

HOMEWORK 1 - Due 9/6/2012

Zhongnan Xu

9/6/12 Thursday

Contents

1	Signup for an account at gitHub.	1
2	Read Chapter 1 in the text book.	1
3	Read Section 5 (Molecules) in dft-book.	2
4	Data fitting.	2
5	Nonlinear algebra	3
6	Linear algebra	4

1 Signup for an account at gitHub.

Print your username here: zhongnanxu (note, I've changed it from my previous 'xuzho' one for consistency)

Set yourself up to watch <https://github.com/jkitchin/dft-course> and <https://github.com/jkitchin/dft-book>.

2 Read Chapter 1 in the text book.

You do not need to write anything. Just do it.

3 Read Section 5 (Molecules) in dft-book.

As part of this assignment, please turn in a pdf copy of dft-book that has been annotated by sticky notes using Adobe Acrobat Reader (you should be able to type Ctrl-6 to get a sticky note while the pdf is open, and then you can move it where you want and type text in it.). Please note any typos, places that are confusing, etc. . .

Please see pdf file in the hw-1 folder

4 Data fitting.

Fit a cubic polynomial to this set of data and estimate the lattice constant that minimizes the total energy. Prepare a figure that shows the data, your fit and your estimated minimum. Hints: `numpy.polyfit`, `numpy.polyder`, `numpy.roots`, `numpy.linspace`, `numpy.polyval` will all help you do this easily.

lattice constant (\AA)	Total Energy (eV)
3.5	-3.649238
3.55	-3.696204
3.6	-3.719946
3.65	-3.723951
3.7	-3.711284
3.75	-3.68426

Code

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 lats, energies = zip(*table)
5 coeffs = np.polyfit(lats, energies, 3)
6 ders = np.polyder(coeffs)
7 roots = np.roots(ders)
8 # We now need to see what the roots are and pick the sensible one
9 print 'The two roots are {0:1.3f} and {1:1.3f}'.format(roots[0], roots[1])
10 print 'Therefore, {0:1.3f} is the minimum lattice constant'.format(roots[1])
11 lats_func = np.linspace(3.45, 3.80)
12 energies_func = np.polyval(coeffs, lats_func)
13
14 fig = plt.figure(1, (5, 4))
15 ax = fig.add_subplot(111)
16 ax.plot(lats, energies, marker='o', linestyle='None', color='k',
17         label='Experimental values')
18 ax.plot(lats_func, energies_func, marker='None', linestyle='--', color='r',
```

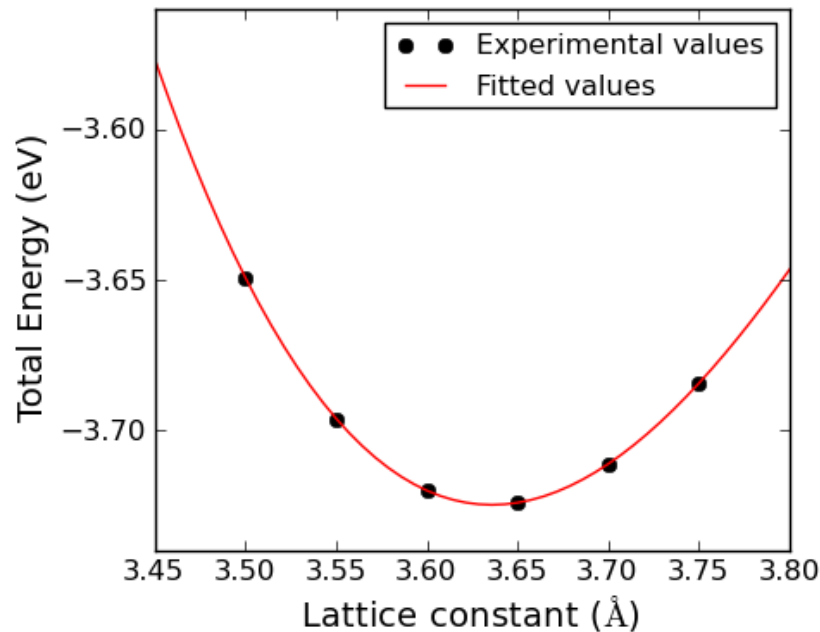
```

19         label='Fitted values')
20 ax.set_xlabel('Lattice constant ($\AA$)', size='large')
21 ax.set_ylabel('Total Energy (eV)', size='large')
22 ax.legend(prop={'size': 'medium'})
23 fig.tight_layout()
24 plt.show()

```

Output

The two roots are 4.233 and 3.636
Therefore, 3.636 is the minimum lattice constant



5 Nonlinear algebra

Solve this equation: $\sin(x^2) = 0.5$ for x . Prepare a plot of the function and show where your solution is. Hint: `scipy.optimize.fsolve`

Code

```

1 import numpy as np
2 from scipy.optimize import fsolve
3 import matplotlib.pyplot as plt
4

```

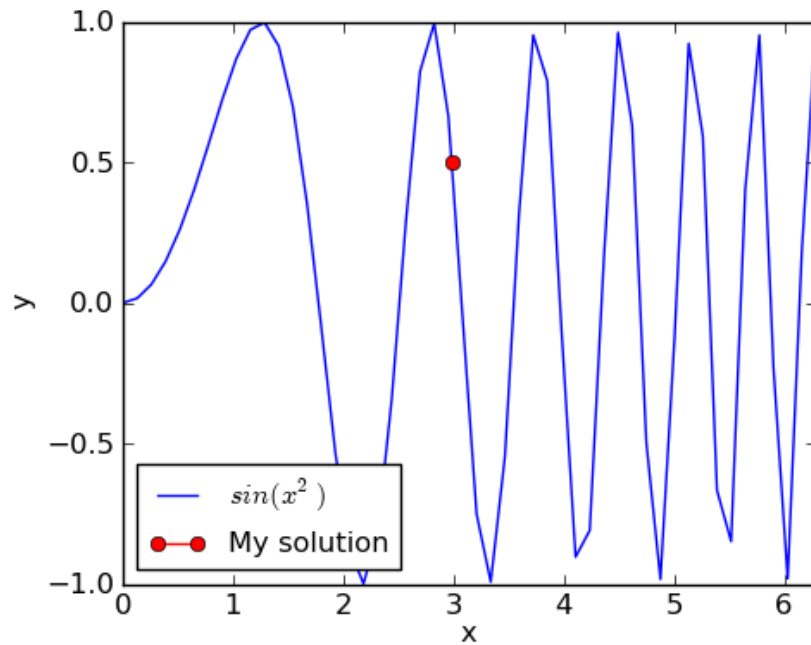
```

5 def func(x):
6     return np.sin(x**2) - 0.5
7 print 'The answer is {0:1.3f}'.format(fsolve(func, np.pi)[0])
8 fig = plt.figure(1, (5, 4))
9 ax = fig.add_subplot(111)
10 x = np.linspace(0, 2 * np.pi)
11 y = func(x) + 0.5
12 ax.plot(x, y, marker='None', color='b', label='$sin(x^2)$')
13 ax.plot(2.983, np.sin(2.983**2), marker='o', color='r', label='My solution')
14 ax.set_xlabel('x')
15 ax.set_ylabel('y')
16 ax.set_xlim((0, 2 * np.pi))
17 ax.legend(prop={'size': 'medium'}, loc=3)
18 fig.tight_layout()
19 plt.show()

```

Output

The answer is 2.983



6 Linear algebra

Solve these equations using python and linear algebra:

$$a_0 - 3a_1 + 9a_2 - 27a_3 = -2 \quad (1)$$

$$a_0 - a_1 + a_2 - a_3 = 2 \quad (2)$$

$$a_0 + a_1 + a_2 + a_3 = 5 \quad (3)$$

$$a_0 + 2a_1 + 4a_2 + 8a_3 = 1 \quad (4)$$

Use linear algebra to verify your solution. Hint: see `numpy.linalg`, `numpy.dot`.

Code

```

1 import numpy as np
2 # This can be solved with matrix algebra, Ax = B
3 a = np.array([[1, -3, 9, -27],
4               [1, -1, 1, -1],
5               [1, 1, 1, 1],
6               [1, 2, 4, 8]])
7 b = np.array([-2, 2, 5, 1])
8 x = np.linalg.solve(a, b)
9 print 'The answer is'
10 print 'a0={0:1.3f}'.format(x[0])
11 print 'a1={0:1.3f}'.format(x[1])
12 print 'a2={0:1.3f}'.format(x[2])
13 print 'a3={0:1.3f}'.format(x[3])
14 # Check that the solution is correct by performing a
15 # A dot x operation
16 print np.dot(a, x)
```

Output

```

The answer is
a0=4.650
a1=1.842
a2=-1.150
a3=-0.342
[-2.  2.  5.  1.]
```