# Chem 30324, Spring 2020, Homework 8

Due April 3, 2020

## Chemical bonding

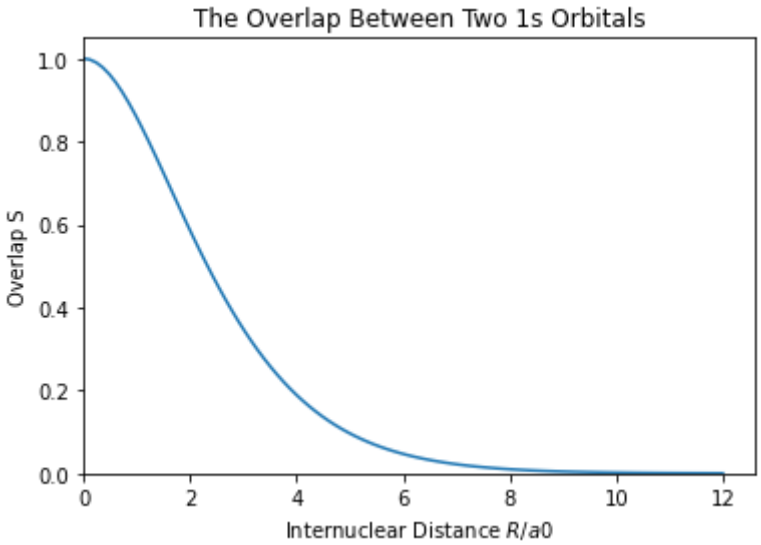
**The electron wavefunctions (molecular orbitals) in molecules can be thought of as coming from combinations of atomic orbitals on the constituent atoms. One of the factors that determines whether two atomic orbitals form a bond is there ability to overlap. Consider two atoms, A and B, aligned on the z axis and separated by a distance $R$.**

**1. The overlap between two 1s orbitals on A and B can be shown to be:**

$$S = \left\{ 1 + \frac{R}{a_0} + \frac{1}{3}\left(\frac{R}{a_0}\right)^2 \right\} e^{-R/a_0}$$

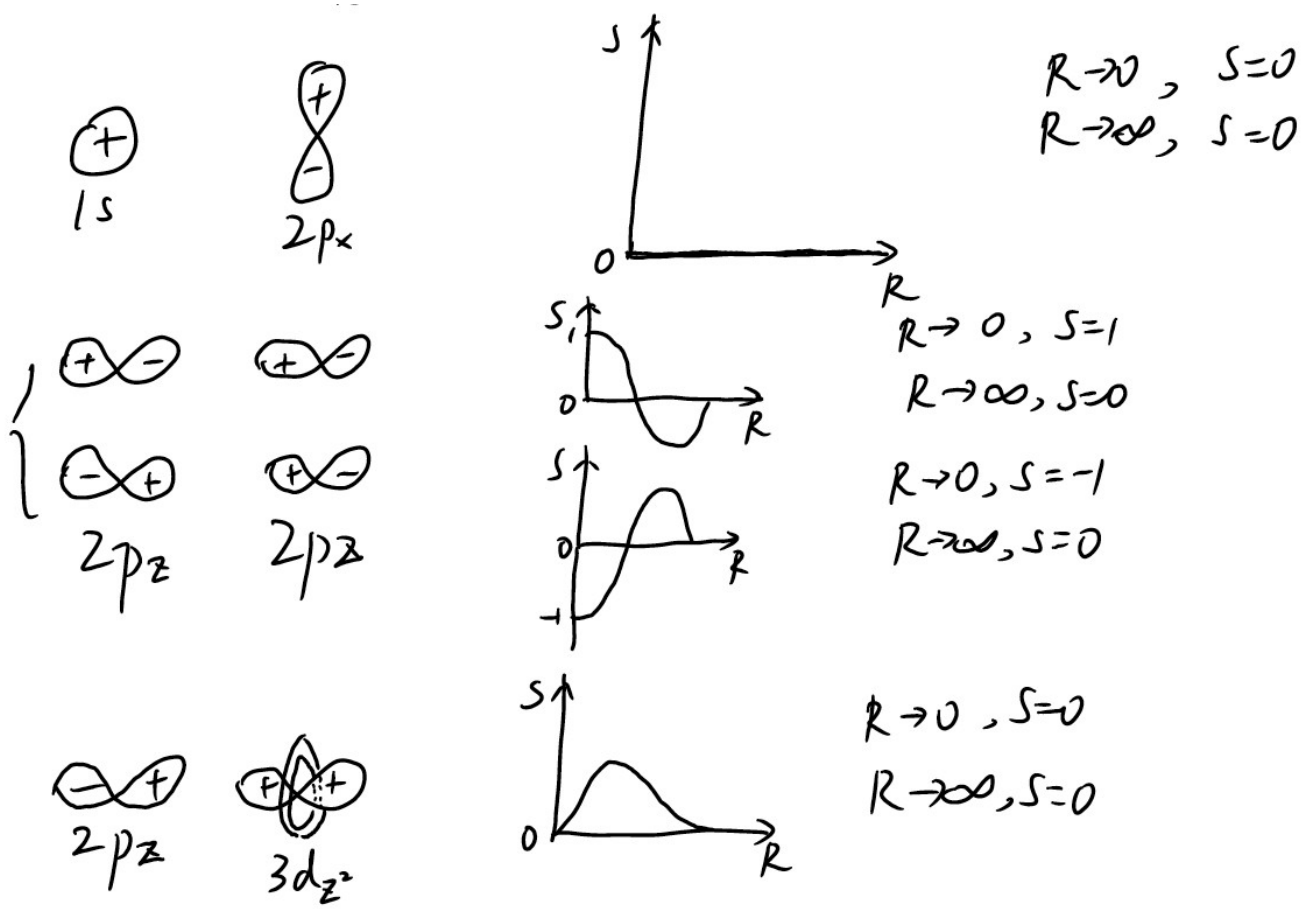
**Plot out the overlap as a function of the internuclear distance $R$. Qualitatively explain why it has the shape it has.**

In [0]:
```
import numpy as np
import matplotlib.pyplot as plt

r = np.linspace(0,12,100) # r=R/a0
P = (1+r+1/3*r**2)*np.exp(-r)

plt.plot(r,P)
plt.xlim(0)
plt.ylim(0)
plt.xlabel('Internuclear Distance $R/a0$')
plt.ylabel('Overlap S')
plt.title('The Overlap Between Two 1s Orbitals')
plt.show()
```




**2. The overlap functions for other pairs of orbitals are more complicated, but the general features are easily inferred. Neatly sketch the orbital overlap between a 1s orbital on A and 2p$_z$ orbital on B as a function $R$. Carefully indicate the limiting values as $R \rightarrow 0$ and $R \rightarrow \infty$.**

**3. Choose some other pair of atomic orbitals on A and B and sketch out their overlap as a function of $R$. Carefully indicate the limiting values as $R \to 0$ and $R \to \infty$.**




**4. What property besides overlap determines whether two atomic orbitals will form a bond?**

The similarity of the energies of the two atomic orbitals, ie the value of $\beta = \langle \phi_1 | \hat{f} | \phi_2 \rangle$

**5. A chemical bond is a compromise between the electrons trying to get close to both nuclei and the nuclei trying to stay apart. The function below captures this compromise as a function of internuclear distance, $R$. Plot the function for different values of the parameters $A$, $\alpha$, and $R_0$. Provide a physical interpretation of each of the parameters.**

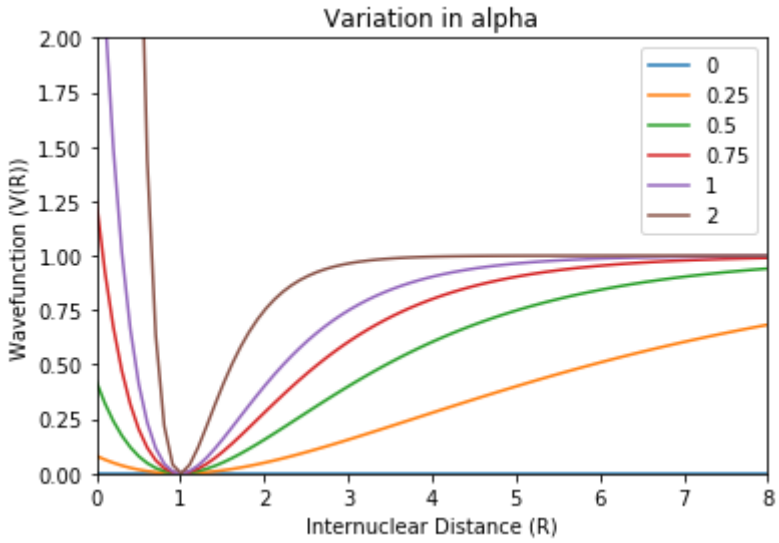
$$V(R) = A\left(1 - e^{(-\alpha(R-R_0))}\right)^2$$

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

A = 1
alpha = 1
R0 =[0,.25,.5,.75,1,2]
R = np.linspace(0,10,100)

for i in R0:
  V = A*(1-np.exp(-alpha*(R-i)))**2
  plt.plot(R,V, label = i)

plt.ylim(0,2)
plt.xlim(0,8)
plt.legend()
plt.xlabel('Internuclear Distance (R)')
plt.ylabel('Wavefunction (V(R))')
plt.title('Variation in R0')

plt.show()
print('R0 is the equilibrium bond distance.')
```

```
<Figure size 640x480 with 1 Axes>
```

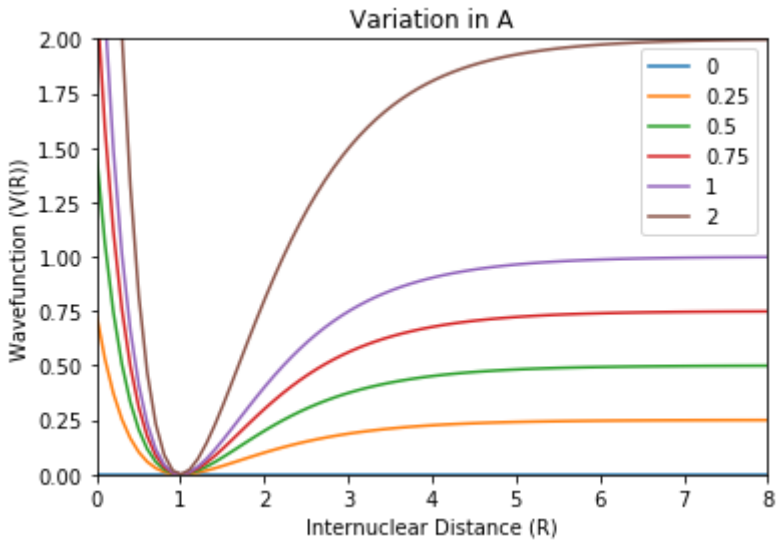
```
R0 is the equilibrium bond distance.
```

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt

A = 1
alpha = [0,.25,.5,.75,1,2]
R0 = 1
R = np.linspace(0,10,100)

for i in alpha:
  V = A*(1-np.exp(-i*(R-R0)))**2
  plt.plot(R,V, label = i)

plt.ylim(0,2)
plt.xlim(0,8)
plt.legend()
plt.xlabel('Internuclear Distance (R)')
plt.ylabel('Wavefunction (V(R))')
plt.title('Variation in alpha')

plt.show()
print('Alpha is the stiffness (spring constant) of the bond between the
 two atoms.')
```




```
Alpha is the stiffness (spring constant) of the bond between the two at
oms.
```

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt

A = [0,.25,.5,.75,1,2]
alpha = 1
R0 = 1
R = np.linspace(0,10,100)

for i in A:
  V = i*(1-np.exp(-alpha*(R-R0)))**2
  plt.plot(R,V, label = i)

plt.ylim(0,2)
plt.xlim(0,8)
plt.legend()
plt.xlabel('Internuclear Distance (R)')
plt.ylabel('Wavefunction (V(R))')
plt.title('Variation in A')

plt.show()
print('A is the difference in energy between a molecule and its atoms---
the bond dissociation energy.')
```




```
A is the difference in energy between a molecule and its atoms---the bo
nd dissociation energy.
```

### 6. For each pair, draw a Lewis dot structure. Indicate which bond is stronger in the pair, and give a very brief rationalization:

**(a) $H_2$ vs LiH**

**(b) $N_2$ vs $H_2$**

**(c) $N_2$ vs CO**

**(d) $H_2$ vs $He_2$**

a)

$$H : H$$
$$Li : H$$

$H_2$ has a stronger bond because the two hydrogens have similar energies.

b)

$$: N ::: N :$$
$$H : H$$

$N_2$ has a stronger bond since there are 3 bonds instead of just one.

c)

$$: N ::: N :$$
$$: C ::: O :$$

An argument for both structures can be made. There is not an agreed upon answer in the literature.

d)

$$H : H$$
$$: He \quad He :$$

$H_2$ has a stronger bond since $He_2$ doesn't have a bond.

## Computational chemistry.

### Today properties of a molecule are more often than not calculated rather than inferred. Quantitative molecular quantum mechanical calculations require specialized numerical solvers like Orca (https://orcaforum.kofo.mpg.de/app.php/portal). Following are instructions for using Orca with the Webmo (https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi) graphical interface.

## Now, let's set up your calculation (you may do this with a partner or partners if you choose):

1. Log into the Webmo server https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi (https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi) using "guest" as your username and password.
2. Select New Job-Creat New Job.
3. Use the available tools to sketch a molecule.
4. Use the right arrow at the bottom to proceed to the Computational Engines.
5. Select Orca
6. Select "Molecular Energy," "B3LYP" functional and the default def2-SVP basis set.
7. Select the right arrow to run the calculation.
8. From the job manager window choose the completed calculation to view the results.

The molecule you are to study depends on your last name. Choose according to the list:

- A-G: **CO**
- H-R: **BN**
- S-Z: **BeO**

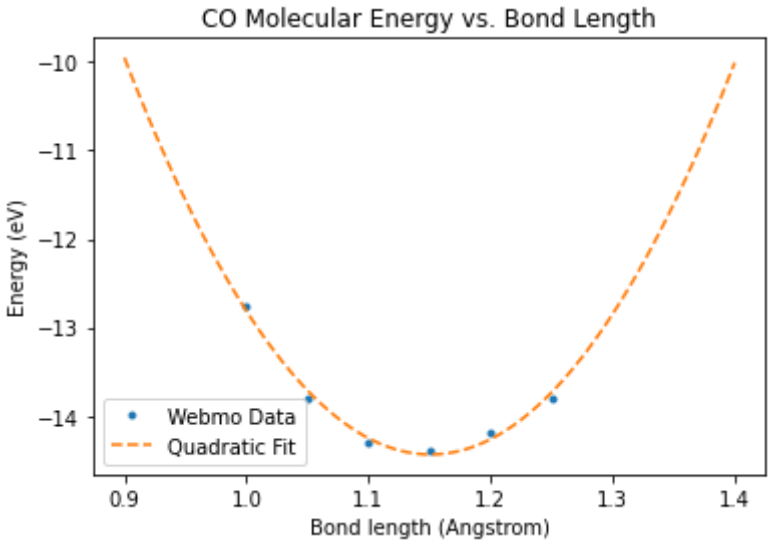
For your convenience, here are the total energies (in Hartree, 27.212 eV/Hartree) of the constituent atoms, calculated using the B3LYP DFT treatment of $\upsilon_{ee}$ and the def2-SVP basis set:

| Atom | Energy | Atom | Energy |
|---|---|---|---|
| B | –24.61703 | N | -54.51279 |
| Be | -14.64102 | O | -74.98784 |
| C | -37.79271 | F | -99.60655 |

## 7. Construct a potential energy surface for your molecule. Using covalent radii, guess an approximate equilbrium bond length, and use the Webmo editor to draw the molecule with that length. Specify the "Molecular Energy" option to Orga and the def2-SVP basis set. Calculate and plot out total molecular energy vs. bond distance in increments of 0.05 Å about your guessed minimum, including enough points to encompass the actual minimum. (You will find it convenient to subtract off the individual atom energies from the molecular total energy and to convert to more convenient units, like eV or kJ/mol.) By fitting the few points nearest the minimum, determine the equilibrium bond length. How does your result compare to literature?

In [0]:
```
# Carbon Monoxide
# From https://cccbdb.nist.gov/bondlengthmodel2.asp?method=12&basis=5, L
= 1.128 Angstrom
import numpy as np
import matplotlib.pyplot as plt

E_C = -37.79271 # Ha, energy of single C atom
E_O = -74.98784 # Ha, energy of single O atom
length = [1.00, 1.05, 1.10, 1.15, 1.2, 1.25] # Angstrom
E_CO = [-113.249199,-113.287858,-113.305895,-113.309135,-113.301902,-11
3.287408] # Ha, energy of CO
E_bond = [] # energy of CO bond
for i in E_CO:
    E_bond.append((i-E_C-E_O)*27.212) # eV, Energy[CO - C - O] = Energy
[bond]
fit = np.polyfit(length, E_bond, 2) # quadratic fit
print("Fitted result: E = %fx^2 + (%f)x + %f"%(fit[0],fit[1],fit[2]))

# Find E_min
x = np.linspace(0.9, 1.4, 100)
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
E_min_CO = min(z) # Find the minimum in energy array
print('E_min_CO = %feV.'%(E_min_CO))

# Plot E vs length
plt.plot(length, E_bond, '.', label='Webmo Data')
plt.plot(x, z, '--',label='Quadratic Fit')
plt.xlabel('Bond length (Angstrom)')
plt.ylabel('Energy (eV)')
plt.title('CO Molecular Energy vs. Bond Length')
plt.legend()
plt.show()

# Find equilbrium bond length
import sympy as sp
x = sp.symbols('x')
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
l = sp.solve(sp.diff(z,x),x)
print('L_equilibrium = %f A > 1.128 A (in literature).'%(l[0])) # equili
brium bond length
```

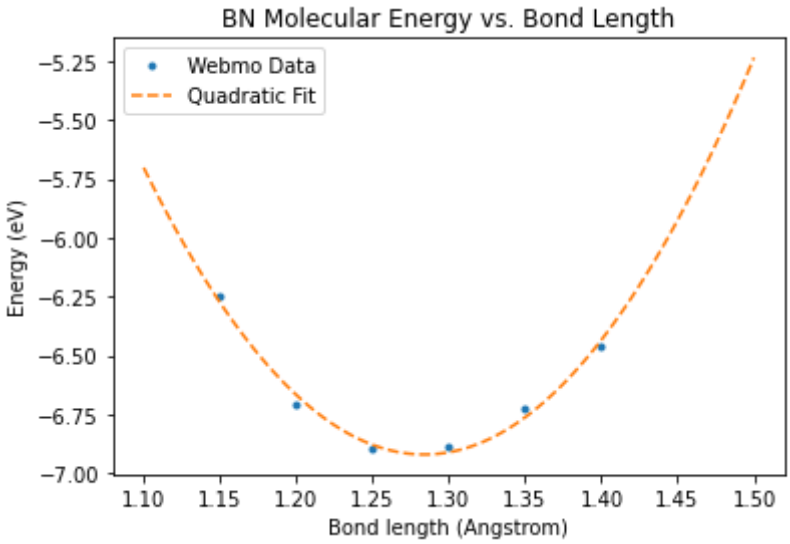
```
Fitted result: E = 71.304187x^2 + (-164.110637)x + 79.990690
E_min_CO = -14.436582eV.
```




```
L_equilibrium = 1.150778 A > 1.128 A (in literature).
```

In [0]:
```
#Boron Nitride
#From https://cccbdb.nist.gov/bondlengthmodel2.asp?method=12&basis=5, L=
1.325 Angstrom
import numpy as np
import matplotlib.pyplot as plt

E_B = -24.61703 # Ha, energy of single B atom
E_N = -54.51279 # Ha, energy of single N atom
length = [1.15, 1.2, 1.25, 1.3, 1.35, 1.4] # Angstrom
E_BN = [-79.359357,-79.376368,-79.383355,-79.382896,-79.377003,-79.36723
6] # Ha, energy of BN
E_bond = [] # energy of BN bond
for i in E_BN:
    E_bond.append((i-E_B-E_N)*27.212)
fit = np.polyfit(length, E_bond, 2) # quadratic fit
print("Fitted result: E = %fx^2 + (%f)x + %f"%(fit[0],fit[1],fit[2]))

# Find E_min
x = np.linspace(1.1, 1.5, 100)
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
E_min_BN = min(z) # Find the minimum in energy array
print('E_min_BN = %feV.'%(E_min_BN))

# Plot E vs length
plt.plot(length, E_bond, '.', label='Webmo Data')
plt.plot(x, z, '--',label='Quadratic Fit')
plt.xlabel('Bond length (Angstrom)')
plt.ylabel('Energy (eV)')
plt.title('BN Molecular Energy vs. Bond Length')
plt.legend()
plt.show()

# Find equilbrium bond length
import sympy as sp
x = sp.symbols('x')
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
l = sp.solve(sp.diff(z,x),x)
print('L_equilibrium = %f A < 1.325 A (in literature).'%(l[0])) # equili
brium bond length
```

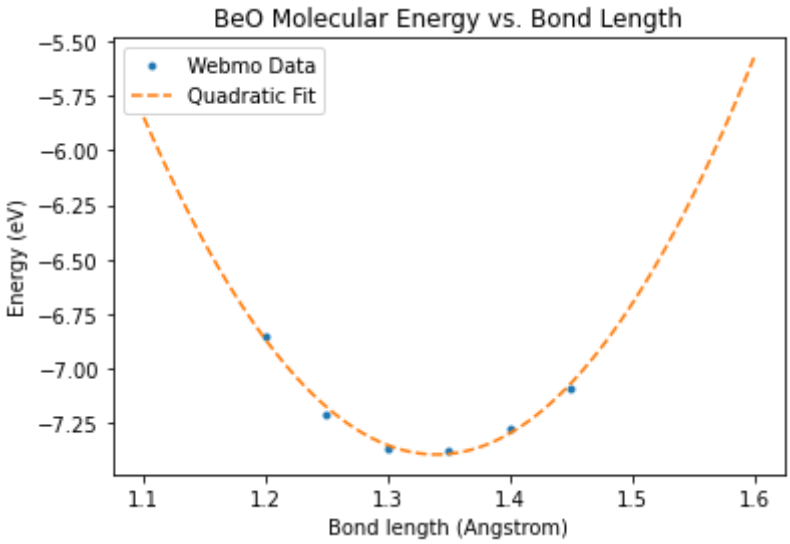
```
Fitted result: E = 36.038407x^2 + (-92.533003)x + 52.477192
E_min_BN = -6.920107eV.
```




```
L_equilibrium = 1.283811 A < 1.325 A (in literature).
```

In [0]:
```
#Berrylium Oxide
#From https://cccbdb.nist.gov/bondlengthmodel2.asp?method=12&basis=5, L
 = 1.331 Angstrom
import numpy as np
import matplotlib.pyplot as plt

E_Be = -14.64102 # Ha
E_O = -74.98784 # Ha
length = [1.2, 1.25, 1.3, 1.35, 1.4, 1.45] # Angstrom
E_BeO = [-89.880569,-89.893740,-89.899599,-89.899934,-89.896149,-89.8893
35] # Ha, energy of BeO
E_bond = [] # energy of BeO bond
for i in E_BeO:
    E_bond.append((i-E_Be-E_O)*27.212)
fit = np.polyfit(length, E_bond, 2) # quadratic fit
print("Fitted result: E = %fx^2 + (%f)x + %f"%(fit[0],fit[1],fit[2]))

# Find E_min
x = np.linspace(1.1, 1.6, 100)
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
E_min_BeO = min(z) # Find the minimum in energy array
print('E_min_BeO = %feV.'%(E_min_BeO))

# Plot E vs length
plt.plot(length, E_bond, '.', label='Webmo Data')
plt.plot(x, z, '--',label='Quadratic Fit')
plt.xlabel('Bond length (Angstrom)')
plt.ylabel('Energy (eV)')
plt.title('BeO Molecular Energy vs. Bond Length')
plt.legend()
plt.show()

# Find equilbrium bond length
import sympy as sp
x = sp.symbols('x')
z = fit[0]*x**2 + fit[1]*x + fit[2] # from result above
l = sp.solve(sp.diff(z,x),x)
print('L_equilibrium = %f A > 1.331 A (in literature).'%(l[0])) # equili
brium bond length
```

```
Fitted result: E = 26.920637x^2 + (-72.138820)x + 40.931304
E_min_BeO = -7.395854eV.
```




```
L_equilibrium = 1.339842 A > 1.331 A (in literature).
```

**8. Use the quadratic fit from Question 8 to determine the harmonic vibrational frequency of your molecule, in cm$^{-1}$. Recall that the force constant is the second derivative of the energy at the minimum, and that the frequency (in wavenumbers) is related to the force constant according to**

$$\tilde{\nu} = \frac{1}{2\pi c}\sqrt{\frac{k}{\mu}}$$

In [0]:
```
print('CO Molecule:')
J = 1.6022e-19 # J, 1 eV = 1.6022e-19 J
L = 1e-10 # m, 1 angstrom = 1e-10 m

# k [=] Energy/Length^2
k_CO = 2*71.30418671*J/L**2 # J/m**2
c = 2.99792e8 # m/s
m_C = 12.0107*1.6605e-27 # kg
m_O = 15.9994*1.6605e-27 # kg
mu_CO = m_C*m_O/(m_C+m_O) # kg, reduced mass

nu_CO = 1/(2*np.pi*c)*np.sqrt(k_CO/mu_CO)/100 # cm^-1, wavenumber
print('The harmonic vibrational frequency is %f cm^-1.'%(nu_CO))
```

```
CO Molecule:
The harmonic vibrational frequency is 2377.567475 cm^-1.
```

In [0]:
```
print('BN Molecule:')
J = 1.6022e-19 # J, 1 eV = 1.6022e-19 J
L = 1e-10 # m, 1 angstrom = 1e-10 m

# k [=] Energy/Length^2
k_BN = 2*36.0384*J/L**2 # J/m**2
c = 2.99792e8 # m/s
m_B = 10.811*1.6605e-27 # kg
m_N = 14.0067*1.6605e-27 # kg
mu_BN = m_B*m_N/(m_B+m_N) # kg, reduced mass

nu_BN = 1/(2*np.pi*c)*np.sqrt(k_BN/mu_BN)/100 # cm^-1, wavenumber
print('The harmonic vibrational frequency is %f cm^-1.'%(nu_BN))
```

```
BN Molecule:
The harmonic vibrational frequency is 1792.324670 cm^-1.
```

In [0]:
```
print('BeO Molecule:')
J = 1.6022e-19 # J, 1 eV = 1.6022e-19 J
L = 1e-10 # m, 1 angstrom = 1e-10 m

# k [=] Energy/Length^2
k_BeO = 2*26.920637*J/L**2 # J/m**2
c = 2.99792e8 # m/s
m_Be = 9.01218*1.6605e-27 # kg
m_O = 15.9994*1.6605e-27 # kg
mu_BeO = m_Be*m_O/(m_Be+m_O) # kg, reduced mass

nu_BeO = 1/(2*np.pi*c)*np.sqrt(k_BeO/mu_BeO)/100 # cm^-1, wavenumber
print('The harmonic vibrational frequency is %f cm^-1.'%(nu_BeO))
```

```
BeO Molecule:
The harmonic vibrational frequency is 1593.677593 cm^-1.
```

## 9. Use your results to determine the zero-point-corrected bond energy of your molecule. How does this model compare with the experimental value?

In [0]:
```
# Get experimental vibrational zero-point energy from NIST database: htt
ps://cccbdb.nist.gov/exp1x.asp
nu_CO_exp = 1084.9 # cm^-1
nu_BN_exp = 760.2 # cm^-1
nu_BeO_exp = 728.5 # cm^-1
```

In [0]:
```
print('CO Molecule:')
# Note: E_ZPC = E_min + ZPE_harmonic_oscillator
h = 6.62607e-34
NA = 6.02214e23
J = 1.6022e-19 # eV to J
E_min_CO = (-16.300903*J)*NA/1000 # converted from eV to kJ/mol from pro
blem 8

# Calculations
E0_CO = (0.5*h*nu_CO*100*c)*NA/1000 # kJ/mol, ZPE harmonic oscillator
EB_CO = E_min_CO + E0_CO # kJ/mol, ZPC bond energy
# Experiments
E0_CO_exp = (0.5*h*nu_CO_exp*100*c)*NA/1000
EB_CO_exp = E_min_CO + E0_CO_exp
print('|E_ZPC| = %f kJ/mol < %f kJ/mol.'%(-EB_CO,-EB_CO_exp))
```

```
CO Molecule:
|E_ZPC| = 1558.599791 kJ/mol < 1566.331647 kJ/mol.
```

In [0]:
```
print('BN Molecule:')
# Note: E_ZPC = E_min + ZPE_harmonic_oscillator
h = 6.62607e-34
NA = 6.02214e23
J = 1.6022e-19 # eV to J
E_min_BN = (-4.633537*J)*NA/1000 # converted from eV to kJ/mol from prob
lem 8

# Calculations
E0_BN = (0.5*h*nu_BN*100*c)*NA/1000 # kJ/mol, ZPE harmonic oscillator
EB_BN = E_min_BN + E0_BN # kJ/mol, ZPC bond energy
# Experiments
E0_BN_exp = (0.5*h*nu_BN_exp*100*c)*NA/1000
EB_BN_exp = E_min_BN + E0_BN_exp
print('|E_ZPC| = %f kJ/mol < %f kJ/mol.'%(-EB_BN,-EB_BN_exp))
```

```
BN Molecule:
|E_ZPC| = 436.354356 kJ/mol < 442.527822 kJ/mol.
```

In [0]:
```
print('BeO Molecule:')
# Note: E_ZPC = E_min + ZPE_harmonic_oscillator
h = 6.62607e-34
NA = 6.02214e23
J = 1.6022e-19 # eV to J
E_min_BeO = (-5.850784*J)*NA/1000 # converted from eV to kJ/mol from pro
blem 8

# Calculations
E0_BeO = (0.5*h*nu_BeO*100*c)*NA/1000 # kJ/mol, ZPE harmonic oscillator
EB_BeO = E_min_BeO + E0_BeO # kJ/mol, ZPC bond energy
# Experiments
E0_BeO_exp = (0.5*h*nu_BeO_exp*100*c)*NA/1000
EB_BeO_exp = E_min_BeO + E0_BeO_exp
print('|E_ZPC| = %f kJ/mol < %f kJ/mol.'%(-EB_BeO,-EB_BeO_exp))
```

```
BeO Molecule:
|E_ZPC| = 554.990706 kJ/mol < 560.165609 kJ/mol.
```

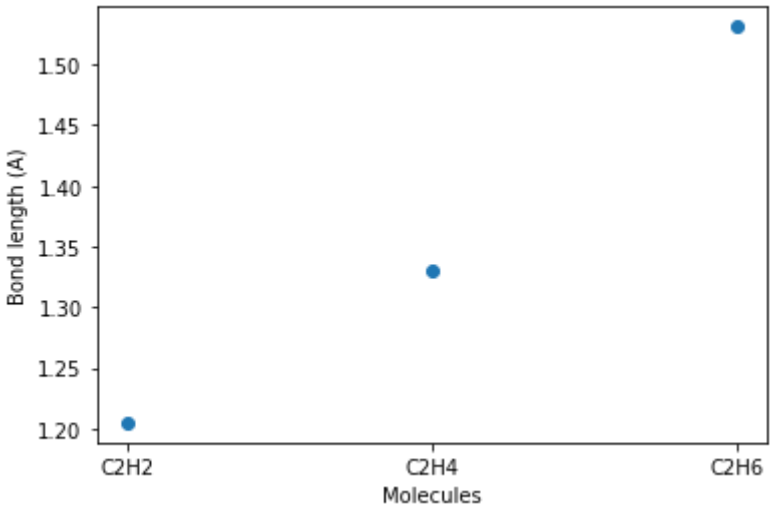
# Computational chemistry, part deux

## Diatomics are a little mundane. These same methods can be used to compute the properties of much more complicated things. As example, the OQMD database http://oqmd.org/ (http://oqmd.org/) contains results for many solids. We don't have time to get this complicated in class, but at least you can compute properties of some molecules.

## 10. Working with some of your classmates, compute the equilibrium structures of $C_2H_6$, $C_2H_4$, and $C_2H_2$. Compare their equilibrium C-C bond lengths. Do they vary in the way you expect?

1. Log into the Webmo server https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi (https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi) using "guest" as your username and password.
2. Select New Job-Creat New Job.
3. Use the available tools to sketch a molecule. Make sure the bond distances and angles are in a plausible range.
4. Use the right arrow at the bottom to proceed to the Computational Engines.
5. Select Orca
6. Select "Geometry optimization," “B3LYP” functional and the default def2-SVP basis set.
7. Select the right arrow to run the calculation.
8. From the job manager window choose the completed calculation to view the results.

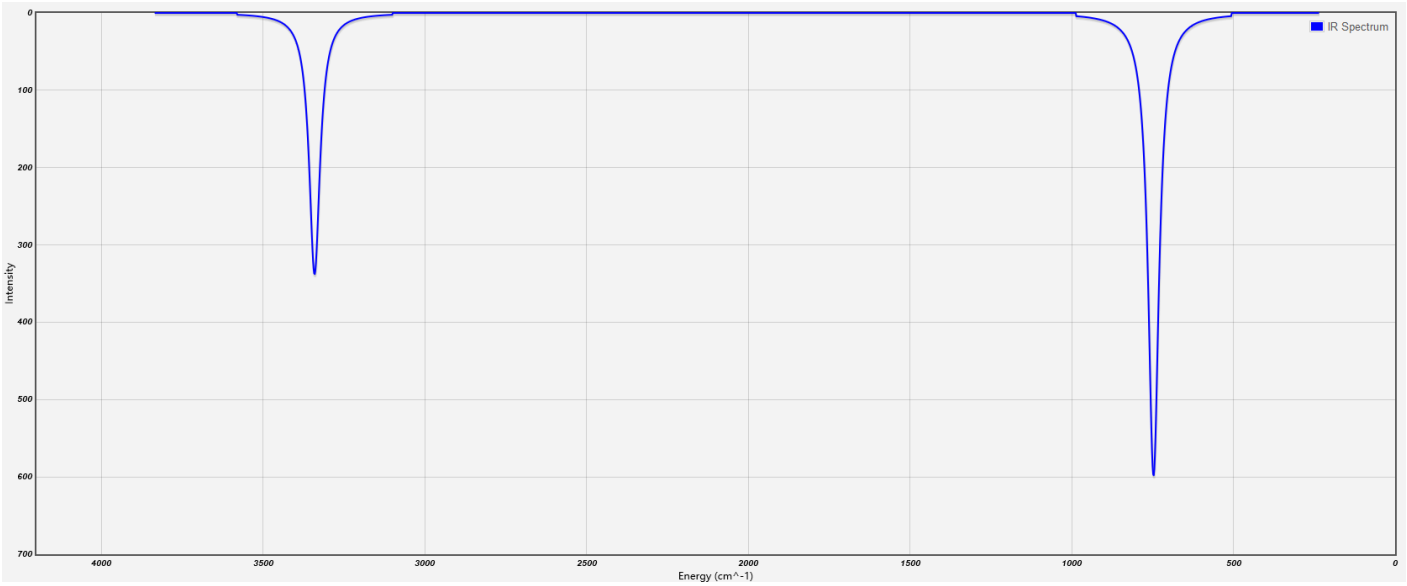
In [0]:
```
C2H6 = 1.531 # Angstrom
C2H4 = 1.331 # Angstrom
C2H2 = 1.205 # Angstrom

import matplotlib.pyplot as plt
plt.scatter([0,1,2],[C2H2,C2H4,C2H6])
plt.xlabel('Molecules')
plt.ylabel('Bond length (A)')
plt.xticks(np.arange(3), ('C2H2','C2H4','C2H6'))
plt.show()
```



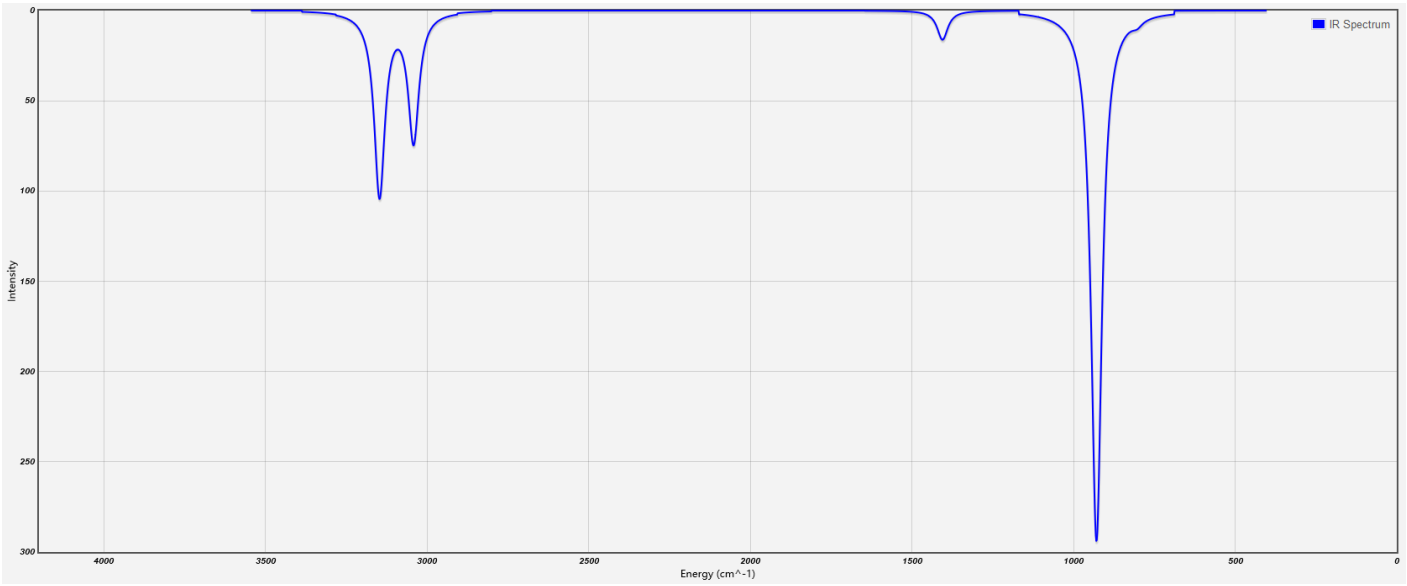

## 11. Compute the corresponding vibrational spectra. Could you distinguish these molecules by their spectra?

1. Log into the Webmo server https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi (https://www.webmo.net/demoserver/cgi-bin/webmo/login.cgi) using "guest" as your username and password.
2. Select the job with the optimized geometry and open it.
3. Use the right arrow at the bottom to proceed to the Computational Engines.
4. Select Orca
5. Select "Vibrational frequency," “B3LYP” functional and the default def2-SVP basis set.
6. Select the right arrow to run the calculation.
7. From the job manager window choose the completed calculation to view the results.

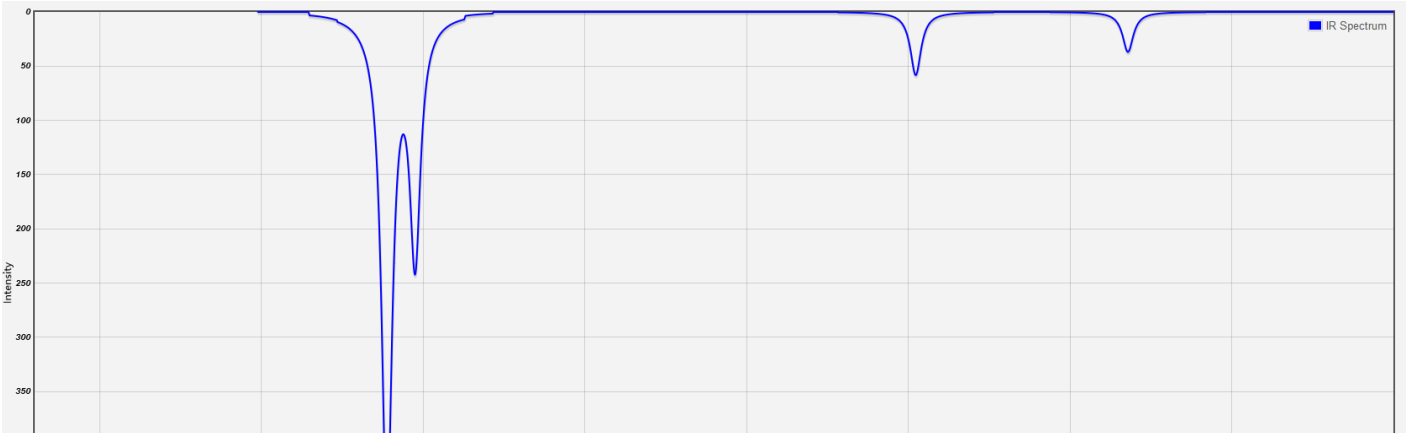
C2H2




C2H4




C2H6




## 12. Compute the structure and energy of $H_2$. Use it to compare the energies to hydrogenate acetylene to ethylene and ethylene to ethane. Which is easier to hydrogenate? Can you see why selective hydrogenation of acetylene to ethylene is difficult to do?

In [0]:
```
E_H2 = -1.16646206791 # Ha
E_C2H2 = -77.3256461775 # Ha, acetylene
E_C2H4 = -78.5874580928  # Ha, ethylene
E_C2H6 = -79.8304174812 # Ha, ethane

E_rxn1 = (E_C2H4 - E_C2H2 - E_H2)*2625.50 # kJ/mol, H2 + C2H2 -> C2H4
E_rxn2 = (E_C2H6 - E_C2H4 - E_H2)*2625.50 # kJ/mol, H2 + C2H4 -> C2H6
print("E_rnx1 = %f kJ/mol, E_rnx2 = %f kJ/mol"%(E_rxn1, E_rxn2))
```

```
E_rnx1 = -250.341024 kJ/mol, E_rnx2 = -200.843715 kJ/mol
```

Exothermic to add the first H2 to acetylene (to make ethylene) and to add second H2 to make ethane. A selective hydrogenation catalyst must help the first to happen and not the second.

In [0]:
```
```