Homework 1 - Due 9/6/2012

John Kitchin

2012-09-01 Sat

Instructions: All calculations should be performed in python. You should turn in the code used, and the answers you got.

1 Signup for an account at gitHub.

Print your username here:

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.

3 Read Section 4 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...

4 Data fitting.

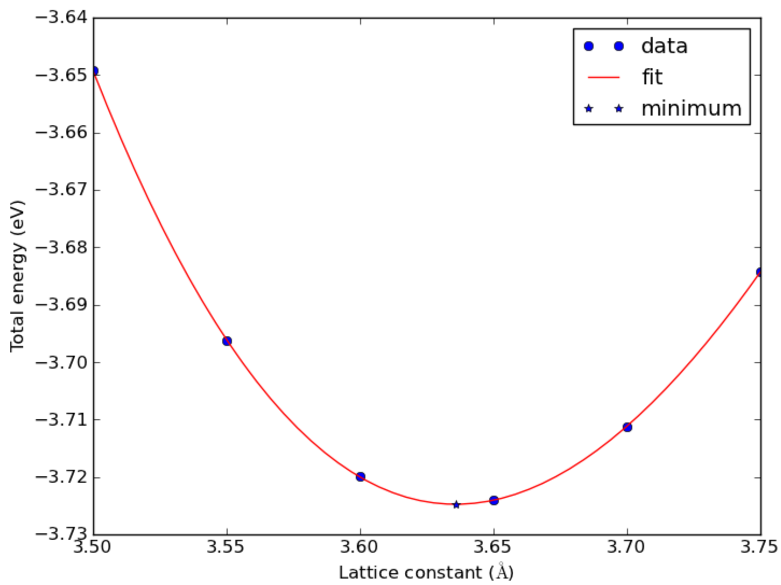
Fit a cubic polynomial to this set of data and estimate the lattice constant that minimizes the total energy. Prepare a figure that shows the data, your fit and your estimated minimum. Hints: `numpy.polyfit`, `numpy.polyder`, `numpy.roots`, `numpy.linspace`, `numpy.polyval` will all help you do this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

4.1 solution

```python
import numpy as np
import matplotlib.pyplot as plt

a = [3.5, 3.55, 3.6, 3.65, 3.7, 3.75]
e = [-3.649238, -3.696204,  -3.719946, -3.723951,  -3.711284,  -3.68426]

# get polynomial fit
pp = np.polyfit(a, e, 3)
x = np.linspace(3.5, 3.75)
fit = np.polyval(pp, x)

# analytical minimum
dp = np.polyder(pp)
r = np.roots(dp)
print 'derivative = 0 at {0}'.format(r)
print 'The minimum is at a lattice constant of {0:1.2f} Ang'.format(r[1])

plt.plot(a, e, 'bo ')
plt.plot(x, fit, 'r-')
plt.plot(r[1], np.polyval(pp, r[1]), '*')
plt.legend(['data', 'fit', 'minimum'], loc='best')
plt.xlabel('Lattice constant ($\AA$)')
plt.ylabel('Total energy (eV)')
plt.savefig('cubic-polynomial.png')
plt.show()
```

```
derivative = 0 at [ 4.23338452  3.63579752]
The minimum is at a lattice constant of 3.64 Ang
```



5 Nonlinear algebra

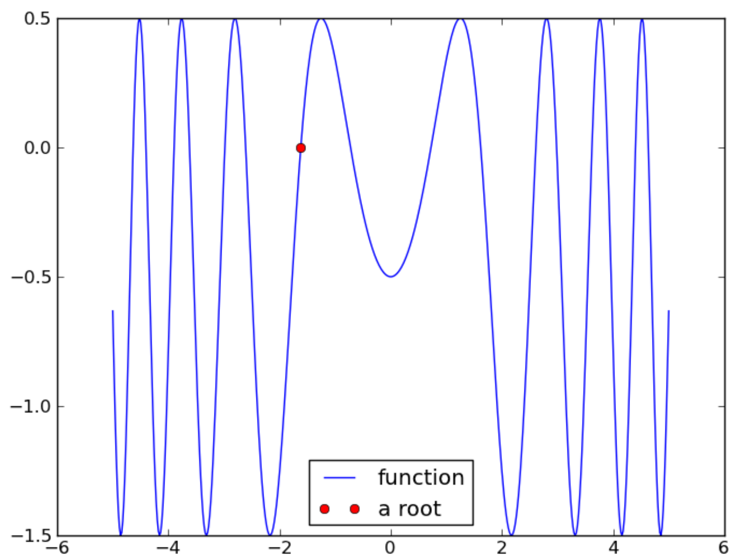
Solve this equation: $\sin(x^2) = 0.5$ for x. Prepare a plot of the function and show where your solution is. Hint: `scipy.optimize.fsolve`

5.1 solution

We have to solve the equation $\sin(x^2) - 0.5 = 0.0$. Plotting this function shows there are many roots. You need to pick one of them and solve the equation like this.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve

def func(x):
    return np.sin(x**2) - 0.5

x = np.linspace(-5,5,500) # use 500 points for smoother graph
plt.plot(x, func(x))

x0 = -2.0

ans = fsolve(func, x0)
```

```
plt.plot(ans, func(ans), 'ro')
plt.legend(['function', 'a root'], loc='best')
plt.savefig('nonlinear-algebra.png')
plt.show()
```



6 Linear algebra

Solve these equations using python and linear algebra:

$$a0 - 3a1 + 9a2 - 27a3 = -2 \quad (1)$$
$$a0 - a1 + a2 - a3 = 2 \quad (2)$$
$$a0 + a1 + a2 + a3 = 5 \quad (3)$$
$$a0 + 2a1 + 4a2 + 8a3 = 1 \quad (4)$$

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

6.1 solution

```
import numpy as np

```

```python
A = np.array([[1, -3, 9, -27],
              [1, -1, 1, -1],
              [1, 1, 1, 1],
              [1, 2, 4, 8]])

b = np.array([-2, 2, 5, 1])

x = np.linalg.solve(A,b)
print 'The solution is {0}'.format(x)

print np.dot(A,x)
print b

# show these are the same with code. Note the use of float tolerance here
TOLERANCE = 1e-9
print (np.abs(np.dot(A,x) - b) < TOLERANCE).all()
```

```
The solution is [ 4.65        1.84166667 -1.15       -0.34166667]
[-2.  2.  5.  1.]
[-2  2  5  1]
True
```