Contents

1 **Review of publishing using Emacs** **1**
1.1 '
' - Line break . . . 1
1.2 '\' - Forced spacing in equations . . . 2
1.3 Publish the file . . . 2

2 **Review of VASP tags and their purpose** **3**
2.1 XC and GGA . . . 3
2.2 VASP 'behind the scenes' . . . 4
2.2.1 #1 - initializing the atoms object . . . 5
2.2.2 #2 - writing the POSCAR . . . 5
2.2.3 #3 - writing the INCAR . . . 5
2.2.4 #4 - writing the POTCAR . . . 5
2.2.5 #5 - writing the KPOINTS . . . 6
2.2.6 #6 - writing the ASE sorting order . . . 6

3 **How to set up a master bibliography file** **6**

1 Review of publishing using Emacs

Notes from last class:

`(find-file(expand-file-name"examples"starter-kit-dir))`

LaTeX is the basis of all document publication that Emacs performs. Here, we review some LaTeX syntax.

1.1 '
' - Line break

Line breaks are most commonly used in equations with multiple lines.

Equations with line breaks:

Type C-c C-x C-l to toggle the images

inline math $x^2 = 4$

$x^3 = 27$

display math

$$x^4 = 16$$

$$\begin{aligned} OVac\_FormE(cID) \rightarrow OVac\_FormE(&mID, mag, num\_atoms, \\ &orientation, correction, calc\_quantity, \\ &calculator, figure) \end{aligned} \tag{1}$$

Equation without:

$$OVac\_FormE(cID) \rightarrow OVac\_FormE(mID, mag, num\_atoms, \quad orientation, correction, calc\_quan \tag{2}$$

NOTE: The '&' symbol indicates where you would like to align the equation. for example:

$$\begin{aligned} Equation &= blah \quad (3) \\ &= blahblah \quad (4) \\ &= blahblahblah \quad (5) \\ & \quad (6) \end{aligned}$$

1.2 '\' - Forced spacing in equations

Sometimes you would like to add space into your equations. This can be done using a single slash:

$$\begin{aligned} Equation &= blah \quad (7) \\ &= blah\ blah \quad (8) \\ &= blah\ blah\ blah \quad (9) \\ & \quad (10) \end{aligned}$$

1.3 Publish the file

C-c e o will start the vanilla export. You get a menu of options. Press l o to make and open a pdf (via LaTeX), or h o to make and open an html file. Note, you may need latex_header tags for some packages. org does some things by default, but not everything.

2 Review of VASP tags and their purpose

It's important to keep in mind that the purpose of VASP is to implement the Kohn-Sham equations in order to solve for properties of interested for various atomic configurations.

All of the tags we implement are a means to that end.

2.1 XC and GGA

Hopefully, this one is pretty self explanatory.

The exchange correlation functional is every part of the Kohn-Sham equation that we don't know the exact solution for. So, it needs to be approximated.

We do this by setting the XC tag to either 'LDA' (Local-Density Approximation), or 'PBE' (Perdew, Burke, and Ernzerhof).

```
from jasp import *
from ase.structure import molecule

atoms = molecule('H2')
atoms.set_cell([6, 6, 6])
atoms.center()

with jasp('./molecules/func-pbe',
          xc='PBE',
          atoms=atoms) as calc:
    try:

    calc.initialize(atoms)
    calc.write_potcar()
```

The GGA tag stands for generalized gradient approximation. This is a class of functionals which include gradients (As opposed to LDA which is constant).

Here are a list of all the functionals you may use in the future: http://cms.mpi.univie.ac.at/vasp/vasp/GGA_tag.html

91 Perdew -Wang 91 PE Perdew-Burke-Ernzerhof RP revised Perdew-Burke-Ernzerhof AM AM05 PS Perdew-Burke-Ernzerhof revised for solids

When implementing the GGA tag, it is recommended that you set xc equal to 'LDA' as follows:

```
from jasp import *
from ase.structure import molecule

atoms = molecule('H2')
```

```python
atoms.set_cell([6, 6, 6])
atoms.center()

with jasp('./molecules/func-ps',
          xc='LDA',
          gga='PS',
          atoms=atoms) as calc:

    calc.initialize(atoms)
    calc.write_potcar()
```

The POTCAR contains all of the information needed to calculate the Kohn-Sham equations for the atoms in your atoms object for a specific functional.

The XC tag tells VASP where to get the POTCAR information from. The producers of VASP have specifically tailored these files to calculate energies for each atom type quickly and effectively.

These files are what separate VASP from other codes which perform molecular simulation studies.

2.2 VASP 'behind the scenes'

Notice in the previous example, I've used JASP to only produce one of the initialization files needed for a VASP calculations.

Keep in mind that JASP is only a 'wrapper' for VASP!

i.e. JASP is a fancy interface, designed to make VASP easier to use

e.g. without JASP and ASE (Atomic Simulation Environment), we would need to make these input files manually!

We can use JASP for all of the Initialization files needed to run a VASP calculation.

The following code block shows the calculation initialization commands being run every time you start a new calculation:

```python
from jasp import *
from ase.structure import molecule

atoms = molecule('CH4')
atoms.set_cell([6, 6, 6])
atoms.center()

with jasp('./molecules/func-initial',
          xc='PBE',
          gga='PS',
          ibrion=-1,
          atoms=atoms) as calc:

```

```
    calc.initialize(atoms)  #1
    write_vasp('POSCAR',
               calc.atoms_sorted,
               symbol_count=calc.symbol_count)  #2
    calc.write_incar(atoms)  #3
    calc.write_potcar()  #4
    calc.write_kpoints()  #5
    calc.write_sort_file()  #6
```

2.2.1 #1 - initializing the atoms object

Running this function defines a large number of variables based on the atoms object

These variables will be utilized by JASP and ASE later to create the other initialization files

JASP also creates a METADATA file here, to record meta-information about the calculation: ./molecules/func-initial/METADATA

2.2.2 #2 - writing the POSCAR

This code generates a POSCAR file using ASEs atom sorting tool

POSCAR is fancy coder abbreviation for POSition Component Application Recourse, which is fancy speak for 'information about atom positions'

The ASE atoms sorter is used to collapse like atoms together so the POTCAR can be written into one succinct file.

Without this sorting feature, each atom would need its OWN POTCAR. 50 atom unit cell? 50 identical POTCAR files.

Here's what the POSCAR file looks like: ./molecules/func-initial/POSCAR

2.2.3 #3 - writing the INCAR

This code writes the INCAR file needed to start a VASP calculation

Like POSCAR, this file is another abbreviation for INput Component Application Recourse, i.e. calculation inputs

This file contains all of the input parameters specific to your calculation, and will likely look very familiar to you: ./molecules/func-initial/INCAR

2.2.4 #4 - writing the POTCAR

Writes the POTential Component Application Recourse file.

As described above, this file contains the nuts and bolts of what makes VASP unique

There is information about how to solve the Kohn-Sham equation specific to the functional being used

There is also an entry for each type of atom in your atoms object!

Try searching for 'PAW_PBE' in this file to see what I mean: `./molecules/func-initial/POTCAR`

The POSCAR and POTCAR files must be generated together, since ASE uses special sorting methods to represent all of the atoms in the POSCAR in *one* POTCAR file.

2.2.5 #5 - writing the KPOINTS

k-points are used to sample the Brillouin zone (`http://en.wikipedia.org/wiki/Brillouin_zone`) of a lattice to discretely measure characteristics of a continuous lattice

You may not have know this existed yet, because we don't need more than one k-point for molecule simulation!

Notice that the KPOINTS file generate has 1 k-point by default `./molecules/func-initial/KPOINTS`

This file is essential for ANY VASP calculation. Don't believe me? Go back and look in one of your old calculation folders.

We will discuss k-points more when we get into bulk systems next class

2.2.6 #6 - writing the ASE sorting order

The ASE sorting utility is useful for collapsing the POTCAR into one file

The ASE sorting file helps ASE keep track of potential differences between indices of atoms in your hand made atoms objects, and the order that ASE prefers for simplifying the POTCAR file.

In this case, there is no difference, perhaps since the atoms object was generated using ASE: `./molecules/func-initial/ase-sort.dat`

3 How to set up a master bibliography file

Inside your jmax directory there is a folder called user

All emacs-lisp code lines in this folder are automatically loaded every time you open emacs

Inside this file, you can create a custom folder to load special modules that you enjoy in emacs:

Here is a code block you can run to generate a master bibliography setup in your home directory:

```
(unless (file-exists-p "~/bibliography")
  (make-directory "~/bibliography"))

(with-current-buffer (find-file-noselect
                      (expand-file-name
                       "user/my-bibliography.el" starter-kit-dir))
  (insert
   "(setq org-ref-bibliography-notes \"~/bibliography/notes.org\"
      org-ref-default-bibliography '(\"~/bibliography/references.bib\")
      org-ref-pdf-directory \"~/bibliography/bibtex-pdfs/\")
")
  (save-buffer)
  (kill-buffer))


(load-file (expand-file-name "user/my-bibliography.el" starter-kit-dir))
```

```
t
```

You only need to run that once, subsequent times you open emacs the new file will be automatically loaded.