Solving a System of Quantum Anharmonic Oscillators in a Harmonic Basis

C.B.
cbaiz[at]umich.edu

June 8, 2009

1 Brief Description

This document describes the code used for generating and diagonalizing the vibrational Hamiltonian for a system of anharmonic oscillators. The code supports an arbitrary number of oscillators and basis functions with harmonic, bilinear, third-order and semi-diagonal fourth-order force constants. The code is written in MATLAB and has been tested in versions 7.6 and later. The program was optimized for efficient memory usage by storing all large matrices as sparse arrays and utilizing an iterative approach to computing the eigenvectors and eigenvalues of the Hamiltonian. This code is released under the GPLv3 license and is free software. The functions "combn.m" and "tRow.m" were obtained from *www.matlabcentral.com* and are provided under the BSD license.

2 Program Inputs

All inputs are set in the file "initialize_parameters.m". A set of sample inputs and force constants is provided for testing. These force constants correspond to the test molecule Rhodium(acetyl acetonato) dicarbonyl computed with density functional theory.

Second-order force constants:
forces.second (2D Array of floats)
Second-order force constants ϕ_{ij} are supplied in a two-dimensional N-by-N array where N is the number of oscillators. The diagonal elements represent the harmonic ϕ_{ii}, and the off-diagonal elements represent ϕ_{ij} "bilinear coupling" constants. Alternatively, if only diagonal elements are needed, the input can be a one-dimensional array containing the ϕ_{ii} constants.

Third- and fourth-order force constants:
forces.third and *forces.fourth* (3D Array of floats)
These are supplied in three dimensional N-by-N-by-N arrays containing entries of the type ϕ_{ijk} (third order) and ϕ_{ijkk} (semi-diagonal fourth-order). If these arrays are not set the program will assume these force constants are zero and only use the second-order terms.

Modes to be included:
params.ham.mode (1D Array of positive integers)
The array of force constants may be larger than the number of oscillators, so modes $i, j, k...$ used in the diagonalization must be specified. For example, *forces.second* may be a 10x10 array populated with all second-order force constants for 10 oscillators, if one wishes to include only oscillators 2 and 3 and ignore all the other force constants, *params.ham.mode* can be set to equal [2 3]. The code is set-up in this manner because electronic structure computations provide anharmonic force constants for all the vibrational modes in the molecule (see below), but usually one is only interested in a subset of the modes and generating a full Hamiltonian with a large number of modes requires large amounts of memory.

Number of basis functions:
params.ham.numBasis (Positive integer)
This variable is a single positive integer containing the number of basis functions to be used in each of the modes. Note that the size of the Hamiltonian matrix grows exponentially with respect to the number of modes. The size of this matrix is the number of basis to the power of the number of modes.

Non-uniform basis:
params.ham.nonuniformbasis (Boolean) and *params.ham.excitations* (1D Array of Positive Integers)
nonuniformbasis is a boolean (T/F), that switches whether a different number of basis functions is to be used in each of the different modes. When this variable is set to "true" the variable *params.ham.excitations* needs to be set. This variable is a one-dimensional array containing the number of basis to be included in each of the modes as defined in *params.ham.mode*.

Number of eigenvectors to compute:
params.ham.numvectIterate (Positive Integer)
To avoid direct diagonalization the program uses an iterative approach to computing the eigenvalues and eigenvectors. This variable sets the number of eigenvectors to compute. The value of this variable will determine which excited states are computed. To minimize memory requirements this should be set to the lowest possible value depending on how many energy levels one needs to compute.

3 Program Outputs

The main output of the program is a structure labeled *output*. This contains the following substructures: *evects* and *evals* contain the computed eigenvalues (energy levels) and eigenvectors of the Hamiltonian (wavefunctions). The substructure *basis* contains the basis vector used to construct the Hamiltonian. All these outputs are automatically written to disk and saved under the filename "output.mat". If this file exists it will be overwritten.

4 Description of the Main Functions

The main functions are located in the program directory with auxiliary functions located in the "./functions" subdirectory. The script that calls all the functions is "main_script.m". The program inputs are set in "initialize_parameters.m" and this script is directly called from "main_script.m".

errorCheck.m

Checks the input for common errors and displays the corresponding error messages.

generateHamiltonianParams.m

Generates parameters containing the mode combinations and force-constants for the modes specified in the input.

generateBasis.m

As the name implies, this function generates the basis vectors using the specified modes.

generateOperatorMatrices.m

This function takes the basis set and generates the position and momentum matrices for each of the oscillators.

Hamiltonian.m

This is the main engine that generates the Hamiltonian matrix by multiplying position and momentum matrices for each of the oscillators with the corresponding force constants. The eigenvalues and eigenvectors are also computed inside this function.

DisplayOutput.m

This function simply displays the output in a somewhat human-readable form. This will only work when the anharmonic wavefunctions resemble the parent harmonic states and will not output meaningful results if the eigenvectors are highly-mixed.

5 Sample Input Provided

A script that reads force constants from a Gaussian03 output file is provided along with a sample output file for Rhodium(acetyl acetonato) dicarbonyl. The electronic structure computation of anharmonic force constants can be set in Gaussian03 by simply adding the following command in the input's route section "Freq(Anharmonic)", further information is provided in the manual. The function "ReadGaussianOutput.m" takes the path to the Gaussian file and outputs the force constants contained in the file in a format that can be directly used as input for the diagonalization code. Usage should be: [forces] = ReadGaussianOutput('Sample_Output.out');

6 Additional Resources

The following references provide useful information regarding the computation of anharmonic force constants from electronic structure methods.

1. Schneider, W.; Thiel, W. *Chemical Physics Letters* **1989**, 157, 367.
2. Dressler, S.; Thiel, W. *Chemical Physics Letters* **1997**, 273, 71.
3. Barone, V. *Journal of Chemical Physics* **2004**, 120, 3059.

4. Califano, S. Vibrational states; John Wiley and Sons: London, *1976*.