CHEM532 Programming Project #1: Harmonic Vibrational Analysis

Francesco A. Evangelista

February 5, 2014

Introduction

In this programming project you will read data from a previous computation (optimized geometry, Hessian) and use it to compute the harmonic vibrational frequencies of a molecule.

Program outline

1. Read the molecular geometry and store it in memory.

   Information regarding the molecular geometry is contained in the file `geometry`. This file has the following structure:

   ```
   <Number of atoms>
   <Comment>
   <Element symbol 1>  <x1> <y1> <z1>
   <Element symbol 2>  <x2> <y2> <z2>
   .
   <Element symbol N>  <xN> <yN> <zN>
   ```

   The first line of the `geometry` file contains an integer that specifies the total number of atoms (M). The second line can either be blank or contain a comment. Lines 3 to $M+2$ specify the symbol (H, C, Li, etc.) and the coordinates (X_A, Y_A, Z_A) of each atom A contained in the molecule. Notice that the coordinates are given in **atomic units**.

To check that you read the geometry correctly perform the following:

a) print the element symbol and coordinates of each atom.

b) print the interatomic distances for all pairs of atoms A and B:

$$R_{AB} = |\mathbf{R}_A - \mathbf{R}_B| = \sqrt{(\mathbf{R}_A - \mathbf{R}_B) \cdot (\mathbf{R}_A - \mathbf{R}_B)}. \quad (1)$$

c) compute the nuclear-nuclear repulsion energy, which in atomic units reads:

$$V_{\mathrm{NN}} = \sum_{A<B}^{M} \frac{Z_A Z_B}{|\mathbf{R}_A - \mathbf{R}_B|}. \quad (2)$$

After processing the geometry file, use the element symbol to determine the mass (M_A) of each atom A. For this purpose use the provided file `label_to_mass.py` which can be downloaded from the course web page. This file can be used to retrieve the relative mass (with respect to that of C^{12}) of an atom A given its label. For example, in Python:

```
# import label_to_mass.py (in the same directory as your script)
from label_to_mass import label2mass

# get the mass of hydrogen
label2mass("H")  # > 1.007825032

# get the mass of carbon
label2mass('c') # > 12.0

# get the mass of iron, even if written in a weird way
label2mass('fE') # > 55.934937475
```

2. Read the Hessian and store it in memory.

The Hessian matrix in **Cartesian coordinates**:

$$H_{\alpha\beta} = \left. \frac{\partial^2 E(\mathbf{R})}{\partial R_\alpha \partial R_\beta} \right|_{\mathbf{R}_0} \quad \text{for } \alpha, \beta = 1, \ldots, 3M. \quad (3)$$

is contained in the file `hessian`. Here we used the notation R_α to indicate a generic component of the vector of atomic positions $\mathbf{R}$, where $\mathbf{R} = (X_1, Y_1, Z_1, X_2, Y_2, Z_2, \ldots, X_{3M}, Y_{3M}, Z_{3M})$. Thus for example, $R_5 = Y_2$ and the mass corresponding to this coordinate is the one of atom number 2. The `hessian` file stores the $3M \times 3M$ entries of the Hessian matrix and has the following structure:

$$\begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} & \cdots & H_{1,3M} \\ H_{2,1} & H_{2,2} & H_{2,3} & \cdots & H_{2,3M} \\ H_{3,1} & H_{3,2} & H_{3,3} & \cdots & H_{3,3M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ H_{3M,1} & H_{3M,2} & H_{3M,3} & \cdots & H_{3M,3M} \end{bmatrix}. \tag{4}$$

Read this data and store it as a matrix using the same format of the file. After reading the Hessian matrix, **print it to the output file**. Notice that the Hessian matrix is given in units of hartree $\times$ bohr^{-2}.

To store, and do computations on matrices and arrays use the library `numpy`. With `numpy` it is very easy to create a matrix:

```
# Using the function "import *" allows us to access all the functions
# available in numpy without having to write numpy. in front of them
from numpy import *
from numpy.linalg import *  # Linear algebra functions

# Create a matrix of dimensions 3M x 3M
H = zeros( (3 * M, 3 * M) )

# Fill the matrix with the product "i x j"
for i in xrange(3 * M):
    for j in xrange(3 * M):
        H[i][j] = i * j

# Print H
print H

# Call the function eigh to diagonalize a real symmetric matrix
lambda, L = eigh(H)
print lambda # prints a vector containing the eigenvalues
print L # prints a matrix containing the eigenvectors

# Multiply two matrices: C = A x B
C = dot(A,B)
```

3. Compute the mass-weighted Hessian:

$$\tilde{H}_{\alpha\beta} = \frac{H_{\alpha\beta}}{\sqrt{M_\alpha M_\beta}}. \tag{5}$$

   Make sure that the mass-weighted Hessian is computed in **atomic units**. After computing $\tilde{H}_{\alpha\beta}$, **print it to the output file**.

4. Compute the mass-weighted Hessian using matrix multiplication:

   Form the diagonal matrix $\mathbf{W}^{-1/2}$ with elements:

$$W_{\alpha\beta} = \delta_{\alpha\beta}\frac{1}{\sqrt{M_\alpha}}. \tag{6}$$

   Then form the mass-weighted Hessian as $\tilde{\mathbf{H}} = \mathbf{W}^{-1/2}\mathbf{H}\mathbf{W}^{-1/2}$.

5. Diagonalize the mass-weighted Hessian:

$$\tilde{\mathbf{H}}\mathbf{L} = \mathbf{L}\boldsymbol{\lambda}, \tag{7}$$

   where $\mathbf{L}$ is the eigenvector matrix and $\boldsymbol{\lambda}$ is the eigenvalue matrix (which is diagonal, and is stored by the Python function `eigh` as a vector).

6. Determine the harmonic vibrational frequencies (ω_i):

$$\omega_i = C \times \sqrt{\lambda_i} \tag{8}$$

   The C in the above equation is a conversion factor. Report the harmonic vibrational frequency in cm^{-1} and MHz. Use the conversion factors provided in the file `phys_constants.py` to help you. In the output, identify the imaginary frequencies corresponding to $\lambda_i < 0$.

7. The Cartesian displacements $\delta\mathbf{R}$ can be related to the internal coordinate displacements via:

$$\delta\mathbf{R} = \mathbf{W}^{1/2}\mathbf{L}\mathbf{Q}. \tag{9}$$

   Form the matrix $\mathbf{T} = \mathbf{W}^{1/2}\mathbf{L}$ and print it. This can be used to convert from internal coordinates to Cartesian coordinates.

8. (Bonus) A unit displacement along the normal coordinate α, corresponds to the vector $\mathbf{Q}^{(\alpha)} = (0, \ldots, 0, 1_\alpha, 0, \ldots)$, which has all components equal to zero

except for the α-th normal mode. The normal mode displacement $\mathbf{Q}^{(\alpha)}$ corresponds to the Cartesian displacement:

$$\delta R_{\beta}^{(\alpha)} = T_{\beta\alpha}. \tag{10}$$

For each normal coordinate α, form the Cartesian displacement vector $\delta\mathbf{R}^{(\alpha)}$ and normalize it:

$$\delta\tilde{\mathbf{R}}^{(\alpha)} = \frac{\delta\mathbf{R}^{(\alpha)}}{\sqrt{\delta\mathbf{R}^{(\alpha)} \cdot \delta\mathbf{R}^{(\alpha)}}} \tag{11}$$

The square of each component of $\delta\tilde{\mathbf{R}}^{(\alpha)}$ can be interpreted as the weight of Cartesian displacement in a normal mode, w_{β}:

$$w_{\beta}^{(\alpha)} = \left[\delta\tilde{R}_{\beta}^{(\alpha)}\right]^2. \tag{12}$$

For each normal coordinate α, identify three Cartesian coordinates β with the largest values of $w_{\beta}^{(\alpha)}$, print $w_{\beta}^{(\alpha)}$ (as a percentage) and the atom and Cartesian coordinate (X, Y, or Z) corresponding to the coordinate β. For example:

```
Mode  11:  47.6% 3-Y(H) +  47.1% 2-Y(O) +   2.5% 3-X(H)
```

meaning that for mode 11 two coordinates contribute the most: a Y displacement of the second atom (hydrogen, 47.6%), and a Y displacement of the first atom (oxygen, 47.1%).