2D-IR simulation: Theory and computational detail

You Li

April 28, 2023

Two dimensional infrared spectra (2d-IR) is a very powerful spectroscopy method to investigate vibration coupling, spectral diffusion and chemical exchange. To simulate a 2d-IR , we can use QVP (quantum vibrational perturbation) method which can calculate chromophore vibration frequencies quantum-mechanically while other freedom degrees were still simulated by classical Molecular Dynamics. Apparently, QVP method combined both accuracy and efficiency for semi-classical IR line-shape simulation. Besides the QVP program (qvptk) ,we also need to transfer time-dependent frequencies to IR line-shape. Here, I use c++ to write 2d-IR simulation program, including different separated module and some response calculation choice. The python package can be easily compiled by cython, which provides a good combination of c++ and python.

1 Theory of 2d-IR simulation

Unlike linear IR spectra, 2d-IR have to calculate response function, and then change it from time-domain to frequency-domain. There's different situation we need to discuss.

1.1 Two transmission system for ultrafast three-pulse echo spectrum

Ultrafast three-pulse echo spectrum is one of third-order nonlinear spectroscoy. The basic formula is [1]:

$$I(\omega_1, t_2, \omega_3) = \mathrm{Re}\int_0^\infty dt_1 \int_0^\infty dt_3 [e^{-i\omega_1 t_1 + i\omega_3 t_3} R_r(t_3, t_2, t_1) + e^{i\omega_1 t_1 + i\omega_3 t_3} R_{nr}(t_3, t_2, t_1)] \tag{1}$$

Where R_r and R_{nr} means rephasing and non-rephasing response function:

$$\begin{aligned} R_r(t_3, t_2, t_1) &= R_1(t_3, t_2, t_1) + R_2(t_3, t_2, t_1) + R_3(t_3, t_2, t_1) \\ R_{nr}(t_3, t_2, t_1) &= R_4(t_3, t_2, t_1) + R_5(t_3, t_2, t_1) + R_6(t_3, t_2, t_1) \end{aligned} \tag{2}$$

Without Condon approximation, we can write R_i as:

$$\begin{aligned} R_1(t_3, t_2, t_1) &= R_2(t_3, t_2, t_1) \\ &= \langle \mu_{10}(0)\mu_{10}(t_1)\mu_{10}(t_1+t_2)\mu_{10}(t_1+t_2+t_3)\phi_r^{(1)}(t_3, t_2, t_1)\rangle \end{aligned} \tag{3}$$

$$\begin{aligned} R_4(t_3, t_2, t_1) &= R_5(t_3, t_2, t_1) \\ &= \langle \mu_{10}(0)\mu_{10}(t_1)\mu_{10}(t_1+t_2)\mu_{10}(t_1+t_2+t_3)\phi_{nr}^{(1)}(t_3, t_2, t_1)\rangle \end{aligned} \tag{4}$$

$$R_3(t_3,t_2,t_1) = -\langle \mu_{10}(0)\mu_{10}(t_1)\mu_{21}(t_1+t_2)\mu_{21}(t_1+t_2+t_3)\phi_r^{(2)}(t_3,t_2,t_1)\rangle \tag{5}$$

$$R_6(t_3,t_2,t_1) = -\langle \mu_{10}(0)\mu_{10}(t_1)\mu_{21}(t_1+t_2)\mu_{21}(t_1+t_2+t_3)\phi_{nr}^{(2)}(t_3,t_2,t_1)\rangle \tag{6}$$

ϕ is called dephasing-induced line broadening factor[2], but I'd like to call it as semi-classical frequency fluctuation function:

$$\phi_r^{(1)} = \exp\left\{ i\int_0^{t_1} d\tau\omega_{10}(\tau) - i\int_{t_1+t_2}^{t_1+t_2+t_3} d\tau\omega_{10}(\tau)\right\} \tag{7}$$

$$\phi_{nr}^{(1)} = \exp\left\{ -i\int_0^{t_1} d\tau\omega_{10}(\tau) - i\int_{t_1+t_2}^{t_1+t_2+t_3} d\tau\omega_{10}(\tau)\right\} \tag{8}$$

$$\phi_r^{(2)} = \exp\left\{ i\int_0^{t_1} d\tau\omega_{10}(\tau) - i\int_{t_1+t_2}^{t_1+t_2+t_3} d\tau\omega_{21}(\tau)\right\} \tag{9}$$

$$\phi_{nr}^{(1)} = \exp\left\{ -i\int_0^{t_1} d\tau\omega_{10}(\tau) - i\int_{t_1+t_2}^{t_1+t_2+t_3} d\tau\omega_{21}(\tau)\right\} \tag{10}$$

They all have same interaction time but with different interaction dipole and semi-classical frequency fluctuation function ϕ. If only one transition permitted, R_3 and R_6 will disappear. And we often use Condon approximation so that μ is time-independent.

In cumulant approximation, we can write R_r and R_{nr} as:

$$\begin{aligned} R_r(t_3,t_2,t_1) =& 2|\mu_{10}|^4 e^{i\langle\omega_{10}\rangle(t_1-t_3)-i\omega_3 t_3+G_1(t_3,t_2,t_1)} \\ & - |\mu_{10}|^2|\mu_{21}|^2 e^{i\langle\omega_{10}\rangle t_1 - i\langle\omega_{21}\rangle t_3 - i\omega_3 t_3 + G_2(t_3,t_2,t_1)} \end{aligned} \tag{11}$$

$$\begin{aligned} R_{nr}(t_3,t_2,t_1) =& 2|\mu_{10}|^4 e^{-i\langle\omega_{10}\rangle(t_1+t_3)-i\omega_3 t_3+G_3(t_3,t_2,t_1)} \\ & - |\mu_{10}|^2|\mu_{21}|^2 e^{-i\langle\omega_{10}\rangle t_1 - \langle\omega_{21}\rangle t_3 + G_4(t_3,t_2,t_1)} \end{aligned} \tag{12}$$

$G_i(t_3,t_2,t_1)$ means different cumulant expansion function. We define the correlation function as:

$$c_{ij}(\tau) = \langle \delta\omega_i(\tau)\delta\omega_j(0)\rangle \tag{13}$$

and its time average function:

$$g_{ij}(t) = \int_0^t d\tau_2 \int_0^{\tau_2} d\tau_1 c_{ij}(\tau_1) \tag{14}$$

Where i,j means 10 transition and 21 transition. Then:

$$\begin{aligned} G_1 &= -g_{11}(t_1) + g_{11}(t_2) - g_{11}(t_3) - g_{11}(t_1+t_2) - g_{11}(t_2+t_3) + g_{11}(t_1+t_2+t_3) \\ G_2 &= -g_{11}(t_1) + g_{12}(t_2) - g_{22}(t_3) - g_{12}(t_1+t_2) - g_{12}(t_2+t_3) + g_{12}(t_1+t_2+t_3) \\ G_3 &= -g_{11}(t_1) - g_{11}(t_2) - g_{11}(t_3) + g_{11}(t_1+t_2) + g_{11}(t_2+t_3) - g_{11}(t_1+t_2+t_3) \\ G_4 &= -g_{11}(t_1) - g_{12}(t_2) - g_{22}(t_3) + g_{12}(t_1+t_2) + g_{12}(t_2+t_3) - g_{12}(t_1+t_2+t_3) \end{aligned} \tag{15}$$

Equation (15) has some interesting property. You can find that G_i has same time grid but different sign and time average function g.

For some reason, we will not use cumulant approximation for this program. The cumulant version has the same structure, but it's more difficult to calculate it due to multi G.

Population relaxation calculation is still a tough problem, so I didn't contain the population relaxation effect. There seems no certain phenomenal-adding method, but we may use lifetime-broadening factors in [2]:

$$
\begin{aligned}
\Gamma_{\mathrm{TA}}(t_3,t_2,t_1) &= \exp\left\{-\frac{\gamma_1+\gamma_2}{2}t_3-\gamma_1 t_2-\frac{\gamma_1}{2}t_1\right\}\\
\Gamma_{\mathrm{SE}}(t_3,t_2,t_1) &= \exp\left\{-\frac{\gamma_1}{2}t_3-\gamma_1 t_2-\frac{\gamma_1}{2}t_1\right\}\\
\Gamma_{\mathrm{GB}}(t_3,t_2,t_1) &= \exp\left\{-\frac{\gamma_1}{2}t_3-\gamma_1 t_2-\frac{\gamma_1}{2}t_1\right\}
\end{aligned}
\tag{16}
$$

then, we can include population relaxation by multiply λ with R, while TA for R_3, R_6 and SE/GB for others.

2 Computational detail

2.1 main structure

The main modules of my program are:

1. input class: parameter and transition.
2. response function calculation
3. Fast Fourier Transformation(FFT)

First of all, you need to offer time-dependent frequencies and some parameters. I made a input class for different transition, including basic information: time-dependent frequencies, transition dipole, and relaxation time T_1. All elements are public, so you can change them as simple as normal variable.

After reading part, we also need a unit transformation for energy(frequency) and time-step. Default time unit is ps and energy is wavenumber. In program, only dt and relaxation time is real time unit while other time variable will be time-normalization unit: $t = n * \Delta t \to t = n$. In that unit system, we only need to use integer number to describe time.

Then, we can calculate our response function i.e. R_r and R_{nr}. Different approximation may be used. Before that, we shall choose a ensemble average method, normally time average metho If we choose Fourier transformation integral area as rectangle, maximum time step of each time average will be $(N - (t_{\max} * 2 - t_2))$.

2.2 non-Condon

In this case, we have to use the original formula from 2 to 10. We can write them in discrete format. For example 2:

$$
\begin{aligned}
R_1(t_3,t_1;t_2) = \sum_{k=0}^{np} & \mu_{10}(k*t_{gap})\mu_{10}(k*t_{gap}+t_1)\mu_{10}(k*t_{gap}+t_1+t_2)\mu_{10}(k*t_{gap}+t_1+t_2+t_3) \\
& * \exp\left(i[\sum_{\tau=k*t_{gap}}^{t_1+k*t_{gap}} \omega(\tau) - \sum_{\tau=k*t_{gap}+t_1+t_2}^{t_1+t_2+t_3+k*t_{gap}} \omega(\tau)] \right)
\end{aligned}
\tag{17}
$$

As usual, I will save the response function as two-dimension matrix. You can find that exponential part is most time-cost, but we can calculate it skillfully. If we denote the first term in exp as $F(t_1, k)$ and second's as $G(t_1, t_3, k)$, we can find:

1. $F(t_1, k)$ is independent of t_3. We just need to update it when t_1 change.
2. $G(t_1, t_3, k)$ becomes 0 when $t_3 = 0$, and can be calculate iteratively when t_1 fixed.
3. $\phi_r^{(i)}$,$\phi_{nr}^{(i)}$ only have difference in sign of F,G.
4. $\phi_r^{(i)}$,$\phi_r^{(j)}$ or $\phi_{nr}^{(i)}$,$\phi_{nr}^{(j)}$ only have difference in G.

Inspired by information above, we can fixed t_1 and cycle t_3 first. Then calculate F,G in every total cycle. Be mention that F,G are just dependent in k and update with correlated time. And response function becomes:

$$
\begin{aligned}
R_r(t_3,t_1;t_2) &= \sum_{k=0}^{np}\{2*R_1(t_3,t_1,k;t_2) + R_3(t_3,t_1,k;t_2)\} \\
&= \sum_{k=0}^{np}\{2*\mu_{10}\exp\{iF(t_1,k) - iG_{10}(t_3,t_1,k)\} \\
&\qquad -\mu_{21}\exp\{iF(t_1,k) - iG_{21}(t_3,t_1,k)\}\}
\end{aligned}
\tag{18}
$$

$$
\begin{aligned}
R_{nr}(t_3,t_1;t_2) &= \sum_{k=0}^{np}\{2*R_4(t_3,t_1,k;t_2) + R_6(t_3,t_1,k;t_2)\} \\
&= \sum_{k=0}^{np}\{2*\mu_{10}\exp\{-iF(t_1,k) - iG_{10}(t_3,t_1,k)\} \\
&\qquad -\mu_{21}\exp\{-iF(t_1,k) - iG_{21}(t_3,t_1,k)\}\}
\end{aligned}
\tag{19}
$$

2.3 Ensemble

For multiple trajectory, we can use normal ensemble method that simply add their response function together. But most former case could only calculate one MD. So we have to use time-average ensemble. I once had a wrong understanding of time average. At that time, I regard configuration far away each other in MD as time-independent start point of trajectory.

When I read some MD textbook, I found that's wrong. Time-average is just an ensemble method based on ergodicity assumption. All statistic property has their time-average value:

$$\langle \hat{A} \rangle \xrightarrow{\text{In ergodicity assumption}} \frac{1}{t_n - t_0} \int_{t_0}^{t_n} A(t)dt \tag{20}$$

In numerical calculation, we can freely change integral boundary and interval to save our time:

$$\frac{1}{t_n - t_0} \int_{t_0}^{t_n} A(t)dt = \frac{1}{n} \sum_{i=0}^{n} A(i * \Delta t) \tag{21}$$

If we use relatively small n to reduce integral calculation, the response function may contain more noise. In [3] , for water 2d IR calculation, they only use 40 fs as time average interval! We have use much small integral interval before!

2.4 FFTW

It's really time-costing to use Direct Fourier Integral (DFI) to do Fourier transformation. Fast Fourier transformation could be done by C++ FFTW package. It should be noted this two method is not identical in result. The FFT is a fast transformation algorithm for Discrete Fourier transformation (DFT). 1d time-domain to frequency-domain DFT can be written as:

$$f(\omega_n) = \frac{\Delta t}{\sqrt{2\pi}} \sum_{k=0}^{N-1} f(t_k) \exp\left(\frac{2\pi i k n}{N}\right) \tag{22}$$

where $f(t_k)$ and $f(\omega_n)$ is time-domain function and frequency-domain function respectively, Δt is time interval of discrete data and N is total number of discrete data. Output grid ω_n obeys:

$$\omega_n = \frac{2\pi n}{N\Delta t} \tag{23}$$

There's some important property of ω_n. First, the maximum frequency is depended on Δt:

$$\omega_N = \frac{2\pi}{2\Delta t} \tag{24}$$

which is called the Nyquist critical frequency. second frequency resolution is depended on total time:

$$\Delta\omega_n = \frac{2\pi}{N\Delta t} \tag{25}$$

This means, if we want to increase our frequency resolution, we need enlarge our time region This transformation could remain some important property of Fourier transformation

There are many useful method to improve FFT efficiency and accuracy, such as zero-padding and trapezoidal rule (see section 9 in[3])

3 question

References

[1] Paesani, F.; Xantheas, S. S.; Voth, G. A. *The Journal of Physical Chemistry B* **2009**, *113*, 13118–13130.

[2] Kwac, K.; Lee, H.; Cho, M. *The Journal of Chemical Physics* **2004**, *120*, 1477–1490.

[3] Hamm, P.; Zanni, M. *Concepts and Methods of 2D Infrared Spectroscopy*; Cambridge University Press, 2011.