000
     0.0000000000000000    0.0000000000000000    1.0000000000000000
   La   Mn   O
     1     1     2
Direct
  0.0027211270853281 -0.0025249716146828  0.0023674154474773
  0.5025136099202444  0.4975936110164044  0.5026560819021719
  0.5024488411985338 -0.0023673170270653  0.5027050286067404
  0.0025264217959022  0.4976486776253500  0.5027104740436096
```

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies relating to the anti-ferromagnetic (AFM) magnetism plots presented in the Magnetism subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy (ABO_3) and one vacancy ($ABO_{3\text{-}\delta}$) structures. INCAR files corresponding to A-type, C-type, and G-type AFM states (for the no vacancy case) are presented below. The same KPOINTS file is used in all cases, while each AFM state shares the same set of POSCAR files. These particular sample input files were used in $LaMnO_3$ structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a unit cell lattice constant of 3.81 Å. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 32: INCAR A-AFM, No Vacancy

```
System = LaMnO3 Perovskite Trial, A-type AFM, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

NUPDOWN = 0
ISPIN = 2
LORBIT = 11
MAGMOM = 8*0.0000 3*4.0000 1*-4.0000 1*4.0000 1*-4.0000 2*-4.0000 24*0.0000
```

Listing 33: INCAR C-AFM, No Vacancy

```
System = LaMnO3 Perovskite Trial, C-type AFM, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

NUPDOWN = 0
ISPIN = 2
LORBIT = 11
MAGMOM = 8*0.0000 1*4.0000 2*-4.0000 2*4.0000 2*-4.0000 1*4.0000 24*0.0000
```

Listing 34: INCAR G-AFM, No Vacancy

```
System = LaMnO3 Perovskite Trial, G-type AFM, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

NUPDOWN = 0
ISPIN = 2
LORBIT = 11
MAGMOM = 8*0.0000 1*4.0000 3*-4.0000 3*4.0000 1*-4.0000 24*0.0000
```

Listing 35: INCAR A-AFM, 1 Vacancy

```
System = LaMnO3 Perovskite Trial, A-type AFM, 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 550
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

EDIFF = 5E-06; EDIFFG = -0.01

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

IBRION = 1
ISIF = 4

VOSKOWN = 1

NUPDOWN = 0
ISPIN = 2
LORBIT = 11
MAGMOM = 8*0.0000 3*4.0000 1*-4.0000 1*4.0000 1*-4.0000 2*-4.0000 23*0.0000
```

Listing 36: KPOINTS

```
3x3x3
0
Monkhorst-Pack
3 3 3
0 0 0
```

```
# POSCAR, No Vacancy

3.92, No Vacancy
3.92
2 0 0
0 2 0
0 0 2
8 8 24
direct
0.0000 0.0000 0.0000
0.5000 0.0000 0.0000
0.0000 0.5000 0.0000
0.0000 0.0000 0.5000
0.5000 0.5000 0.0000
0.5000 0.0000 0.5000
0.0000 0.5000 0.5000
0.5000 0.5000 0.5000

0.2500 0.2500 0.2500
0.7500 0.2500 0.2500
0.2500 0.7500 0.2500
0.2500 0.2500 0.7500
0.7500 0.7500 0.2500
0.7500 0.2500 0.7500
0.2500 0.7500 0.7500
0.7500 0.7500 0.7500

0.2500 0.2500 0.0000
0.7500 0.2500 0.0000
0.2500 0.7500 0.0000
0.7500 0.7500 0.0000
0.2500 0.2500 0.5000
0.7500 0.2500 0.5000
0.2500 0.7500 0.5000
```

```
0.7500 0.7500 0.5000

0.2500 0.0000 0.2500
0.7500 0.0000 0.2500
0.2500 0.0000 0.7500
0.7500 0.0000 0.7500
0.2500 0.5000 0.2500
0.7500 0.5000 0.2500
0.2500 0.5000 0.7500
0.7500 0.5000 0.7500

0.0000 0.2500 0.2500
0.0000 0.7500 0.2500
0.0000 0.2500 0.7500
0.0000 0.7500 0.7500
0.5000 0.2500 0.2500
0.5000 0.7500 0.2500
0.5000 0.2500 0.7500
0.5000 0.7500 0.7500
```

```
# POSCAR, 1 Vacancy

3.92, 1 Vacancy
3.92000000000000
2 0 0
0 2 0
0 0 2
8     8    23
Direct
  0.0043244861035553  0.0089865134506992  0.0049248669395951
  0.5043231401739000  0.0017005259058259  0.0071035698145792
  0.0040652800837972  0.4997179668738934  0.0038927035283525
  0.0043287767058974  0.0103216017049444  0.5033826357422219
  0.5041912245517790  0.5071813967992704  0.0073906470558105
  0.5042756342946315  0.0011724938270852  0.5030687437824404
  0.0043229035741018  0.4990051021480670  0.5028076017584815
  0.5043581851572757  0.5065533684241877  0.5034609441350077
  0.2532827549633257  0.2542891944173236  0.2549774901006525
  0.7552491126291685  0.2542859670916745  0.2549920705155340
  0.2559225339421715  0.7542403003020285  0.2552361188077008
  0.2573874195099096  0.2542381047339223  0.7551523833219139
  0.7526527034619318  0.7542562833441877  0.2552519047507869
  0.7511998706967639  0.2542211713148777  0.7551424308574004
  0.2679314677910164  0.7542438166610681  0.7551643575572452
  0.7408020159482285  0.7542444391395552  0.7551700722314018
  0.2440386574623968  0.2531878207982836  0.0065134922767130
  0.7644535809886143  0.2537266081580204  0.0065423344880057
  0.2813745419110245  0.7553192073150522  0.0010031932942768
  0.7274891661172205  0.7548054875051716  0.0010904618757559
  0.2441138684803450  0.2553124857544174  0.5037315748998956
  0.7643057948773951  0.2549647236702154  0.5037102353115830
  0.2809556078719236  0.7531548336529098  0.5095210848265869
  0.7277368998349989  0.7535861136398696  0.5094662778658609
  0.2489731748352681  0.0051554464849795  0.2562183369610327
  0.7595582541896462  0.0050799788043282  0.2557644638916718
  0.2698235653649744  0.9976864180625603  0.7541599788852096
  0.7388621289981566  0.9976257809491110  0.7546467288943027
  0.2482013440465873  0.5033780701732234  0.2542683887230491
  0.7604040675592535  0.5034771655033642  0.2547029746605734
  0.2693673479269817  0.5107792360330220  0.7564166629060011
  0.7394557543199066  0.5108407187648907  0.7559974322805232
  0.0042572985466502  0.2536448927676955  0.2559212209565696
  0.0042143601036935  0.7546564486954588  0.2554686581715570
  0.0043129340802124  0.2539100469427989  0.7551590863341180
  0.0043409688550295  0.7541576095748725  0.7571798617327424
  0.5042681846784323  0.2547207471473903  0.2550208686794558
  0.5044112056602089  0.7539596076218690  0.2553120501585898
  0.5042831437035955  0.2545976378420538  0.7549730060268388
```

3.6.2 Plot Generation

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the $A =$ La (FM) plot presented in the Magnetism subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate this plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```python
# LaBO3 FM Magnetism Data Aggregator and Plot Generation
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Mag_state = ’FM’
Asite = ’La’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, gsym, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select sy.sym,ov.num_atoms,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where (sy.sym=’Pm-3m’ or sy.sym=’R-3c’ or sy.sym =’Pbnm’)
and s.morph=’perovskite’
and c1.at1=’La’
and (ov.num_atoms=5 or ov.num_atoms=10 or ov.num_atoms=20)
and ov.correction=’none’
and ov.mag=’FM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
;’’’):
    datapts_list = []
    a,b,c,d = row
    datapts_list.append( (a,b,c,d) )

    if a not in label_1:
        gsym += [a,b]

    if c not in label_2:
        Bsites += [c]

    label_1 += [a,b]
    label_2 += [c]
    Evac += [d]

    datapts_dict[row] = datapts_list
```

```

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k, l in datapts_dict:
    SortElist[i,j].append( (k,l) )
    SortElist[i,j].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[’Pm-3m’,5]
print EBsiteMatch[’Pm-3m’,10]
print EBsiteMatch[’Pm-3m’,20]
print EBsiteMatch[’R-3c’,10]
print EBsiteMatch[’Pbnm’,20]

x = [i for i in range( num_Bsites )]
x_2 = [0, 1, 2]
ax = plt.gca()

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[’Pm-3m’,5], ’bo-’, label = r’1 $\times$ 1 $\times$ 1’)
plt.plot(x, EBsiteMatch[’Pm-3m’,10], ’ro-’, label = r’1 $\times$ 1 $\times$ 2’)
plt.plot(x, EBsiteMatch[’Pm-3m’,20], ’go-’, label = r’2 $\times$ 2 $\times$ 1’)
plt.plot(x, EBsiteMatch[’R-3c’,10], ’ko-’, label = r’R$\overline{3}$c’)
plt.plot(x_2, EBsiteMatch[’Pbnm’,20], ’mo-’, label = r’Pbnm’)

plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (1.5, 6.5) )
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(loc = ’lower left’, prop={’size’:11})
```

```python
plt.text(0.035, 0.85, 'A = {0}, {1}'.format(Asite,Mag_state),
         style = 'oblique', color = 'white', fontsize = 16,
         transform = ax.transAxes,
         bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})

plt.gcf().subplots_adjust(left=0.15)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('Magstate' + Mag_state + '_' + Asite + '.' + ext)
plt.clf()
```

```
[5.491461357, 5.14702995, 4.819359073, 4.48618289, 3.935402755,
3.259031986, 2.681342989, 2.261258388]
[5.782325838, 5.422974499, 5.046133679, 4.605401268, 4.020821331,
3.27922669, 2.646609269, 1.882061713]
[6.059993651, 5.104277141, 5.240125322, 4.70354933, 3.99482176,
3.235884651, 2.490093714, 1.626762356]
[6.231747367, 6.139242132, 5.701020762, 5.150758961, 4.524855209,
3.976637719, 3.352752064, 2.54373933]
[6.435204552, 6.145285698, 5.760361657]
```

Table S7: Oxygen vacancy formation energies (eV) of ferromagnetic cubic supercell, rhombohedral unit cell, and orthorhombic unit cell (8d site oxygen vacancy) La$\boldsymbol{B}$O$_3$ systems without U correction.

Supercell, Symmetry	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
$1 \times 1 \times 1$, $Pm\bar{3}m$	5.49	5.15	4.82	4.49	3.94	3.26	2.68	2.26
$1 \times 1 \times 2$, $Pm\bar{3}m$	5.78	5.42	5.05	4.61	4.02	3.28	2.65	1.88
$2 \times 2 \times 1$, $Pm\bar{3}m$	6.06	5.10	5.24	4.70	3.99	3.24	2.49	1.63
$1 \times 1 \times 1$, $R\bar{3}c$	6.23	6.14	5.70	5.15	4.52	3.98	3.35	2.54
$1 \times 1 \times 1$, $Pbnm$	6.44	6.15	5.76	-	-	-	-	-

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the A = La (AFM) plot presented in the Magnetism subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate this plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```
# LaBO3 AFM Magnetism Data Aggregator and Plot Generation
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Mag_state = ’AFM’
Asite = ’La’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, label_3, magtyp, Bsites, Evac = [], [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select ov.mag,ov.num_atoms,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and (ov.num_atoms=10 or ov.num_atoms=20 or ov.num_atoms=40)
and ov.correction=’none’
and (ov.mag=’AFM’ or ov.mag=’AAFM’ or ov.mag=’CAFM’ or ov.mag=’GAFM’)
and ov.orientation=’bulk’
;’’’):
    datapts_list = []
    a,b,c,d = row
    datapts_list.append( (a,b,c,d) )

    if a not in label_1:
        magtyp += [a,b]

    if c not in label_2:
        Bsites += [c]

    label_1 += [a,b]
    label_2 += [c]
    Evac += [d]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k, l in datapts_dict:
    SortElist[i,j].append( (k,l) )
    SortElist[i,j].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}
AFM_222_list = []

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
        if j == (’CAFM’, 40) or j == (’GAFM’, 40):
            AFM_222_list.append(SortElist[j][k][1])

    EBsiteMatch[j] = matchlist

EBsiteMatch[’favAFM’,40] = sorted(AFM_222_list, reverse=True)

print EBsiteMatch[’AFM’,10]
print EBsiteMatch[’AFM’,20]
print EBsiteMatch[’favAFM’,40]
print EBsiteMatch[’AAFM’,40]

x = [i for i in range( num_Bsites )]
x_2 = [1, 2, 3]
x_3 = [3]
ax = plt.gca()

plt.figure(figsize=(3,4))
```

```python
plt.plot(x, EBsiteMatch['AFM',10], 'bo-', label = r'1 $\times$ 1 $\times$ 2')
plt.plot(x, EBsiteMatch['AFM',20], 'ro-', label = r'2 $\times$ 2 $\times$ 1')
plt.plot(x_2, EBsiteMatch['favAFM',40], 'go-', label = r'2 $\times$ 2 $\times$ 2')
plt.plot(x_3, EBsiteMatch['AAFM',40], 'ko-', label = r'A-type')
plt.xlabel( 'B-site' )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (1.5, 6.5) )
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'lower left', prop={'size':11})
plt.text(0.02, 0.85, 'A = {0}, {1}'.format(Asite,Mag_state),
         style = 'oblique', color = 'white', fontsize = 16,
         transform = ax.transAxes,
         bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})

plt.gcf().subplots_adjust(left=0.15)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('Magstate' + Mag_state + '_' + Asite + '.' + ext)
plt.clf()
```

```
[5.782329396, 5.580842668, 5.067818972, 4.65882232, 3.774776934,
3.200049091, 2.634354143, 1.885352171]
[6.058618847, 5.638196719, 5.146131568, 4.412178703, 3.810952976,
2.917845639, 2.48938304, 1.627849548]
[5.544884674, 5.245581121, 4.235279196]
[4.645482231]
```

Table S8: Oxygen vacancy formation energies of anti-ferromagnetic cubic supercell LaBO$_3$ systems without U correction.

Supercell, Magnetism	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
1 × 1 × 2, AFM	5.78	5.58	5.07	4.66	3.77	3.20	2.63	1.89
2 × 2 × 1, AFM	6.06	5.64	5.15	4.41	3.81	2.92	2.49	1.63
2 × 2 × 2, AFM	-	5.54	5.25	4.24	-	-	-	-
2 × 2 × 2, AAFM	-	-	-	4.64	-	-	-	-

For the $2 \times 2 \times 2$ series of the above calculations, $B =$ V, Cr, Mn and these systems respectively observe C-type, G-type, and C-type AFM magnetic states, while for the A-type series, $B =$ Mn.

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the $A =$ Sr (FM) plot presented in the Magnetism subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate this plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```
# SrBO3 FM Magnetism Data Aggregator and Plot Generation
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Mag_state = ’FM’
Asite = ’Sr’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, gsym, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select sy.sym,ov.num_atoms,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where (sy.sym=’Pm-3m’ or sy.sym=’R-3c’ or sy.sym =’Pbnm’)
and s.morph=’perovskite’
and c1.at1=’Sr’
and (ov.num_atoms=5 or ov.num_atoms=10 or ov.num_atoms=20 or ov.num_atoms=40)
and ov.correction=’none’
and ov.mag=’FM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
;’’’):
    datapts_list = []
    a,b,c,d = row
    datapts_list.append( (a,b,c,d) )

    if a not in label_1:
        gsym += [a,b]

    if c not in label_2:
        Bsites += [c]

    label_1 += [a,b]
    label_2 += [c]
    Evac += [d]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k, l in datapts_dict:
    SortElist[i,j].append( (k,l) )
    SortElist[i,j].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[’Pm-3m’,5]
print EBsiteMatch[’Pm-3m’,10]
print EBsiteMatch[’Pm-3m’,20]
print EBsiteMatch[’Pm-3m’,40]
print EBsiteMatch[’R-3c’,10]
print EBsiteMatch[’Pbnm’,20]

```

```
x = [i for i in range( num_Bsites )]
ax = plt.gca()
x_2 = [2, 3, 4]
x_3 = [0, 1, 2, 3, 4]
plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[’Pm-3m’,5], ’bo-’, label = r’1 $\times$ 1 $\times$ 1’)
plt.plot(x, EBsiteMatch[’Pm-3m’,10], ’ro-’, label = r’1 $\times$ 1 $\times$ 2’)
plt.plot(x, EBsiteMatch[’Pm-3m’,20], ’go-’, label = r’2 $\times$ 2 $\times$ 1’)
plt.plot(x_2, EBsiteMatch[’Pm-3m’,40], ’ko-’, label = r’2 $\times$ 2 $\times$ 2’)
plt.plot(x, EBsiteMatch[’R-3c’,10], ’mo-’, label = r’R$\overline{3}$c’)
plt.plot(x_3, EBsiteMatch[’Pbnm’,20], ’yo-’, label = r’Pbnm’)

plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (-0.5, 6.5) )
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(loc = ’upper right’, prop={’size’:10})
plt.text(0.035, 0.85, ’A = {0}, {1}’.format(Asite,Mag_state),
         style = ’oblique’, color = ’white’, fontsize = 16,
         transform = ax.transAxes,
         bbox={’facecolor’:’blue’, ’alpha’:0.5, ’pad’:10})

plt.gcf().subplots_adjust(left=0.15)

for ext in [’png’, ’pdf’, ’eps’]:
    plt.savefig(’Magstate’ + Mag_state + ’_’ + Asite + ’.’ + ext)
plt.clf()
```

```
[5.632547272, 5.100106661, 4.073631121, 3.268625598, 2.887217512,
2.013142446, 1.139505804, 0.448947572]
[5.768203402, 5.274080262, 3.742868094, 2.849843743, 2.731914835,
1.817186632, 0.549742934, 0.359598855]
[5.672393544, 5.266724894, 3.816760006, 2.048193877, 2.374782817,
1.381759101, 0.224631029, -0.037931212]
[2.750521829, 1.769689432, 2.088740008]
[6.181864569, 5.543998457, 4.205324378, 3.353369667, 3.29530036,
2.23641029, 1.212210921, 0.718446456]

[5.893304323, 5.384027989, 3.4812847, 2.324969355, 2.563100538]
```

Table S9: Oxygen vacancy formation energies of ferromagnetic cubic supercell, rhombohedral unit cell, and orthorhombic unit cell ($8d$ site oxygen vacancy) SrBO_3 systems without U correction.

Supercell, Symmetry	Ti	V	Cr	Mn	Fe	Co	Ni	Cu
$1 \times 1 \times 1$, $Pm\overline{3}m$	5.63	5.10	4.07	3.27	2.89	2.01	1.14	0.45
$1 \times 1 \times 2$, $Pm\overline{3}m$	5.77	5.27	3.74	2.85	2.73	1.82	0.55	0.36
$2 \times 2 \times 1$, $Pm\overline{3}m$	5.67	5.27	3.82	2.05	2.37	1.38	0.22	-0.04
$1 \times 1 \times 1$, $R\overline{3}c$	6.18	5.54	4.21	3.35	3.30	2.24	1.21	0.72
$2 \times 2 \times 2$, $Pm\overline{3}m$	-	-	2.76	1.77	2.09	-	-	-
$1 \times 1 \times 1$, $Pbnm$	5.89	5.38	3.48	2.32	2.56	-	-	-

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the A = Sr (AFM) plot presented in the Magnetism subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate this plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded.

```python
# SrBO3 AFM Magnetism Data Aggregator and Plot Generation
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Mag_state = ’AFM’
Asite = ’Sr’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, atomnum, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select ov.num_atoms,c2.at2,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’Sr’
and (ov.num_atoms=10 or ov.num_atoms=20)
and ov.correction=’none’
and ov.mag=’AFM’
and ov.orientation=’bulk’
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        atomnum += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])

    EBsiteMatch[j] = matchlist

print EBsiteMatch[10]
print EBsiteMatch[20]

x = [i for i in range( num_Bsites )]
ax = plt.gca()
plt.figure(figsize=(3,4))

plt.plot(x, EBsiteMatch[10], ’ro-’, label = r’1 $\times$ 1 $\times$ 2’)
plt.plot(x, EBsiteMatch[20], ’go-’, label = r’2 $\times$ 2 $\times$ 1’)

plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylim( (-0.5, 6.5) )
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(loc = ’upper right’, prop={’size’:10})
plt.text(0.025, 0.85, ’A = {0}, {1}’.format(Asite,Mag_state),
         style = ’oblique’, color = ’white’, fontsize = 16,
         transform = ax.transAxes,
         bbox={’facecolor’:’blue’, ’alpha’:0.5, ’pad’:10})

plt.gcf().subplots_adjust(left=0.15)

for ext in [’png’, ’pdf’, ’eps’]:
    plt.savefig(’Magstate’ + Mag_state + ’_’ + Asite + ’.’ + ext)
plt.clf()
```

```
[5.772868736, 5.164858433, 3.898073989, 2.802676992, 2.505322695,
 1.524794041, 0.470399562, 0.360720863]
[5.725268723, 5.231009227, 3.615267316, 2.621154355, 1.71204049,
 0.904995027, 0.156551026, -0.037914185]
```

Table S10: Oxygen vacancy formation energies (eV) of anti-ferromagnetic cubic supercell SrBO$_3$ systems without U correction.

Supercell, Symmetry	**Ti**	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**	**Cu**
$1 \times 1 \times 2$, $Pm\bar{3}m$	5.77	5.16	3.90	2.80	2.51	1.52	0.47	0.36
$2 \times 2 \times 1$, $Pm\bar{3}m$	5.73	5.23	3.62	2.62	1.71	0.90	0.16	-0.04

3.7 DFT+U section

Data for the A = La plot generated in this section, which depicts the oxygen vacancy formation energies of unit cell systems with Hubbard U parameters attached to B-site 3d cation orbitals, encompasses all studied B-sites and applies all other simplifications (with the exclusion of ignoring electron-electron interaction error) to its constituent calculations. In the current state of the database, this data can be aggregated via an RBDMS command adhering to the relational algebraic expression below:

$\Pi_{U\_value,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' La')]$

$\sigma \wedge [(num\_atoms = 5)]$

$\sigma \wedge [(correction =' U\_3d')]$

$\sigma \wedge [(mag =' PM')]$

$\sigma \wedge [(orientation =' bulk')]$

$\sigma \wedge [(calculator =' VASP')]$

$\sigma \wedge [(U\_value =' 0.00') \vee (U\_value =' 2.00') \vee (U\_value =' 2.50') \vee (U\_value =' 3.00')$

$\sigma \vee (U\_value =' 3.50') \vee (U\_value =' 4.00') \vee (U\_value =' 4.50')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE \bowtie OVac\_OrderU]$

Data for the A = Sr plot generated in this section, which depicts the oxygen vacancy formation energies of unit cell systems with Hubbard U parameters attached to B-site $3d$ cation orbitals, encompasses all studied B-sites and applies all other simplifications (with the exclusion of ignoring electron-electron interaction error) to its constituent calculations. In the current state of the database, this data can be aggregated via an RBDMS command adhering to the relational algebraic expression below:

$\Pi_{U\_value,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' Sr')]$

$\sigma \wedge [(num\_atoms = 5)]$

$\sigma \wedge [(correction =' U\_3d')]$

$\sigma \wedge [(mag =' PM')]$

$\sigma \wedge [(orientation =' bulk')]$

$\sigma \wedge [(calculator =' VASP')]$

$\sigma \wedge [(U\_value =' 0.00') \vee (U\_value =' 2.00') \vee (U\_value =' 3.50') \vee (U\_value =' 4.00')$

$\sigma \vee (U\_value =' 4.50') \vee (U\_value =' 5.50') \vee (U\_value =' 6.50')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE \bowtie OVac\_OrderU]$

Data for the A = La plot generated in this section, which depicts the oxygen vacancy formation energies of unit cell systems with Hubbard U parameters attached to A-site (La) $4f$ cation orbitals, encompasses all studied B-sites and applies all other simplifications (with the exclusion of ignoring electron-electron interaction error) to its constituent calculations. In the current state of the database, this data can be aggregated via an RBDMS command adhering to the relational algebraic expression below:

$\Pi_{U\_value,at1,at2,calc\_quantity}$

$\sigma[(struct =' Pm - 3m')]$

$\sigma \wedge [(morph =' perovskite')]$

$\sigma \wedge [(at1 =' La')]$

$\sigma \wedge [(num\_atoms = 5)]$

$\sigma \wedge [(correction =' U\_4f')]$

$\sigma \wedge [(mag =' none')]$

$\sigma \wedge [(orientation =' bulk')]$

$\sigma \wedge [(U\_value =' 0.00') \vee (U\_value =' 3.00') \vee (U\_value =' 6.00') \vee (U\_value =' 12.00')]$

$[Symmetry \bowtie Structure \bowtie Composition1 \bowtie Composition2 \bowtie OVac\_FormE \bowtie OVac\_OrderU]$

3.7.1 Sample Input Files

Samples of the input INCAR, KPOINTS, and POSCAR files used to calculate the oxygen vacancy formation energies presented in the ”Effect of U on oxygen vacancy formation energy” portion of the DFT + U subsection of the Results and Discussion section are shown below and labeled accordingly for both no vacancy ($AB\text{O}_3$) and one vacancy ($AB\text{O}_{3\text{-}\delta}$) structures. The same KPOINTS file is used in both cases. These particular sample input files were used in LaFeO_3 structural relaxation calculations run at a constant cell volume (ISIF = 4) corresponding to a lattice constant of 3.87 Å and input U value of 4.00 eV added to B-site 3d orbitals. In both no vacancy and one vacancy cases, energies resulting from these calculations were fitted to their respective cell volumes via the Birch-Murnaghan equation of state[S10] to resolve total energies used in the calculation of oxygen vacancy formation energies.

Listing 37: INCAR No Vacancy

```
System = LaFeO3 Perovskite Trial, U = 4.00 eV on 3d B-site, No Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 600
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 1; SYMPREC = 1E-06

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = 5E-05

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

ISPIN = 2

LDAU = .TRUE.
LDAUTYPE = 2
LDAUL = -1 2 -1
LDAUPRINT = 1

LASPH = .TRUE.
LMAXMIX = 4

LDAUJ = 0.00 0.00 0.00
LDAUU =0.00  4.00  0.00
```

Listing 38: INCAR 1 Vacancy

```
System = LaFeO3 Perovskite Trial, U = 4.00 eV on 3d B-site, 1 Vacancy
ISTART = 0 ; ICHARG = 2
ENCUT = 600
ISMEAR = 0 ; SIGMA = 0.05

ISYM = 0

IBRION = 1
ISIF = 4
EDIFF = 5E-06; EDIFFG = -0.005

MAXMIX = -100
NELMIN = 5
NELMDL = -10
NELM = 200
NSW = 100

ISPIN = 2

LDAU = .TRUE.
LDAUTYPE = 2
LDAUL = -1 2 -1
LDAUPRINT = 1

LASPH = .TRUE.
LMAXMIX = 4

LDAUJ = 0.00 0.00 0.00
LDAUU =0.00  4.00  0.00
```

Listing 39: KPOINTS

```
8x8x8
0
Monkhorst
8 8 8
0 0 0
```

Listing 40: POSCAR No Vacancy

```
3.87, LaFeO3 at U = 4.00 eV, No Vacancy
3.87
1 0 0
0 1 0
0 0 1
1 1 3
direct
0.000000 0.000000 0.000000
0.500000 0.500000 0.500000
0.500000 0.500000 0.000000
0.500000 0.000000 0.500000
0.000000 0.500000 0.500000
```

Listing 41: POSCAR 1 Vacancy

```
3.87, LaFeO3 at U = 4.00 eV, 1 Vacancy
3.87
1 0 0
0 1 0
0 0 1
1 1 2
direct
0.000010 0.000030 0.010400
0.500030 0.490080 0.500010
0.500100 0.000050 0.500020
0.010070 0.500190 0.500009
```

3.7.2 Plot Generation

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the "A = Sr, 3d" plot presented in the "Effect of U on oxygen vacancy formation energy" portion of the DFT + U subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded. Note that data for many of these systems exists from U = 2.00 to 8.00 eV in 0.5 eV increments, thus ovu.U_value variables can be set in the data query to access different trends of constant U.

```python
# DFT + U Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Asite = 'Sr'
corr = '3d'

db = sqlite3.connect('EOCCS_data.sqlite')

label_1, label_2, U_value, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute('''
select ovu.U_value, c2.at2, ov.calc_quantity from Structure as s
inner join Symmetry as sym on s.struct=sym.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ovu.cID=Ov.cID
where sym.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’Sr’
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and (ovu.U_value = ’0’ or ovu.U_value = ’2’ or ovu.U_value = ’3.5’
or ovu.U_value = ’4’ or ovu.U_value = ’4.5’ or ovu.U_value = ’5.5’
or ovu.U_value = ’6.5’)
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        U_value += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}
```

```

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[0.0]
print EBsiteMatch[2.0]
print EBsiteMatch[3.5]
print EBsiteMatch[4.0]
print EBsiteMatch[4.5]
print EBsiteMatch[5.5]
print EBsiteMatch[6.5]

x = [i for i in range( num_Bsites )]
ax = plt.gca()

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[0.0], color = ’c’,
         linestyle = ’dashed’, linewidth = 1.5, dash_capstyle = ’round’,
         dash_joinstyle = ’round’, label = r’U = 0’)
plt.plot(x, EBsiteMatch[2.0], color = ’y’,
         linestyle = ’dashed’, linewidth = 1.5, dash_capstyle = ’round’,
         dash_joinstyle = ’round’, label = r’U = 2’)
plt.plot(x, EBsiteMatch[3.5], ’ko-’, linewidth = 2.0, label = r’U = 3.5’)
plt.plot(x, EBsiteMatch[4.0], ’mo-’, linewidth = 2.0, label = r’U = 4’)
plt.plot(x, EBsiteMatch[4.5], ’go-’, linewidth = 2.0, label = r’U = 4.5’)
plt.plot(x, EBsiteMatch[5.5], ’bo-’, linewidth = 2.0, label = r’U = 5.5’)
plt.plot(x, EBsiteMatch[6.5], ’ro-’, linewidth = 2.0, label = r’U = 6.5’)

plt.xlabel( ’B-site’ )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylabel(’$\Delta$E$_{vac}$ (eV/O)’)
plt.legend(loc = ’upper right’, prop={’size’:9})
plt.text(0.048, 0.845, ’A = {0}, 3d’.format(Asite),
         style = ’oblique’, color = ’white’, fontsize = 16,
    transform = ax.transAxes,
    bbox={’facecolor’:’blue’, ’alpha’:0.5, ’pad’:10})
```

```
plt.gcf().subplots_adjust(left=0.16)
for ext in ['png', 'pdf', 'eps']:
    plt.savefig('TestU_' + corr + 'B_A' + Asite + '.' + ext)
plt.clf()
```

```
[5.77905119, 5.159387203, 4.206012261, 3.447335378, 3.048426258,
2.115858158, 1.263919244, 0.747871163]
[5.758506489, 4.859532785, 3.670123881, 3.222528704, 2.872084991,
1.983306055, 1.167332198, 0.66181245]
[5.643211632, 4.397455309, 3.113720251, 3.011480724, 2.588517151,
1.769428545, 1.16256433, 0.591864571]
[5.576475808, 4.171164326, 2.933365067, 2.912751407, 2.495814885,
1.697734298, 1.194296065, 0.567726571]
[5.502750768, 3.938632484, 2.767486918, 2.81316664, 2.406136724,
1.625454317, 1.417031463, 0.543311072]
[5.347253394, 3.455418683, 2.520177941, 2.630947442, 2.238336204,
1.560492289, 1.520721283, 0.493317522]
[5.127248983, 2.953940181, 2.325477591, 2.448807779, 2.101046664,
1.481586129, 1.63962244, 0.44224582]
```

Table S11: Oxygen vacancy formation energies (eV) of paramagnetic cubic unit cell Sr*B*O$_3$ systems with U corrections perturbing 3d *B*-site orbitals.

U_{3d}	**Ti**	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**	**Cu**
0.0	5.78	5.16	4.21	3.45	3.05	2.12	1.26	0.75
2.0	5.76	4.86	3.67	3.22	2.87	1.98	1.17	0.66
3.5	5.64	4.40	3.11	3.01	2.59	1.77	1.16	0.59
4.0	5.58	4.17	2.93	2.91	2.50	1.70	1.19	0.57
4.5	5.50	3.94	2.77	2.81	2.41	1.63	1.42	0.54
5.5	5.35	3.46	2.52	2.63	2.24	1.56	1.52	0.49
6.0	5.23	3.21	2.42	2.54	2.17	1.52	1.53	0.47
6.5	5.13	2.95	2.33	2.45	2.10	1.48	1.64	0.44

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the ”$A =$ La, 3d” plot presented in the ”Effect of U on oxygen vacancy formation energy” portion of the DFT + U subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded. Note that data for many of these systems exists from U = 2.00 to 4.00 eV and 6.00 to 8.00 eV (2.00 to 8.00 eV for B = Fe, Co, Ni) in 0.5 eV increments, thus ovu.U_value variables can be set in the data query to access different trends of constant U.

```
# DFT + U Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Asite = ’La’
corr = ’3d’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, U_value, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select ovu.U_value, c2.at2, ov.calc_quantity from Structure as s
inner join Symmetry as sym on s.struct=sym.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ovu.cID=Ov.cID
where sym.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and (ovu.U_value = ’0’ or ovu.U_value = ’2’ or ovu.U_value = ’2.5’
or ovu.U_value = ’3’ or ovu.U_value = ’3.5’ or ovu.U_value = ’4’
or ovu.U_value = ’4.5’)
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        U_value += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Ti": 0, "V": 1, "Cr": 2, "Mn": 3, "Fe": 4, "Co": 5, "Ni": 6, "Cu": 7}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[0.0]
print EBsiteMatch[2.0]
print EBsiteMatch[2.5]
print EBsiteMatch[3.0]
print EBsiteMatch[3.5]
print EBsiteMatch[4.0]
print EBsiteMatch[4.5]

x = [i for i in range( num_Bsites )]
x2 = [4, 5, 6]
ax = plt.gca()

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[0.0], color = 'k',
         linestyle = 'dashed', linewidth = 1.5, dash_capstyle = 'round',
         dash_joinstyle = 'round', label = r'U = 0')
plt.plot(x, EBsiteMatch[2.0], color = 'c',
         linestyle = 'dashed', linewidth = 1.5, dash_capstyle = 'round',
         dash_joinstyle = 'round', label = r'U = 2')
plt.plot(x, EBsiteMatch[2.5], color = 'y',
         linestyle = 'dashed', linewidth = 1.5, dash_capstyle = 'round',
         dash_joinstyle = 'round', label = r'U = 2.5')
plt.plot(x, EBsiteMatch[3.0], 'bo-', linewidth = 2.0, label = r'U = 3.0')
plt.plot(x, EBsiteMatch[3.5], 'ro-', linewidth = 2.0, label = r'U = 3.5')
plt.plot(x, EBsiteMatch[4.0], 'go-', linewidth = 2.0, label = r'U = 4.0')
plt.plot(x2, EBsiteMatch[4.5], 'mo-', linewidth = 2.0, label = r'U = 4.5')

plt.xlabel( 'B-site' )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'lower left', prop={'size':10})
plt.text(0.055, 0.85, 'A = {0}, {1}'.format(Asite,corr),
         style = 'italic', color = 'white', fontsize = 16,
    transform = ax.transAxes,
    bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})
plt.gcf().subplots_adjust(left=0.21)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('TestU_' + corr + 'B_A' + Asite + '.' + ext)
plt.clf()
```

```
[5.797296026, 5.345244383, 5.17020074, 4.828124849, 4.19219244,
 3.580299449, 3.234577097, 3.011754336]
```

```
[5.700779799, 5.422005852, 5.19906108, 4.931467462, 4.405170025,
 3.511612897, 3.160261209, 2.929555613]
[5.663542245, 5.336845115, 5.193705147, 4.909246736, 4.510935676,
 3.483561566, 3.174900237, 2.90697004]
[5.624371737, 5.255804489, 5.183227727, 4.880344982, 4.614715967,
 3.455471809, 3.183004353, 2.883557996]
[5.603070092, 5.167679008, 5.167649754, 4.846201941, 4.748168464,
 3.412100073, 3.192354726, 2.859338436]
[5.566167831, 5.072931763, 5.146991751, 4.808446708, 4.874620474,
 3.268499148, 3.203863805, 2.834332351]
[4.836116621, 3.118438435, 3.216825658]
```

Table S12: Oxygen vacancy formation energies (eV) of paramagnetic cubic unit cell La$\boldsymbol{B}$O$_3$ systems with U corrections perturbing 3d $\boldsymbol{B}$-site orbitals.

U$_{3d}$	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**
0.0	5.35	5.17	4.83	4.19	3.58	3.23
2.0	5.42	5.20	4.93	4.41	3.51	3.16
2.5	5.34	5.19	4.91	4.51	3.48	3.17
3.0	5.26	5.18	4.88	4.61	3.46	3.18
3.5	5.17	5.16	4.85	4.75	3.41	3.19
4.0	5.07	5.15	4.81	4.87	3.27	3.20
4.5	-	-	-	4.84	3.12	3.22

The Python[S2] code written in the org-mode[S1] syntax transcribed below was used to aggregate the data associated with the ”A = La, 4f” plot presented in the ”Effect of U on oxygen vacancy formation energy” portion of the DFT + U subsection of the Results and Discussion section, then generate the plot itself. The actual numerical values of the oxygen vacancy formation energy used to generate the plot are written at the conclusion of the code; they are taken from the output of the code, tabulated, and rounded. Note that data for many of these systems exists from U = 1.00 to 12.00 eV in 0.5 eV increments, thus ovu.U_value variables can be set in the data query to access different trends of constant U.

```
# DFT + U Data Aggregator and Plot Generator
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Asite = ’La’
corr = ’4f’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, U_value, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select ovu.U_value, c2.at2, ov.calc_quantity from Structure as s
inner join Symmetry as sym on s.struct=sym.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ovu.cID=Ov.cID
where sym.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and ov.correction=’U_4f’
and ov.mag=’none’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and (ovu.U_value = ’0’ or ovu.U_value = ’3’ or ovu.U_value = ’6’
or ovu.U_value = ’12’)
;’’’):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        U_value += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)
Sort_Bsite = {"Sc": 0, "Ti": 1, "V": 2, "Cr": 3, "Mn": 4, "Fe": 5, "Co": 6, "Ni": 7, "Cu": 8}

for i, j, k in datapts_dict:
    SortElist[i].append( (j,k) )
    SortElist[i].sort(key=lambda val: Sort_Bsite[val[0]])

num_Bsites = len(Bsites)
EBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    matchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        matchlist.append(SortElist[j][k][1])
    EBsiteMatch[j] = matchlist

print EBsiteMatch[0.0]
print EBsiteMatch[3.0]
print EBsiteMatch[6.0]
print EBsiteMatch[12.0]

x = [i for i in range( num_Bsites )]
ax = plt.gca()

plt.figure(figsize=(3,4))
plt.plot(x, EBsiteMatch[0.0], color = ’b’,
         linestyle = ’dashed’, linewidth = 1.5,
         dash_capstyle =’round’, dash_joinstyle = ’round’,
         label = r’U = 0’)
plt.plot(x, EBsiteMatch[3.0], color = ’r’,
         linestyle = ’dashed’, linewidth = 1.5,
         dash_capstyle =’round’, dash_joinstyle = ’round’,
         label = r’U = 3’)
plt.plot(x, EBsiteMatch[6.0], color = ’g’,
         linestyle = 'dashed', linewidth = 1.5,
         dash_capstyle ='round', dash_joinstyle = 'round',
         label = r'U = 6')
plt.plot(x, EBsiteMatch[12.0], color = 'k',
         linestyle = 'dashed', linewidth = 1.5,
         dash_capstyle ='round',
         dash_joinstyle ='round', label = r'U = 12')

plt.xlabel( 'B-site' )
xticks = Bsites
plt.xticks(x, xticks)
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'lower left', prop={'size':10})
plt.text(0.06, 0.85, 'A = {0}, {1}'.format(Asite,corr),
         style = 'italic', color = 'white', fontsize = 16,
    transform = ax.transAxes,
    bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})
plt.gcf().subplots_adjust(left=0.21)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('TestU_' + corr + 'A_A' + Asite + '.' + ext)
plt.clf()
```

```
[6.042962815, 5.787644577, 5.32388389, 4.795756344, 4.313559822,
 3.872861467, 3.425349129, 3.152068696, 3.045446285]
[6.03223401, 5.748761423, 5.307433465, 4.802272999, 4.332076628,
 3.89225215, 3.445824941, 3.171179257, 3.053963374]
[6.006022311, 5.720551319, 5.288903433, 4.79691575, 4.333331728,
 3.893817055, 3.448666363, 3.170907725, 3.050547933]
[5.963629536, 5.679288248, 5.255150507, 4.775574542, 4.318351925,
 3.879140915, 3.436247368, 3.153163375, 3.023325897]
```

Table S13: Oxygen vacancy formation energies (eV) of non-magnetic cubic unit cell LaBO$_3$ systems with U corrections perturbing 4f La orbitals.

U$_{4f}$	**V**	**Cr**	**Mn**	**Fe**	**Co**	**Ni**
0.0	5.32	4.80	4.31	3.87	3.43	3.15
3.0	5.31	4.80	4.33	3.89	3.45	3.17
6.0	5.29	4.80	4.33	3.89	3.45	3.17
12.0	5.26	4.78	4.32	3.88	3.44	3.15

3.7.3 Comparison to Experimental and Other First-principles Results

Limited experimental data concerning the oxygen vacancy formation energies of the LaBO$_3$ and SrBO$_3$ systems analyzed in this manuscript exist. In both Lee et al.[S22] and Pavone et al.[S23] oxygen vacancy formation energies of approximately 4.3, 5.1, and 2.2 eV are resolved for LaMnO$_3$, LaFeO$_3$, and LaCoO$_3$ (respectively) from several distinct references.[S24–S26] These energies are ordered via the list "Bsitelist" and transcribed into the list "E_MnFeCo" in the code below. As illustrated in Wang et al.,[S12] there are systematic and non-systematic contributions to errors in energetic trends comprised of metal oxide systems. Systematic errors, which can be observed in energetic trends as a constant shift of values along the axis of a dependent variable, result from an error that consistently and equivalently affects each system in a trend. These errors can be resolved by adding or subtracting constant values from an energetic trend. An example of a systematic error affecting a trend present in this paper can be observed when adding half of the overbonded O$_2$ diatomic molecule energy to each system in such an energetic trend. The correction offered by Wang et al.[S12] is written into the code below as "cederEO2_shift", while the variable amount added to this built-in correction is coded as "addedEO2_shift". On the other hand, non-systematic errors affect individual systems linked together via an energetic trend differently. In the manuscript, a primary example of non-systematic error – which observed the most significant differences in its effects on different systems within a trend – is found when accounting for electron-electron interaction error via the Hubbard U model.

Therefore, in an initial attempt to reconcile experimental results for aforementioned systems with their analogues calculated in the manuscript, data collected on $LaMnO_3$, $LaFeO_3$, and $LaCoO_3$ systems subjected to the Hubbard U model is compared to experimental results in the code below. This comparison is facilitated via a tunable constant parameter used to represent the systematic error in an energetic trend (”addedEO2_shift”), with non-systematic errors represented solely by the Hubbard U inclusive data collected from a database search.

```
# DFT + U Comparison to Experiment
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

Asite = ’La’
corr = ’3d’

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, U_value, Bsites, Evac = [], [], [], [], []
datapts_dict = {}

for row in db.execute(’’’
select ovu.U_value, c2.at2, ov.calc_quantity from Structure as s
inner join Symmetry as sym on s.struct=sym.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ovu.cID=Ov.cID
where sym.sym=’Pm-3m’
and s.morph=’perovskite’
and c1.at1=’La’
and (c2.at2=’Mn’ or c2.at2=’Fe’ or c2.at2=’Co’)
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
and (ovu.U_value = ’0’ or ovu.U_value = ’2’ or ovu.U_value = ’2.5’
or ovu.U_value = ’3’ or ovu.U_value = ’3.5’ or ovu.U_value = ’4’
or ovu.U_value = ’4.5’ or ovu.U_value = ’5’ or ovu.U_value = ’5.5’
or ovu.U_value = ’6’ or ovu.U_value = ’6.5’ or ovu.U_value = ’7’
or ovu.U_value = ’7.5’ or ovu.U_value = ’8’)
;'''):
    datapts_list = []
    a,b,c = row
    datapts_list.append( (a,b,c) )

    if a not in label_1:
        U_value += [a]

    if b not in label_2:
        Bsites += [b]

    label_1 += [a]
    label_2 += [b]
    Evac += [c]

    datapts_dict[row] = datapts_list

SortElist = defaultdict(list)

for i, j, k in datapts_dict:
    SortElist[j].append( (i,k) )
    SortElist[j].sort()

num_Bsites = len(Bsites)
EBsiteMatch = {}
UBsiteMatch = {}

for i,j in enumerate(SortElist.keys()):
    Ematchlist = []
    Umatchlist = []
    Bsiterange = len(SortElist[j])
    for k in range( len(SortElist[j]) ):
        Umatchlist.append( SortElist[j][k][0] )
        Ematchlist.append( SortElist[j][k][1] )
    UBsiteMatch[j] = Umatchlist
    EBsiteMatch[j] = Ematchlist

addedEO2_shift = 0.25
cederEO2_shift = 0.5*1.36
totalEO2_shift = addedEO2_shift + cederEO2_shift

```

```
Bsitelist = [’Mn’, ’Fe’, ’Co’]
E_MnFeCo = [4.3, 5.1, 2.2]
theory_energy_dict = {}

for i,a in enumerate(Bsitelist):
    expt_energy = E_MnFeCo[i]
    theory_energy_list = []

    for j in range( len(EBsiteMatch[a]) ):
        theory_energy = EBsiteMatch[a][j] + addedEO2_shift
        theory_energy_list.append(theory_energy)

    theory_energy_dict[a] = theory_energy_list

    checkU_counter = 0
    for j in range( len(theory_energy_list) - 1):
        if ( (theory_energy_list[j] <= expt_energy) and (expt_energy <= theory_energy_list[j+1]) or (
           (theory_energy_list[j] >= expt_energy) and (expt_energy >= theory_energy_list[j+1]) ) ):
            print "For system La" + str(Bsitelist[i]) + "O3, a value of U between "
            print str(UBsiteMatch[a][j]) + " and " + str(UBsiteMatch[a][j+1])
            print " matches experimental and theoretical energetics,"
            print " given a constant systematic correction of " + str(totalEO2_shift) + "\n"

            print "Energy(" + str(UBsiteMatch[a][j]) + ") = " + str(theory_energy_list[j])
            print "Energy(" + str(UBsiteMatch[a][j+1]) + ") = " + str(theory_energy_list[j+1])
            print "Energy(experiment) = " + str(expt_energy) + "\n"

            checkU_counter += 1

    if checkU_counter < 1:
        print "For a constant systematic correction of " + str(totalEO2_shift) + ","
        print "there exists no tested U value allowing experimental and theoretical"
        print "energetics to match for system La" + str(Bsitelist[i]) + "O3\n"

        print "Energy(experiment) = " + str(expt_energy) + "\n"

ax = plt.gca()
plt.figure(figsize=(3,4))
plt.plot(UBsiteMatch[’Mn’], theory_energy_dict[’Mn’], color = ’b’,
         linestyle = ’dashed’, linewidth = 1.5, dash_capstyle = ’round’,
         dash_joinstyle = ’round’, label = r’Mn’)
```

```python
plt.plot(UBsiteMatch['Fe'], theory_energy_dict['Fe'], color = 'r',
        linestyle = 'dashed', linewidth = 1.5, dash_capstyle = 'round',
        dash_joinstyle = 'round', label = r'Fe')
plt.plot(UBsiteMatch['Co'], theory_energy_dict['Co'], color = 'g',
        linestyle = 'dashed', linewidth = 1.5, dash_capstyle = 'round',
        dash_joinstyle = 'round', label = r'Co')
plt.plot([0,8], [E_MnFeCo[0],E_MnFeCo[0]], 'bo-', linewidth = 2.0, label = r'Mn expt.')
plt.plot([0,8], [E_MnFeCo[1],E_MnFeCo[1]], 'ro-', linewidth = 2.0, label = r'Fe expt.')
plt.plot([0,8], [E_MnFeCo[2],E_MnFeCo[2]], 'go-', linewidth = 2.0, label = r'Co expt.')

plt.xlabel( '$U$ (eV)' )
plt.ylabel('$\Delta$E$_{vac}$ (eV/O)')
plt.legend(loc = 'lower left', prop={'size':10})
plt.text(0.055, 0.85, 'A = {0}, {1}'.format(Asite,corr),
        style = 'italic', color = 'white', fontsize = 16,
    transform = ax.transAxes,
    bbox={'facecolor':'blue', 'alpha':0.5, 'pad':10})
plt.gcf().subplots_adjust(left=0.21)

for ext in ['png', 'pdf', 'eps']:
    plt.savefig('ExptU_' + corr + 'B_A' + Asite + '.' + ext)
plt.clf()
```

```
For a constant systematic correction of 0.93,
there exists no tested U value allowing experimental and theoretical
energetics to match for system LaMnO3


Energy(experiment) = 4.3


For system LaFeO3, a value of U between 3.5 and 4.0
 matches experimental and theoretical energetics,
 given a constant systematic correction of
0.93


Energy(3.5) = 4.998168464
Energy(4.0) = 5.124620474

Energy(experiment) = 5.1


For system LaFeO3, a value of U between 4.0 and 4.5

 matches experimental and theoretical energetics,

 given a constant systematic correction of

0.93


Energy(4.0) = 5.124620474

Energy(4.5) = 5.086116621

Energy(experiment) = 5.1


For system LaCoO3, a value of U between 7.5 and 8.0

 matches experimental and theoretical energetics,

 given a constant systematic correction of

0.93


Energy(7.5) = 2.21816523

Energy(8.0) = 2.144169365

Energy(experiment) = 2.2
```

Note that $LaMnO_3$ does not have energetic data for the range U_{3d} = 4.5-5.5 eV. As shown via manipulation of the constant term above (”addedEO2_shift”), there exists no constant correction of systematic error such that tested values of U can match experimental results simultaneously. When using other first-principles data from Lee et al. (set E_MnFeCo = [3.7, 4.5, 2.8]),[S22] the same conclusion is reached. This largely results from the consideration of solely a single form of non-systematic error in the above analysis, namely one that can be corrected by accounting for electron-electron interaction. If other non-systematic effects were considered – such as the possibly different crystallographic symmetries of systems characterizing the experimental measurements adapted to the code above – with the application of the Hubbard U method simultaneously, numerically precise first-principles calculations of vacancy formation energies may be deducible. The consideration of crystallographic symmetry as a possible candidate for additional non-systematic effects evolves from considering the nearly systematic constant shifts of vacancy formation energy with changes in crystallographic symmetry in the manuscript, in which constant shifts were observed when entire energetic trends possessed a single shared crystal structure. Given that different perovskite compositions are generally stable in different crystal structures across energetic trends, errors associated with crystal structure would, as a result, be non-systematic over entire trends.

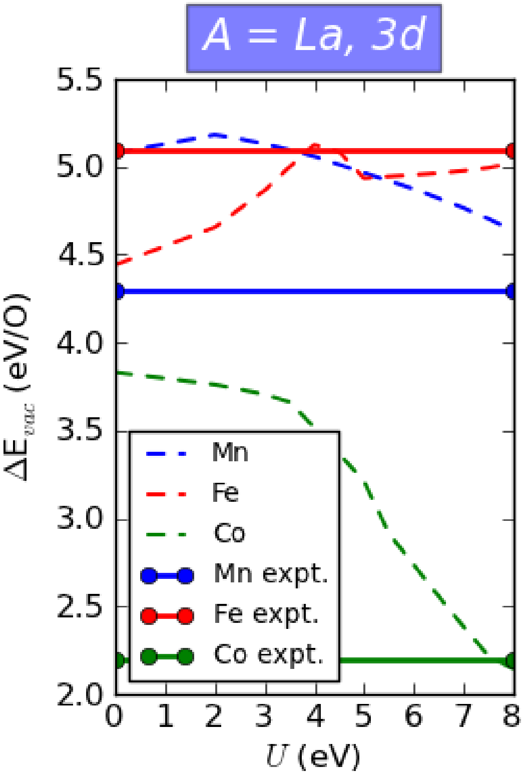

Figure S13: Comparison of oxygen vacancy formation energies from Hubbard U calculations (dashed lines) of $LaMnO_3$, $LaFeO_3$, and $LaCoO_3$ with experimental results (solid lines) from various sources.

3.7.4 Linear Response: Resolving Energetic Ground States and Equilibrium Cell Volumes in Quantum Espresso

For the Linear Response subsection of the Results and Discussion section of this article, the Quantum Espresso (QE) software package was necessarily used instead of VASP (VASP does not have a built-in linear response Hubbard U calculator) to calculate values of U solely from first-principles and not from fitting to any experimental data, as this is consistent with the intent of constructing a thermodynamic framework capable of predicting ΔE_{vac} ordering across large sets of $AB\mathrm{O}_3$ (and, more broadly, polymorphic metal oxide) systems. Therefore, the ground state energetics and equilibrium cell volumes of all six ($\mathrm{LaVO_3}$, $\mathrm{LaCrO_3}$, $\mathrm{LaMnO_3}$, $\mathrm{LaFeO_3}$, $\mathrm{SrCrO_3}$, and $\mathrm{SrMnO_3}$) systems for which U parameters were resolved were recalculated in QE to generate suitable input structures for the linear response calculations themselves. Using the Birch-Murnaghan equation of state applied previously, equilibrium structures for linear response calculations were derived from sample input files similar to that shown below, which was used for a paramagnetic (PM) $\mathrm{LaFeO_3}$ system calculated at U = 0 eV and a lattice constant of 7.30 Bohr (3.86 $\AA$). Calculations involving the ferromagnetic (FM) phase simply require modification of the parameter ”starting_magnetization(1)”, such that it equals 0.5 rather than 0.0. In order to run these calculations, replace the paths specified in pseudo_dir and outdir such that they point to directories containing the pseudopotential files specified in the ATOMIC_SPECIES section of the input file and the location of calculation output, respectively.

```
# QE Input File, Structural Relaxation

&CONTROL
 calculation = "relax",
 verbosity = "low",
 restart_mode = "from_scratch",
 pseudo_dir = "../",
 outdir = "./",
 title = "GGA + U, Test Lattice",
 prefix = "ABO3_LatC",
 nstep = 100,
 wf_collect = .false.
 forc_conv_thr = 5.0E-3
 etot_conv_thr = 1.0E-6
 /
&SYSTEM
 ibrav = 1
 nat = 5
 ntyp = 3
 ecutwfc = 50.0
 ecutrho = 600.0
 nspin = 2
 starting_magnetization(1) = 0.0
 occupations = "smearing",
 smearing = "gaussian",
 degauss = 0.01
 lda_plus_u = .true.
 lda_plus_u_kind = 0
 celldm(1) = 7.30
 Hubbard_U(1) = 1D-10
 Hubbard_U(2) = 1D-10
 Hubbard_U(3) = 1D-10
 /
&ELECTRONS
 electron_maxstep = 100
 conv_thr = 1D-9
 diagonalization = "david",
 diago_thr_init = 1D-2
 diago_full_acc = .false.
 startingwfc = "atomic+random",
 mixing_mode = "plain",
 mixing_beta = 0.3
 /
&IONS
 ion_dynamics = "bfgs",
 upscale = 1000
 /
ATOMIC_SPECIES
Fe   1.0   Fe.pbe-sp-van.UPF
La   1.0   La.pbe-nsp-van.UPF
O   1.0   O.pbe-rrkjus.UPF
```

```

ATOMIC_POSITIONS (crystal)
Fe  0.0  0.0  0.0
La  0.5  0.5  0.5
O   0.0  0.0  0.5
O   0.0  0.5  0.0
O   0.5  0.0  0.0

K_POINTS (automatic)
 8 8 8 0 0 0
```

By comparing the energetic results produced by VASP and QE within the context of selecting structural data for linear response U calculations, key differences between how structural data in VASP and QE are affected by magnetic state and related input parameters were manifested. In order to discern these differences between VASP and QE calculations, three different types of input files, each of which models the features of a particular magnetic state of interest, were provided to these two calculators. Table S14 shown below categorizes structural data for pertinent systems according to both the magnetic state modeled by the data and the software package used to calculate it. In particular, non-magnetic (NM) calculations were performed and tabulated for VASP calculations by not incorporating spin polarization (ISPIN = 1 in VASP, nspin = 1 in QE) during structural relaxation, paramagnetic (PM) calculations were completed and tabulated for both VASP and QE by incorporating collinear spin-polarization (ISPIN = 2 in VASP, nspin = 2 in QE) and no initial starting magnetic moments (MAGMOM = 0.0×5 for a $1 \times 1 \times 1$ cubic unit cell in VASP and starting_magnetization(1) = 0 for the Fe atom in QE), and ferromagnetic (FM) calculations were done for QE calculations with spin-polarization and initial starting magnetic moments greater than 0 applied to B-site cations.

The effects of changing between different sets of input parameters (as was detailed previously) on the equilibrium cell volumes (and thus cubic lattice constants, a) of systems modeled identically across VASP and QE can be distinctly different across both calculators. As shown in Table S14 below, equilibrium cell volumes are almost uniformly increased by solely considering spin-polarization without magnetic moments (PM case) in VASP, while cell volume is only affected after considering both spin-polarization and non-zero starting magnetic moments in QE calculations. When analyzing solely PM systems across both VASP and QE calculators and applying values of U resolved in QE to energetic calculations in VASP, the equilibrium cell volumes of the ground state structures resolved by both calculators will be distinctly different. As will be shown in the subsequent section, the contribution of a significant value of U to calculations using either VASP or QE magnifies the difference between the ground state structures resolved in both cases. Considering the effect of changing the positions of relaxed atomic coordinates – which can be directly related to uniform expansion and contraction (or tensile and compressive strain) of equilibrium cell volume – on the values of U derived from linear response and thus the energetics of studied systems,[S27] the differences between how ground state structures in QE and VASP are affected by the input parameters held equivalent between them justifies further study of the effects of equilbrium cell volume on calculated U values. Despite the similarities between the structural data of PM QE and NM VASP calculations – as well as those between FM QE and PM VASP calculations – shown in Table S14, the equilibrium cell volume expansion observed for a given system in one calculator cannot necessarily be applied to the same system analyzed in a different calculator, as neither the absolute values of equilibrium cell volumes nor the amounts of cell volume expansion are shared across all systems between QE and VASP calculations and resolving a more precise level of uncertainty introduced by the difference in equilibrium cell volume expansion helps further verify ΔE_{vac} ordering. The code generating the results in this table is also reproduced below:

```
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, label_3, ABsites, mag, calc, EqVol = [], [], [], [], [], [], []
```

```

datapts_dict = {}

conv_V = 0.529177249
frac_V = 1./3.


# 1st Query: QE results, all
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,ov.calc_quantity from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’none’
and (ov.mag=’PM’ or ov.mag=’FM’)
and ov.orientation=’bulk’
and ov.calculator=’QE’
;’’’):
    datapts_list = []
    a,b,c,d,e = row
    series = a,b,c,d,e

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    EqVol += [e]

    datapts_dict[a,b,c,d] = datapts_list



label_1, label_2, label_3, ABsites, mag, calc, EqVol = [], [], [], [], [], [], []

# 2nd Query: VASP results, NM and FM
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,rdev.VNoVac_fit from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_EV as ovev on ov.cID=ovev.cID
inner join RawData_EV_NoVac as rdev on ovev.ENoVac_fit=rdev.ENoVac_fit
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’none’
and (ov.mag=’none’ or ov.mag=’FM’)
and ov.orientation=’bulk’
and ov.calculator=’VASP’
;’’’):
    datapts_list = []
    a,b,c,d,e = row
    series = a,b,c,d,e

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    EqVol += [e]

    datapts_dict[a,b,c,d] = datapts_list


# 3rd Query: VASP results, PM (U = 0 eV)
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,rdev.VNoVac_fit from Structure as s
inner join Symmetry as sy on sy.struct=s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ov.cID=ovu.cID
inner join RawData_EV_NoVac as rdev on ovu.ENoVac_fit=rdev.ENoVac_fit
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’U_3d’
and ov.mag=’PM’
and ovu.U_value=’0.0’
and ov.orientation=’bulk’
and ov.calculator=’VASP’
;’’’):
    datapts_list = []
    a,b,c,d,e = row
    series = a,b,c,d,e

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    EqVol += [e]

    datapts_dict[a,b,c,d] = datapts_list


EBsiteMatch = {}

for i,j in enumerate( datapts_dict.keys() ):
    Asite = datapts_dict[j][0]
    Bsite = datapts_dict[j][1]
    Mag = datapts_dict[j][2]
    Calc = datapts_dict[j][3]

    EBsiteMatch[Asite,Bsite,Mag,Calc] = datapts_dict[j]

    if Calc == ’QE’:
        EBsiteMatch[Asite,Bsite,Mag,Calc][-1] = EBsiteMatch[Asite,Bsite,Mag,Calc][-1] * (conv_V**(1/frac_V))

    LatC = ( EBsiteMatch[Asite,Bsite,Mag,Calc][-1] )**(frac_V)
    EBsiteMatch[Asite,Bsite,Mag,Calc].append(LatC)


print ( EBsiteMatch[’La’,’V’,’PM’,’QE’][-2],
        EBsiteMatch[’La’,’V’,’none’,’VASP’][-2],
        EBsiteMatch[’La’,’V’,’PM’,’QE’][-1],
        EBsiteMatch[’La’,’V’,’none’,’VASP’][-1],
        EBsiteMatch[’La’,’V’,’FM’,’QE’][-2],
        EBsiteMatch[’La’,’V’,’PM’,’VASP’][-2],
        EBsiteMatch[’La’,’V’,’FM’,’QE’][-1],
        EBsiteMatch[’La’,’V’,’PM’,’VASP’][-1])

print ( EBsiteMatch[’La’,’Cr’,’PM’,’QE’][-2],
        EBsiteMatch[’La’,’Cr’,’none’,’VASP’][-2],
        EBsiteMatch[’La’,’Cr’,’PM’,’QE’][-1],
        EBsiteMatch[’La’,’Cr’,’none’,’VASP’][-1],
        EBsiteMatch[’La’,’Cr’,’FM’,’QE’][-2],
        EBsiteMatch[’La’,’Cr’,’PM’,’VASP’][-2],
        EBsiteMatch[’La’,’Cr’,’FM’,’QE’][-1],
        EBsiteMatch[’La’,’Cr’,’PM’,’VASP’][-1])

print ( EBsiteMatch[’La’,’Mn’,’PM’,’QE’][-2],
        EBsiteMatch[’La’,’Mn’,’none’,’VASP’][-2],
        EBsiteMatch[’La’,’Mn’,’PM’,’QE’][-1],
        EBsiteMatch[’La’,’Mn’,’none’,’VASP’][-1],
        EBsiteMatch[’La’,’Mn’,’FM’,’QE’][-2],
        EBsiteMatch[’La’,’Mn’,’PM’,’VASP’][-2],
        EBsiteMatch[’La’,’Mn’,’FM’,’QE’][-1],
        EBsiteMatch[’La’,’Mn’,’PM’,’VASP’][-1])

print ( EBsiteMatch[’La’,’Fe’,’PM’,’QE’][-2],
        EBsiteMatch[’La’,’Fe’,’none’,’VASP’][-2],
        EBsiteMatch[’La’,’Fe’,’PM’,’QE’][-1],
        EBsiteMatch[’La’,’Fe’,’none’,’VASP’][-1],
        EBsiteMatch[’La’,’Fe’,’FM’,’QE’][-2],
        EBsiteMatch[’La’,’Fe’,’PM’,’VASP’][-2],
        EBsiteMatch[’La’,’Fe’,’FM’,’QE’][-1],
        EBsiteMatch[’La’,’Fe’,’PM’,’VASP’][-1])

print ( EBsiteMatch[’Sr’,’Cr’,’PM’,’QE’][-2],
        EBsiteMatch[’Sr’,’Cr’,’none’,’VASP’][-2],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’QE’][-1],
        EBsiteMatch[’Sr’,’Cr’,’none’,’VASP’][-1],
        EBsiteMatch[’Sr’,’Cr’,’FM’,’QE’][-2],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’VASP’][-2],
        EBsiteMatch[’Sr’,’Cr’,’FM’,’QE’][-1],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’VASP’][-1])

print ( EBsiteMatch[’Sr’,’Mn’,’PM’,’QE’][-2],
        EBsiteMatch[’Sr’,’Mn’,’none’,’VASP’][-2],
        EBsiteMatch[’Sr’,’Mn’,’PM’,’QE’][-1],
        EBsiteMatch[’Sr’,’Mn’,’none’,’VASP’][-1],
        EBsiteMatch[’Sr’,’Mn’,’FM’,’QE’][-2],
        EBsiteMatch[’Sr’,’Mn’,’PM’,’VASP’][-2],
        EBsiteMatch[’Sr’,’Mn’,’FM’,’QE’][-1],
        EBsiteMatch[’Sr’,’Mn’,’PM’,’VASP’][-1])
```

Table S14: Comparison of equilibrium cell volumes (Å^3) and cubic lattice constants (a, Å) between PM QE and NM VASP calculations (first four columns) and between FM QE and PM VASP calculations (last four columns), grouped based on how changes in equilibrium cell volume are observed.

Perovskite	**Vol. QE**	**Vol. VASP**	***a* QE**	***a* VASP**	**Vol. QE**	**Vol. VASP**	***a* QE**	***a* VASP**
$LaVO_3$	58.17	58.93	3.87	3.88	59.67	59.09	3.91	3.89
$LaMnO_3$	55.42	56.24	3.81	3.83	59.38	59.38	3.90	3.90
$LaFeO_3$	54.90	55.69	3.80	3.82	57.60	58.24	3.86	3.88
$SrCrO_3$	55.34	55.90	3.81	3.82	56.72	57.28	3.84	3.85
$SrMnO_3$	54.60	55.11	3.79	3.81	56.53	56.83	3.84	3.84

3.7.5 Linear Response: Effects of U on Equilibrium Cell Volume in Quantum Espresso vs. VASP

As mentioned in the previous section, the addition of a U parameter of significant magnitude can exacerbate the differences in equilibrium cell volume between sets of $AB\mathrm{O}_3$ systems modeled using QE and VASP, as adding U values to sets of systems with cell volumes already affected differently by the same magnetic state can unevenly increase cell volume in one set versus another. In order to evaluate the magnitude of this effect, the effect of increasing U from 0.0 eV to 7.0 eV was evaluated over several systems, using QE input files mirroring the input reproduced below. The value of $U = 7.0$ eV used to evaluate the effect of U on different NM and PM systems in QE and VASP was selected due to trial-and-error testing of the linear response method and previous results achieved in similar systems:[S28] In order to run these calculations, replace the paths specified in pseudo_dir and outdir such that they point to directories containing the pseudopotential files specified in the ATOMIC_SPECIES section of the input file and the location of calculation output, respectively.

```
# QE Input File, Evaluating Structural Effects of U Contributions

&CONTROL
```

```
 calculation = "relax",
 verbosity = "low",
 restart_mode = "from_scratch",
 pseudo_dir = "../",
 outdir = "./",
 title = "GGA + U, Test Lattice",
 prefix = "ABO3_LatC",
 nstep = 100,
 wf_collect = .false.
 forc_conv_thr = 5.0E-3
 etot_conv_thr = 1.0E-6
 /
&SYSTEM
 ibrav = 1
 celldm(1) = 7.00
 nat = 5
 ntyp = 3
 ecutwfc = 30.0
 ecutrho = 300.0
 nspin = 2
 starting_magnetization(1) = 0.0
 occupations = "smearing",
 smearing = "gaussian",
 degauss = 0.01
 lda_plus_u = .true.
 Hubbard_U(1) = 1D-10
 /
&ELECTRONS
 electron_maxstep = 100
 conv_thr = 1D-9
 diagonalization = "david",
 diago_thr_init = 1D-2
 diago_full_acc = .false.
 startingwfc = "atomic+random",
 mixing_mode = "plain",
 mixing_beta = 0.2
 /
&IONS
 ion_dynamics = "bfgs",
 upscale = 1000
 /
```

```
ATOMIC_SPECIES
Mn   1.0   Mn.pbe-sp-van.UPF
Sr   1.0   Sr.pbe-nsp-van.UPF
O   1.0   O.pbe-rrkjus.UPF

ATOMIC_POSITIONS (crystal)
Mn  0.0  0.0  0.0
Sr  0.5  0.5  0.5
O   0.0  0.0  0.5
O   0.0  0.5  0.0
O   0.5  0.0  0.0

K_POINTS (automatic)
 6 6 6 0 0 0
```

The results of this analysis are reproduced in the table below. Table S15 reveals that the differences in lattice constants (i.e.: cell volumes) between PM QE calculations of all systems studied is minor or absent, while also illustrating the significant difference between NM and PM VASP calculation lattice constants. In almost all cases, introducing a U parameter of 7.0 eV to the $3d$ B-sites of studied perovskites further increased the lattice constants of PM, albeit by quantities that depended on $AB\mathrm{O}_3$ composition. Therefore, in order to account for the entire possible impact of equilibrium cell volume changes on calculated linear response U values across the materials studied, both the effects of changing from QE to VASP and from input values of $U = 0$ eV to $U = 7$ eV on cell volume are considered. Over the three materials sampled using this method, alluded to in the article as $\mathrm{LaCrO_3}$, $\mathrm{LaMnO_3}$, and $\mathrm{LaFeO_3}$, the overall respective lattice constant increases of 3.83 to 3.95 Å, 3.81 to 3.94 Å, and 3.80 to 3.92 Å were observed. In the most extreme $\mathrm{LaBO_3}$ case, $\mathrm{LaMnO_3}$, this increase in lattice constant approximately corresponds to a uniform tensile strain of 3.5%, constituting the largest expected distortion of the lattice constant with respect to the GGA QE ground state. This level of strain forms the basis for the strained U calculations performed in the article itself. The code used to generate Table S15 is also reproduced below:

```
import matplotlib.pyplot as plt
import sqlite3
from collections import defaultdict

conv_V = 0.529177249
frac_V = 1./3.

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, label_3, label_4, ABsites, mag, calc, Uval, EqVol = [], [], [], [], [], [], [], [], []

datapts_dict = {}


# 1st Query: QE, all
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,ovu.U_value,ovu.VNoVac_fit from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_TestU as ovu on ov.cID=ovu.cID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’U_3d’
and ov.mag=’PM’
and (ovu.U_value=’1D-10’ or ovu.U_value=’7.0’)
and ov.orientation=’bulk’
and ov.calculator=’QE’
;’’’):
    datapts_list = []
    a,b,c,d,e,f = row
    series = a,b,c,d,e,f

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    if e not in label_4:
        Uval += [e]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    label_4 += [e]
    EqVol += [f]

    datapts_dict[a,b,c,d,e] = datapts_list


label_1, label_2, label_3, label_4, ABsites, mag, calc, Uval, EqVol = [], [], [], [], [], [], [], [], []

# 2nd query: VASP, PM
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,ovu.U_value,rdev.VNoVac_fit from Structure as s
inner join Symmetry as sy on sy.struct = s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_OrderU as ovu on ov.cID=ovu.cID
inner join RawData_EV_NoVac as rdev on ovu.ENoVac_fit=rdev.ENoVac_fit
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’U_3d’
and ov.mag=’PM’
and (ovu.U_value=’0.0’ or ovu.U_value=’7.0’)
and ov.orientation=’bulk’
and ov.calculator=’VASP’
```

```
;’’’):
    datapts_list = []
    a,b,c,d,e,f = row
    series = a,b,c,d,e,f

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    if e not in label_4:
        Uval += [e]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    label_4 += [e]
    EqVol += [f]

    datapts_dict[a,b,c,d,e] = datapts_list


label_1, label_2, label_3, label_4, ABsites, mag, calc, Uval, EqVol = [], [], [], [], [], [], [], [], []

# 3rd Query: VASP, NM
for row in db.execute(’’’
select c1.at1,c2.at2,ov.mag,ov.calculator,rdev.VNoVac_fit from Structure as s
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_EV as ovev on ov.cID=ovev.cID
inner join RawData_EV_NoVac as rdev on ovev.ENoVac_fit=rdev.ENoVac_fit
where s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=5
and ov.correction=’none’
and ov.mag=’none’
and ov.orientation=’bulk’
;’’’):
    datapts_list = []
    e = 0.0
    a,b,c,d,f = row
    series = a,b,c,d,e,f

    for z in series:
        datapts_list.append(z)

    if (a,b) not in label_1:
        ABsites += [a,b]

    if c not in label_2:
        mag += [c]

    if d not in label_3:
        calc += [d]

    if e not in label_4:
        Uval += [e]

    label_1 += [a,b]
    label_2 += [c]
    label_3 += [d]
    label_4 += [e]
    EqVol += [f]

    datapts_dict[a,b,c,d,e] = datapts_list


EBsiteMatch = {}

for i,j in enumerate( datapts_dict.keys() ):
    Asite = datapts_dict[j][0]
    Bsite = datapts_dict[j][1]
    Mag = datapts_dict[j][2]
    Calc = datapts_dict[j][3]
    Uvalue = datapts_dict[j][4]

    EBsiteMatch[Asite,Bsite,Mag,Calc,Uvalue] = datapts_dict[j]

    if Calc == ’QE’:
        EBsiteMatch[Asite,Bsite,Mag,Calc,Uvalue][-1] = (EBsiteMatch[Asite,Bsite,
                                                                   Mag,Calc,Uvalue][-1]
                                                        * (conv_V**(1/frac_V)))

    EBsiteMatch[Asite,Bsite,Mag,Calc,Uvalue][-1] = ( EBsiteMatch[Asite,Bsite,Mag,
                                                                 Calc,Uvalue][-1] )**(frac_V)

print ( EBsiteMatch[’La’,’V’,’PM’,’QE’,’1D-10’][-1],
        EBsiteMatch[’La’,’V’,’PM’,’QE’,7.0][-1],
        EBsiteMatch[’La’,’V’,’none’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’V’,’PM’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’V’,’PM’,’VASP’,7.0][-1])

print ( EBsiteMatch[’La’,’Cr’,’PM’,’QE’,’1D-10’][-1],
        EBsiteMatch[’La’,’Cr’,’PM’,’QE’,7.0][-1],
        EBsiteMatch[’La’,’Cr’,’none’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Cr’,’PM’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Cr’,’PM’,’VASP’,7.0][-1])

print ( EBsiteMatch[’La’,’Mn’,’PM’,’QE’,’1D-10’][-1],
        EBsiteMatch[’La’,’Mn’,’PM’,’QE’,7.0][-1],
        EBsiteMatch[’La’,’Mn’,’none’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Mn’,’PM’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Mn’,’PM’,’VASP’,7.0][-1])

print ( EBsiteMatch[’La’,’Fe’,’PM’,’QE’,’1D-10’][-1],
        EBsiteMatch[’La’,’Fe’,’PM’,’QE’,7.0][-1],
        EBsiteMatch[’La’,’Fe’,’none’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Fe’,’PM’,’VASP’,0.0][-1],
        EBsiteMatch[’La’,’Fe’,’PM’,’VASP’,7.0][-1])

print ( EBsiteMatch[’Sr’,’Cr’,’PM’,’QE’,’1D-10’][-1],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’QE’,7.0][-1],
        EBsiteMatch[’Sr’,’Cr’,’none’,’VASP’,0.0][-1],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’VASP’,0.0][-1],
        EBsiteMatch[’Sr’,’Cr’,’PM’,’VASP’,7.0][-1])

print (EBsiteMatch[’Sr’,’Mn’,’PM’,’QE’,’1D-10’][-1],
       EBsiteMatch[’Sr’,’Mn’,’PM’,’QE’,7.0][-1],
       EBsiteMatch[’Sr’,’Mn’,’none’,’VASP’,0.0][-1],
       EBsiteMatch[’Sr’,’Mn’,’PM’,’VASP’,0.0][-1],
       EBsiteMatch[’Sr’,’Mn’,’PM’,’VASP’,7.0][-1])
```

Table S15: Equilibrium cubic lattice constants (Å) or calculations performed at U = 0.0 eV and 7.0 eV in QE (PM states) and VASP (NM and PM states). Note that the first column in the table below is analogous to the third column in Table S14, though slightly different results are achieved due to the differences in the input files used to achieve them. The second and fourth columns below are identical to the fourth and eight columns, respectively, in Table S14.

Perovskite	a QE ($U = 0$)	a QE ($U = 7$)	a VASP NM ($U = 0$)	a VASP PM ($U = 0$)	a VASP PM ($U = 7$)
$LaVO_3$	3.88	3.89	3.89	3.89	4.00
$LaCrO_3$	3.83	3.83	3.85	3.90	3.95
$LaMnO_3$	3.81	3.82	3.83	3.90	3.96
$LaFeO_3$	3.80	3.82	3.82	3.88	3.92
$SrCrO_3$	3.81	3.80	3.82	3.85	3.99
$SrMnO_3$	3.79	3.82	3.81	3.84	3.96

3.7.6 Linear Response: Sample Calculation

A sample input file used for the initial perturbation calculation of the linear response U value of $LaVO_3$, which specifies all of the parameters used in all other linear response calculations, is reproduced below. In order to run these calculations, replace the paths specified in pseudo_dir and outdir such that they point to directories containing the pseudopotential files specified in the ATOMIC_SPECIES section of the input file and the location of calculation output, respectively.

```
# QE Input File, Linear Response Calculation

&CONTROL
 calculation = "scf",
 verbosity = "high",
 restart_mode = "from_scratch",
 disk_io = "high",
 pseudo_dir = "../../",
 outdir = "./",
 title = "GGA + U, U = 1D-40, init",
 prefix = "VLaO_U0",
 wf_collect = .false.
 /
&SYSTEM
 ibrav = 1
 celldm(1) = 14.671
 nat = 40
 ntyp = 4
 nspin = 2
 ecutwfc = 50.0
 ecutrho = 600.0
 starting_magnetization(1) = 0.0
 starting_magnetization(2) = 0.0
 occupations = "smearing",
 smearing = "gaussian",
 degauss = 0.035
 nosym = .false.
 lda_plus_U = .true.
 lda_plus_U_kind = 0
 U_projection_type = "atomic",
 Hubbard_U(1) = 1D-40
 Hubbard_U(2) = 1D-40
 Hubbard_U(3) = 1D-40
 Hubbard_U(4) = 1D-40
 /
&ELECTRONS
 electron_maxstep = 150
 conv_thr = 1D-8
 diagonalization = "david",
 diago_thr_init = 1D-2
 diago_full_acc = .false.
 startingpot = "atomic",
 startingwfc = "atomic+random",
 mixing_mode = "plain",
 mixing_beta = 0.1
 mixing_ndim = 8
 /
ATOMIC_SPECIES
V1   1.0   V.pbe-sp-van.UPF
V2   1.0   V.pbe-sp-van.UPF
```

```
La1   1.0   La.pbe-nsp-van.UPF
O1   1.0   O.pbe-rrkjus.UPF

ATOMIC_POSITIONS (crystal)
V1  0.0  0.0  0.0
V2  0.5  0.0  0.0
V2  0.0  0.5  0.0
V2  0.0  0.0  0.5
V2  0.5  0.5  0.0
V2  0.5  0.0  0.5
V2  0.0  0.5  0.5
V2  0.5  0.5  0.5
La1  0.75  0.75  0.75
La1  0.25  0.75  0.75
La1  0.75  0.25  0.75
La1  0.75  0.75  0.25
La1  0.25  0.25  0.75
La1  0.25  0.75  0.25
La1  0.75  0.25  0.25
La1  0.25  0.25  0.25
O1  0.25  0.25  0.0
O1  0.75  0.25  0.0
O1  0.25  0.75  0.0
O1  0.75  0.75  0.0
O1  0.25  0.25  0.5
O1  0.75  0.25  0.5
O1  0.25  0.75  0.5
O1  0.75  0.75  0.5
O1  0.25  0.0  0.25
O1  0.75  0.0  0.25
O1  0.25  0.0  0.75
O1  0.75  0.0  0.75
O1  0.25  0.5  0.25
O1  0.75  0.5  0.25
O1  0.25  0.5  0.75
O1  0.75  0.5  0.75
O1  0.0  0.25  0.25
O1  0.0  0.75  0.25
O1  0.0  0.25  0.75
O1  0.0  0.75  0.75
O1  0.5  0.25  0.25
```

```
O1  0.5  0.75  0.25
O1  0.5  0.25  0.75
O1  0.5  0.75  0.75

K_POINTS (automatic)
 8 8 8 0 0 0
```

3.7.7 Linear Response: Effects of Spreading on U and $\boldsymbol{B}$-site Atomic Perturbations Used to Calculate Finalized U Values

In determining the suitability of calculated U values, several factors are considered, namely the intersection of the initial (χ_0) and final (χ) response functions at a level of no orbital perturbation ($\alpha = 0.0$ eV) and the linearity of both response functions.[S29] In particular, the latter criterion is largely impacted by the magnitude of the Gaussian spreading (degauss) applied to the perturbed orbital occupations. In the case where an insufficient spreading is applied, the initial response observes parabolic behavior, generally leading to underestimation of values of U due to the negative curvature generally found in these systems when low spreading is applied. However, applying excessive spreading leads to slightly improved linearity in initial responses albeit excessive reduction of χ_0 slopes; given slopes of χ remain constant, this reduction would lead to underestimation of U as well. The magnitude of this effect is best demonstrated via $SrCrO_3$, which, upon reducing spreading from 0.04 to 0.035 and then to 0.030, experienced an increase in its linear response derived U value from 5.5, 6.0, and 6.6 eV while largely maintaining its linearity at both values of spreading. Reducing the spreading further dramatically increased the difference between χ_0 and χ at zero perturbation, causing the responses to intersect at a level of perturbation distinctly different from 0.

Therefore, an optimal value of spreading will account both for these effects that lead to U parameter underestimation by producing responses that are both well-fitted to linear functions and have linear slopes that are as high as possible. Through trial-and-error, the criteria determining the suitability of calculated U values in this study state that the difference between χ and χ_0 at $\alpha = 0.0$ eV is approximately 0.001 eV or less, that the R^2 of χ_0 over the range α = [-0.15, 0.15] eV be approximately equal to 0.999 or greater, and that the R^2 also be approximately equal to or less than 0.9999. These criteria were met by each calculation performed in the article, as shown in Table S16 below. All U values were derived via the response matrix calculator provided by Dr. Matteo Cococcioni in the Lab section at the ”Summer school on Materials modeling from first principles: theory and practice, ICMR, University of California at Santa Barbara”, entitled ”LSDA, collinear and noncollinear magnetism; spin-orbit coupling; DFT+U”. The code used to generate this table is also reproduced below:

```
import matplotlib.pyplot as plt
import sqlite3
from collections import OrderedDict
from numpy import array,ones
from scipy import stats

db = sqlite3.connect(’EOCCS_data.sqlite’)

label_1, label_2, label_3 = [], [], []

Asites, Bsites, LatC = [], [], []

datapts_dict = {}

for row in db.execute(’’’
select c1.at1,c2.at2,celldm1,degauss,ovl.chi0_m150,ovl.chi0_m100,
ovl.chi0_m050,ovl.chi0_000,ovl.chi0_050,ovl.chi0_100,ovl.chi0_150,
ovl.chi1_000 from Structure as s
inner join Symmetry as sy on sy.struct=s.struct
inner join Composition1 as c1 on s.sID=c1.sID
inner join Composition2 as c2 on c1.mID=c2.mID
inner join OVac_FormE as ov on c1.mID=ov.mID
inner join OVac_LRCalc as ovl on ov.cID=ovl.cID
where sy.sym=’Pm-3m’
and s.morph=’perovskite’
and ((c1.at1=’La’ and (c2.at2=’V’ or c2.at2=’Cr’ or c2.at2=’Mn’ or c2.at2=’Fe’))
or  (c1.at1=’Sr’ and (c2.at2=’Cr’ or c2.at2=’Mn’)))
and ov.num_atoms=40
and ov.correction=’U_3d’
and ov.mag=’PM’
and ov.orientation=’bulk’
and ov.calculator=’QE’
;’’’):
    datapts_list = []
    a,b,c,d,e,f,g,h,i,j,k,l = row
    series = a,b,c,d,e,f,g,h,i,j,k,l

    for z in series:
        datapts_list.append(z)

    if a not in label_1:
        Asites += [a]

    if b not in label_2:
        Bsites += [b]

    if c not in label_3:
        LatC += [c]

    label_1 += [a]
    label_2 += [b]
    label_3 += [c]

    datapts_dict[a,b,c,d] = datapts_list



EBsiteMatch = {}

for i,j in enumerate( datapts_dict.keys() ):
    Asite = datapts_dict[j][0]
    Bsite = datapts_dict[j][1]
    LatC = datapts_dict[j][2]
    Spread = datapts_dict[j][3]

    EBsiteMatch[Asite,Bsite,LatC,Spread] = datapts_dict[j]
    EBsiteMatch[Asite,Bsite,LatC,Spread][-1] = (EBsiteMatch[Asite,Bsite,LatC,Spread][-1]
                                                - EBsiteMatch[Asite,Bsite,LatC,Spread][-5])

```

```

pturbs = [0.15, 0.10, 0.05, 0.00, -0.05, -0.10, -0.15]
r2list = []

for i,j in enumerate( datapts_dict.keys() ):
    linalgLRlist = []

    for k in range( len(pturbs) ):
        linalgLRlist.append( EBsiteMatch[j][-2-k] )

    slope, intercept, r_value, p_value, std_err = stats.linregress(linalgLRlist,pturbs)
    r_sq = r_value**2
    EBsiteMatch[j].append(r_sq)
    r2list.append( r_sq )


orderLR = sorted(datapts_dict.keys())

strainlist = []
for i in orderLR:
    for j in orderLR:
        if ((EBsiteMatch[i][0] == EBsiteMatch[j][0] )
            and ( EBsiteMatch[i][1] == EBsiteMatch[j][1] )
            and ( EBsiteMatch[j][2] >= EBsiteMatch[i][2] )
            and ( EBsiteMatch[j] not in strainlist)):
            strain = (EBsiteMatch[j][2] - EBsiteMatch[i][2]) / EBsiteMatch[i][2]
            strainlist.append(EBsiteMatch[j])
            EBsiteMatch[j].append(strain)

print EBsiteMatch[’La’, ’V’, 14.671, 0.035]
print EBsiteMatch[’La’, ’Cr’, 14.526, 0.02]
print EBsiteMatch[’La’, ’Cr’, 15.038, 0.02]
print EBsiteMatch[’La’, ’Mn’, 14.442, 0.03]
print EBsiteMatch[’La’, ’Mn’, 14.927, 0.03]
print EBsiteMatch[’La’, ’Fe’, 14.385, 0.035]
print EBsiteMatch[’La’, ’Fe’, 14.878, 0.035]
print EBsiteMatch[’Sr’, ’Cr’, 14.418, 0.03]
print EBsiteMatch[’Sr’, ’Mn’, 14.363, 0.03]
```

Table S16: Values of orbital occupancy at different levels of orbital perturbation (α, eV) for the initial response (χ_0), the differences of the occupation levels of the initial and final responses at zero perturbation ($\chi(0.0) - \chi_0(0.0)$, or $\Delta\chi(0)$), the coefficient of correlation (R^2) of the initial response, the Gaussian spreading values, and levels of strain of all systems tested.

Perovskite	**-0.15**	**-0.10**	**-0.05**	**0.00**	**0.05**	**0.10**	**0.15**	$\Delta\chi(0)$	**{**R\$^{2}\$**}**	**Spreading**	**Strain**
$LaVO_3$	5.10012	4.75405	4.41358	4.08205	3.76261	3.45811	3.17101	0.00274	0.99897	0.035	0
$LaCrO_3$	6.80442	6.34038	5.85581	5.36226	4.87207	4.39749	3.94970	-0.00060	0.99983	0.020	0
$LaCrO_3$	6.80115	6.28916	5.75467	5.21179	4.67546	4.16029	3.67943	-0.00026	0.99973	0.020	+3.5
$LaMnO_3$	7.27716	6.89193	6.48255	6.05428	5.61314	5.16568	4.71873	0.00019	0.99932	0.030	0
$LaMnO_3$	7.24820	6.83715	6.40132	5.94677	5.48033	5.00935	4.54140	-0.00198	0.99946	0.030	+3.4
$LaFeO_3$	8.02657	7.72539	7.39797	7.04666	6.67455	6.28530	5.88310	0.00004	0.99756	0.035	0
$LaFeO_3$	7.99067	7.67047	7.32318	6.95160	6.55929	6.15041	5.72961	0.00024	0.99782	0.035	+3.4
$SrCrO_3$	6.17764	5.83413	5.48689	5.13981	4.79682	4.46182	4.13848	0.00001	0.99987	0.030	0
$SrMnO_3$	6.99640	6.62794	6.24266	5.84513	5.44042	5.03392	4.63115	-0.00169	0.99973	0.030	0

References

(S1) Dominik, C. *The Org-Mode 7 Reference Manual: Organize Your Life with GNU Emacs*; Network Theory: UK, 2010; with contributions by David O'Toole, Bastien Guerry, Philip Rooke, Dan Davison, Eric Schulte, and Thomas Dye.

(S2) Python. `http://python.org/`.

(S3) Codd, E. F. *Relational Completeness of Data Base Sublanguages*; IBM Corporation, 1972.

(S4) Fagin, R. Multivalued Dependencies and a New Normal Form for Relational Databases. **1977**, *2*, 262–278.

(S5) MySQL. `http://www.mysql.com/`.

(S6) sqlite. `https://sqlite.org/`.

(S7) PlantUML. `http://plantuml.sourceforge.net/`.

(S8) Giannozzi, P.; Baroni, S.; Bonini, N.; Calandra, M.; Car, R.; Cavazzoni, C.; Ceresoli, D.; Chiarotti, G. L.; Cococcioni, M.; Dabo, I.; Corso, A. D.; de Gironcoli, S.; Fabris, S.; Fratesi, G.; Gebauer, R.; Gerstmann, U.; Gougoussis, C.; Kokalj, A.; Lazzeri, M.; Martin-Samos, L.; Marzari, N.; Mauri, F.; Mazzarello, R.; Paolini, S.;

Pasquarello, A.; Paulatto, L.; Sbraccia, C.; Scandolo, S.; Sclauzero, G.; Seitsonen, A. P.; Smogunov, A.; Umari, P.; Wentzcovitch, R. M. QUANTUM ESPRESSO: A Modular and Open-source Software Project for Quantum Simulations of Materials. **2009**, *21*, 395502:1–19.

(S9) Mozilla Firefox. `http://www.mozilla.org/en-US/firefox/new/`, Last visited April 20, 2014.

(S10) Murnaghan, F. D. The Compressibility of Media under Extreme Pressures. **1944**, *30*, 244–247.

(S11) Costa, G. A.; Kaiser, E. Structural and Thermal Properties of the Alkaline Cuprate $KCuO_2$. **1995**, *269 - 270*, 591 – 598.

(S12) Wang, L.; Maxisch, T.; Ceder, G. Oxidation Energies of Transition Metal Oxides within the GGA + U Framework. **2006**, *73*, 195107:1–6.

(S13) Linstrom, P. J.; Mallard, E., W. G. *NIST Chemistry WebBook, NIST Standard Reference Database Number 69*; National Institute of Standards and Measurements: Gaithersburg, MD 20899, 2011; Retrieved April 2, 2014.

(S14) Foster, A. S.; Lopez Gejo, F.; Shluger, A. L.; Nieminen, R. M. Vacancy and Interstitial Defects in Hafnia. **2002**, *65*, 174117:1–13.

(S15) Aroyo, M. I.; Perez-Mato, J. M.; Capillas, C.; Kroumova, E.; Ivantchev, S.; Madariaga, G.; Kirov, A.; Wondratschek, H. Bilbao Crystallographic Server: I. Databases and Crystallographic Computing Programs. **2006**, *221*, 15–27.

(S16) Momma, K.; Izumi, F. *VESTA*: A Three-dimensional Visualization System for Electronic and Structural Analysis. **2008**, *41*, 653–658.

(S17) Haas, O.; Struis, R.; McBreen, J. Synchrotron X-ray Absorption of $LaCoO_3$ Perovskite. **2004**, *177*, 1000 – 1010.

(S18) Geller, S.; Raccah, P. M. Phase Transitions in Perovskite-like Compounds of the Rare Earths. **1970**, *2*, 1167–1172.

(S19) Tsuda, K.; Ogata, Y.; Takagi, K.; Hashimoto, T.; Tanaka, M.

(S20) Norby, P.; Andersen, I.; Andersen, E.; Andersen, N. The Crystal Structure of Lanthanum Manganate(iii), $LaMnO_3$, at Room Temperature and at 1273 K under N_2. **1995**, *119*, 191 – 196.

(S21) De Graef, M.; McHenry, M. E. *Structure of Materials: An Introduction to Crystallography, Diffraction and Symmetry*; Cambridge University Press, 2007.

(S22) Lee, Y.-L.; Kleis, J.; Rossmeisl, J.; Morgan, D. *Ab initio* Energetics of LaBO$_3$(001) (B = Mn, Fe, Co, and Ni) for Solid Oxide Fuel Cell Cathodes. **2009**, *80*, 224101:1–20.

(S23) Pavone, M.; Ritzmann, A. M.; Carter, E. A. Quantum-mechanics-based Design Principles for Solid Oxide Fuel Cell Cathode Materials. **2011**, *4*, 4933–4937.

(S24) Nonstoichiometry and defect structure of the perovskite-type oxides $La_{1-x}Sr_xFeO_{3-\delta}$. **1985**, *58*, 257–266.

(S25) Nonstoichiometry of the perovskite-type oxides $La_{1-x}Sr_xCoO$. **1989**, *80*, 102–111.

(S26) Nowotny, J.; Rekas, M. Defect Chemistry of (La,Sr)MnO_3. **1998**, *81*, 67–80.

(S27) Kulik, H. J.; Marzari, N. Accurate Potential Energy Surfaces with a DFT+U(R) Approach. **2011**, *135*, 194105:1–10.

(S28) Hsu, H.; Blaha, P.; Wentzcovitch, R. M. Ferromagnetic Insulating State in Tensile-strained $LaCoO_3$ Thin Films from LDA + U Calculations. **2012**, *85*, 140404(R):1–5.

(S29) Kulik, H. J.; Cococcioni, M.; Scherlis, D. A.; Marzari, N. Density Functional Theory in Transition-Metal Chemistry: A Self-Consistent Hubbard U Approach. **2006**, *97*, 103001:1–4.