hwk3.org Due 10/12/2012

John Kitchin

2012-09-22 Sat

Contents

1	Rea	action energy of CO oxidation	1
	1.1	Compute the reaction energy for $CO + 1/2 O_2 \rightarrow CO_2$	1
	1.2	Convergence test	3
2	Zero-point energy corrections		
	2.1	Compute vibrational modes for CO , CO_2 and O_2	6
		2.1.1 O2	6
		2.1.2 CO2	7
		2.1.3 CO	7
	2.2	Compute the CO oxidation reaction energy with zero-point	
		energy corrections	8
	2.3	Compare your computed energy to a value from the literature.	9
3	Plo	t the electron density of the CO2 molecule.	9

1 Reaction energy of CO oxidation

1.1 Compute the reaction energy for $CO + 1/2 O_2 \rightarrow CO_2$

Use a cutoff energy of 250 eV. The molecules should all be relaxed to their lowest energy geometry (perform a geometry optimization). Demonstrate that all the forces on the molecule are less than 0.05 eV/Å.

from jasp import *
from ase.data.molecules import molecule

```
atoms = molecule('CO')
4
    atoms.set_cell((6,6,6), scale_atoms=False)
5
6
    atoms.center()
    with jasp('hwk3/co-250',
8
              xc='PBE',
9
10
               encut=250,
              ibrion=2, nsw=20, ediffg=-0.05,
11
              atoms=atoms) as calc:
12
13
            co = atoms.get_potential_energy()
14
15
         except (VaspSubmitted, VaspQueued):
            co = None
16
17
    atoms = molecule('CO2')
18
    atoms.set_cell((6,6,6), scale_atoms=False)
19
20
    atoms.center()
21
    with jasp('hwk3/co2-250',
              xc='PBE',
23
               encut=250,
24
              ibrion=2, nsw=20,
25
              atoms=atoms) as calc:
26
27
            co2 = atoms.get_potential_energy()
28
29
        except (VaspSubmitted, VaspQueued):
            co2 = None
30
31
    atoms = molecule('02')
32
    atoms.set_cell((6,6,6), scale_atoms=False)
33
    atoms.center()
    atoms.set_initial_magnetic_moments([1,1])
35
36
    with jasp('hwk3/o2-250',
37
38
              xc='PBE',
39
               encut=250,
              ispin=2,
40
               ibrion=2, nsw=20,
41
               atoms=atoms) as calc:
42
43
        try:
44
            o2 = atoms.get_potential_energy()
         except (VaspSubmitted, VaspQueued):
45
            o2 = None
46
47
    print co, o2, co2
48
49
    if None not in (co, o2, co2):
50
        Erxn = co2 - co - 0.5*o2
51
        print 'The reaction energy is {0:1.3f} eV'.format(Erxn)
52
    from jasp import *
1
    with jasp('hwk3/o2-250') as calc:
2
        atoms = calc.get_atoms()
3
        print '02'
4
        print atoms.get_forces()
5
```

```
6
    with jasp('hwk3/co2-250') as calc:
7
        atoms = calc.get_atoms()
8
        print 'CO2'
9
10
        print atoms.get_forces()
11
12
    with jasp('hwk3/co-250') as calc:
13
        atoms = calc.get_atoms()
14
        print 'CO'
15
        print atoms.get_forces()
16
```

For a reason I am not clear on, VASP converged the CO calculation with a residual force of 0.44 eV/ang until I specified via EDIFFG to make the force less than 0.05 eV/ang. This could be because the change in total energy between two steps was less than the EDIFFG default setting which defaults to 1 meV. Lesson: you must specify the criteria you want converged! In this case, that means set EDIFFG=-0.05. The negative sign means it is a force criteria, not an energy criteria.

1.2 Convergence test

Repeat the previous problem at 350, 450, and 500 eV. Reoptimize the geometry at each ENCUT value. Compare (in a graph) the convergence of the total energy of each species with the convergence of the reaction energy. Which converges faster?

```
from jasp import *
1
    from ase.data.molecules import molecule
3
    for encut in [350, 450, 500]:
4
5
        atoms = molecule('CO')
6
        atoms.set_cell((6,6,6), scale_atoms=False)
7
         atoms.center()
8
        with jasp('hwk3/co-{0}'.format(encut),
9
                   xc='PBE',
10
                   encut=encut,
11
12
                   ibrion=2, nsw=20,
                   atoms=atoms) as calc:
13
14
                 co = atoms.get_potential_energy()
15
             except (VaspSubmitted, VaspQueued):
16
17
                 co = None
18
         atoms = molecule('CO2')
         atoms.set_cell((6,6,6), scale_atoms=False)
20
21
         atoms.center()
22
```

```
with jasp('hwk3/co2-{0}'.format(encut),
23
                   xc='PBE',
24
25
                   encut=encut.
                   ibrion=2, nsw=20,
26
                   atoms=atoms) as calc:
27
28
             try:
29
                 co2 = atoms.get_potential_energy()
             except (VaspSubmitted, VaspQueued):
30
                 co2 = None
31
32
         atoms = molecule('02')
33
34
         atoms.set_cell((6,6,6), scale_atoms=False)
         atoms.center()
35
36
         atoms.set_initial_magnetic_moments([1,1])
37
         with jasp('hwk3/o2-{0}'.format(encut),
38
39
                   xc='PBE',
                   encut=encut,
40
                   ispin=2,
41
                   ibrion=2, nsw=20,
42
                   atoms=atoms) as calc:
43
44
             try:
                 o2 = atoms.get_potential_energy()
45
46
             except (VaspSubmitted, VaspQueued):
                 o2 = None
47
48
         if None not in (co, o2, co2):
49
50
             Erxn = co2 - co - 0.5*o2
51
             print 'At {1} eV the reaction energy is {0:1.3f} eV'.format(Erxn, encut)
52
```

```
1
    from jasp import *
    import numpy as np
2
    import matplotlib.pyplot as plt
4
    encuts = [250, 350, 450, 500]
5
6
    co, co2, o2 = [], [],
7
9
    for encut in encuts:
        with jasp('hwk3/co-{0}'.format(encut)) as calc:
10
11
            atoms = calc.get_atoms()
            co.append(atoms.get_potential_energy())
12
13
        with jasp('hwk3/co2-{0}'.format(encut)) as calc:
14
            atoms = calc.get_atoms()
15
            co2.append(atoms.get_potential_energy())
16
17
        with jasp('hwk3/o2-{0}'.format(encut)) as calc:
18
            atoms = calc.get_atoms()
19
20
            o2.append(atoms.get_potential_energy())
21
   co = np.array(co)
22
23
    co2 = np.array(co2)
    o2 = np.array(o2)
```

```
25
    plt.plot(encuts, co - co.min())
26
    plt.plot(encuts, co2 - co2.min())
27
    plt.plot(encuts, o2 - o2.min())
28
    plt.xlabel('ENCUT (eV)')
29
    plt.ylabel('$\Delta E$ (eV)')
30
31
    plt.legend(['CO', 'CO2', 'O2'], loc='best')
    plt.savefig('hwk3/molecular-convergence.png')
32
33
34
    plt.figure()
    plt.plot(encuts, co2 - co - 0.5*o2)
35
36
    plt.xlabel('ENCUT (eV)')
    plt.ylabel('$H_{rxn}$ (eV)')
37
38
    plt.savefig('hwk3/rxn-encut-convergence.png')
39
    plt.show()
40
```

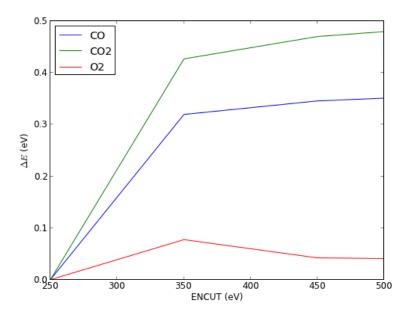


Figure 1: Convergence with ENCUT of molecules. The difference is with respect to the molecule energy at 250 eV.

It is probably safe to say that the reaction energy is within $0.02~{\rm eV}$ of $3.11~{\rm eV}$. The energy has clearly not completely converged, even at $500~{\rm eV}$. Note the reaction energy change due to convergence is much smaller than the changes in the CO and CO2 molecular energies.

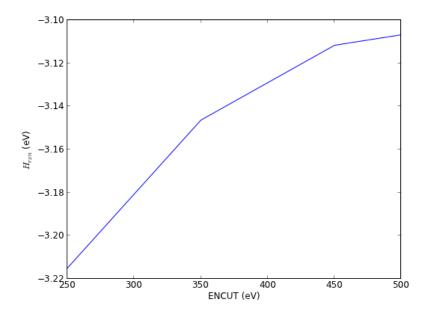


Figure 2: Convergence of the CO oxidation reaction.

2 Zero-point energy corrections

2.1 Compute vibrational modes for CO, CO_2 and O_2

Compute the vibrational modes of each molecule in the CO oxidation reaction. Do this at 350 eV cutoff energy only. Prepare a table of the vibrational modes for molecule.

2.1.1 O2

```
1
    from jasp import *
2
3
    with jasp('hwk3/o2-350') as calc:
        calc.clone('hwk3/o2-350-vib')
        atoms = calc.get_atoms()
5
    with jasp('hwk3/o2-350-vib',
7
8
               ibrion=5, nfree=2, nsw=1,
               atoms=atoms) as calc:
10
        try:
            calc.calculate()
11
            energies, modes = calc.get_vibrational_modes()
12
            print '#+caption: 02 vibrational energies'
```

```
print '| mode | energy (eV) |'
print '|-'
for i, energy in enumerate(energies):
print '| {0} | {1} |'.format(i, energy)
except (VaspSubmitted, VaspQueued):
pass
```

2.1.2 CO2

```
1
    from jasp import *
2
    with jasp('hwk3/co2-350') as calc:
3
         calc.clone('hwk3/co2-350-vib')
4
         atoms = calc.get_atoms()
5
6
7
    with jasp('hwk3/co2-350-vib',
8
               ibrion=5, nfree=2, nsw=1,
               atoms=atoms) as calc:
9
10
         try:
            calc.calculate()
11
            energies, modes = calc.get_vibrational_modes()
12
            print '#+caption: CO2 vibrational energies'
13
            print '| mode | energy (eV) |'
14
15
            print '|-'
             for i, energy in enumerate(energies):
16
                 print '| {0} | {1} |'.format(i, energy)
17
         except (VaspSubmitted, VaspQueued):
18
19
            pass
```

2.1.3 CO

```
from jasp import *
    with jasp('hwk3/co-350') as calc:
2
         calc.clone('hwk3/co-350-vib')
        atoms = calc.get_atoms()
4
5
    with jasp('hwk3/co-350-vib',
6
               ibrion=5, nfree=2, nsw=1,
8
               atoms=atoms) as calc:
9
        try:
            calc.calculate()
10
            energies, modes = calc.get_vibrational_modes()
11
            print '#+caption: CO vibrational energies'
12
13
            print '| mode | energy (eV) |'
            print '|-'
14
15
             for i, energy in enumerate(energies):
                 print '| {0} | {1} |'.format(i, energy)
16
17
         except (VaspSubmitted, VaspQueued):
18
            pass
```

2.2 Compute the CO oxidation reaction energy with zeropoint energy corrections.

Compare the reaction energy with and without the zero-point energy correction.

We compute zero point energy as the sum of $0.5*h*nu_i$

```
from jasp import *
   import numpy as np
3
    c = 3e10 # speed of light cm/s
    h = 4.135667516e-15 # eV/s
    with jasp('hwk3/o2-350-vib') as calc:
        freq = calc.get_vibrational_frequencies()
7
8
        o2_ZPE = 0.0
9
10
        for f in freq:
            if not isinstance(f,float):
11
                continue #skip complex numbers
12
            nu = f*c # convert to frequency
13
            o2_ZPE += 0.5*h*nu
14
        print o2_ZPE
15
16
17
    with jasp('hwk3/co-350-vib') as calc:
18
        freq = calc.get_vibrational_frequencies()
19
20
        co_ZPE = 0.0
21
        for f in freq:
22
23
            if not isinstance(f,float):
               continue #skip complex numbers
24
25
            nu = f*c # convert to frequency
            co_ZPE += 0.5*h*nu
26
27
        print co_ZPE
28
29
    with jasp('hwk3/co2-350-vib') as calc:
30
        freq = calc.get_vibrational_frequencies()
31
32
        co2\_ZPE = 0.0
33
        for f in freq:
34
35
            if not isinstance(f,float):
                continue #skip complex numbers
36
            nu = f*c # convert to frequency
37
            co2_ZPE += 0.5*h*nu
38
        print co2_ZPE
39
40
    rxn_ZPE = co2_ZPE - co_ZPE - 0.5*o2_ZPE
41
42
    print 'The ZPE correction to the reaction is {0:1.3f} eV'.format(rxn_ZPE)
43
44
    print 'E_rxn_zpe = {0:1.3f} eV'.format(-3.1 + rxn_ZPE)
```

2.3 Compare your computed energy to a value from the literature.

Provide a reference for your literature value.

Table 1: Standard enthalpies of formation (webbook.nist.gov)

The standard enthalpy of reaction is -282.98 kJ/mol which is about -2.95 eV.

Our calculated value is **remarkably** close to the experimental value after zero point energy correction! This is certainly due to fortuitous error cancellations. DFT is not usually this good.

3 Plot the electron density of the CO2 molecule.

Include the figure in your homework.

```
from jasp import *
    from enthought.mayavi import mlab
    from ase.data import vdw_radii
    from ase.data.colors import cpk_colors
4
    with jasp('hwk3/co2-350') as calc:
6
        atoms = calc.get_atoms()
        x, y, z, cd = calc.get_charge_density()
8
9
    mlab.figure(1, bgcolor=(1, 1, 1)) # make a white figure
10
   mlab.contour3d(x, y, z, cd)
11
    mlab.view(azimuth=-90, elevation=90, distance='auto')
    mlab.savefig('hwk3/co2-cd.png')
```

It looks a lot like the CO charge density looking in the mirror!

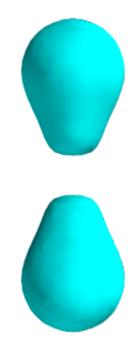


Figure 3: CO2 charge density.