Tecniche di programmazione in chimica computazionale

Examples

Emanuele Coccia

Dipartimento di Scienze Chimiche e Farmaceutiche

Dipole moment and center of charge in molecules

For any molecule/cluster the dipole moment $\vec{\mu}$:

$$\vec{\mu} = \sum_{\alpha} q_{\alpha} \vec{r}$$

with α running on the atoms, q_{α} are the charges on atoms (Mulliken, Lowdin etc.) and $\vec{r}$ are the atomic coordinates

Dipole moment and center of charge in molecules

For any molecule/cluster the dipole moment $\vec{\mu}$:

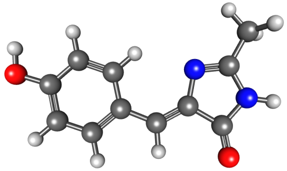
$$\vec{\mu} = \sum_{\alpha} q_\alpha \vec{r}$$

with α running on the atoms, q_α are the charges on atoms (Mulliken, Lowdin etc.) and $\vec{r}$ are the atomic coordinates
For charged systems, the center of charge $\vec{cc}$ is:

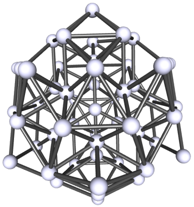
$$\vec{cc} = \frac{\sum_\alpha q_\alpha \vec{r}}{\sum_\alpha q_\alpha}$$

Dipole moment and center of charge in molecules

- Copy files from /home/tpcc/2022/CC to your work space
- HBDI (chromophore of GFP)



- Ag_{55}^{5+}



Exercises

- Compute center of charge and dipole of the two systems (cc_dip.f90)
- Compute center of mass of the two systems (com.f90)

Matrix diagonalization

- Square matrix **A** (*NxN*)

Matrix diagonalization

- Square matrix **A** (*NxN*)
- Matrix diagonalization: $\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{U}^{-1}$
- **D** =diag(a_1, a_2...a_N), a_i eigenvalues of **A**
- **U**: composed of eigenvectors of **A**

Matrix diagonalization

- Square matrix **A** (*NxN*)
- Matrix diagonalization: $\mathbf{A} = \mathbf{U}\ \mathbf{D}\ \mathbf{U}^{-1}$
- **D** =diag(a_1, a_2...a_N), a_i eigenvalues of **A**
- **U**: composed of eigenvectors of **A**
- Link to math libraries *mkl* (only ifort!) for compilation

Matrix diagonalization

- Square matrix **A** (*NxN*)
- Matrix diagonalization: $\mathbf{A} = \mathbf{U}\ \mathbf{D}\ \mathbf{U}^{-1}$
- **D** =diag(a_1, a_2...a_N), a_i eigenvalues of **A**
- **U**: composed of eigenvectors of **A**
- Link to math libraries *mkl* (only ifort!) for compilation
- Example diag.f90

Matrix transpose

- Square matrix $\mathbf{A}$
- Transpose of $\mathbf{A} \rightarrow \mathbf{A}^T_{ij} = \mathbf{A}_{ji}$

Matrix transpose

- Square matrix $\mathbf{A}$
- Transpose of $\mathbf{A} \rightarrow \mathbf{A}^T_{ij} = \mathbf{A}_{ji}$
- Example transpose.f90

Matrix transpose

- Square matrix $\mathbf{A}$
- Transpose of $\mathbf{A} \rightarrow \mathbf{A}^T_{ij} = \mathbf{A}_{ji}$
- Example transpose.f90
- Transpose conjugated of a matrix: example tconjug.f90

Franck-Condon factors

Born-Oppenheimer approximation

$$\begin{aligned}
\Psi_{e\nu}(q_e, q_N) &= \psi_e(q_e; q_N)\chi^e_\nu(q_N) \\
\mu_{e,\nu;e',\nu'} &= \int dq_N \chi^{e,*}_\nu(q_N) M_{ee'} \chi^{e'}_{\nu'}(q_N) \\
M_{ee'} &= \int dq_e \psi^*_e(q_e; q_N)\hat{\mu}\psi_{e'}(q_e; q_N)
\end{aligned}$$

Franck-Condon factors

Born-Oppenheimer approximation

$$\begin{aligned}
\Psi_{e\nu}(q_e, q_N) &= \psi_e(q_e; q_N)\chi_\nu^e(q_N) \\
\mu_{e,\nu;e',\nu'} &= \int dq_N \chi_\nu^{e,*}(q_N) M_{ee'} \chi_{\nu'}^{e'}(q_N) \\
M_{ee'} &= \int dq_e \psi_e^*(q_e; q_N)\hat{\mu}\psi_{e'}(q_e; q_N)
\end{aligned}$$

Electronic contribution to transition dipole moment not varying with q_N

$$\mu_{e,\nu;e',\nu'} = M_{ee'}(\bar{q}_N)\int dq_N \chi_\nu^{e,*}(q_N)\chi_{\nu'}^{e'}(q_N)$$

Franck-Condon factors

Born-Oppenheimer approximation

$$\begin{aligned}
\Psi_{e\nu}(q_e, q_N) &= \psi_e(q_e; q_N)\chi_\nu^e(q_N) \\
\mu_{e,\nu;e',\nu'} &= \int dq_N \chi_\nu^{e,*}(q_N) M_{ee'} \chi_{\nu'}^{e'}(q_N) \\
M_{ee'} &= \int dq_e \psi_e^*(q_e; q_N)\hat{\mu}\psi_{e'}(q_e; q_N)
\end{aligned}$$

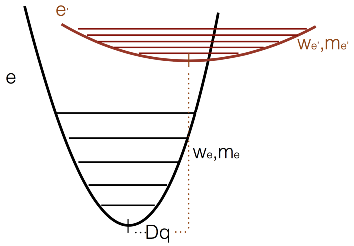
Electronic contribution to transition dipole moment not varying with q_N

$$\mu_{e,\nu;e',\nu'} = M_{ee'}(\bar{q}_N) \int dq_N \chi_\nu^{e,*}(q_N)\chi_{\nu'}^{e'}(q_N)$$

$$\begin{aligned}
S_{\nu,\nu'}^{e,e'} &= \int dq_N \chi_\nu^{e,*}(q_N)\chi_{\nu'}^{e'}(q_N) \\
\mathrm{FC}_{\nu,\nu'}^{e,e'} &= |S_{\nu,\nu'}^{e,e'}|^2
\end{aligned}$$

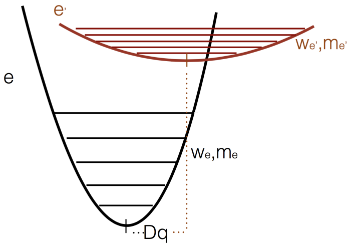
Franck-Condon factors

- Harmonic oscillator: $\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega\hat{x}^2$ ($\omega = \sqrt{\frac{k}{m}}$)



Franck-Condon factors

- Harmonic oscillator: $\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega\hat{x}^2 \; (\omega = \sqrt{\frac{k}{m}})$

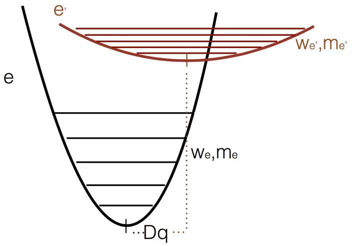


- Harmonic eigenfunctions

$$\begin{aligned}
\chi_0^e(q_N) &= \left(\frac{m_e\omega_e}{\pi}\right)^{1/4} \exp[-(m_e\omega_e)q_N^2/2] \\
\chi_1^e(q_N) &= \sqrt{2}\left(\frac{m_e\omega_e}{\pi}\right)^{1/4} (\sqrt{m_e\omega_e}q_N) \exp[-(m_e\omega_e)q_N^2/2] \\
\chi_2^e(q_N) &= \frac{1}{\sqrt{2}}\left(\frac{m_e\omega_e}{\pi}\right)^{1/4} \left[2m_e\omega_e q_N^2 - 1\right] \exp[-(m_e\omega_e)q_N^2/2]
\end{aligned}$$

Franck-Condon factors

- Harmonic oscillator: $\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega\hat{x}^2$ $(\omega = \sqrt{\frac{k}{m}})$



- Harmonic eigenfunctions

$$\chi_0^e(q_N) = \left(\frac{m_e\omega_e}{\pi}\right)^{1/4} \exp[-(m_e\omega_e)q_N^2/2]$$

$$\chi_1^e(q_N) = \sqrt{2}\left(\frac{m_e\omega_e}{\pi}\right)^{1/4} (\sqrt{m_e\omega_e}q_N) \exp[-(m_e\omega_e)q_N^2/2]$$

$$\chi_2^e(q_N) = \frac{1}{\sqrt{2}}\left(\frac{m_e\omega_e}{\pi}\right)^{1/4} \left[2m_e\omega_e q_N^2 - 1\right] \exp[-(m_e\omega_e)q_N^2/2]$$

- Example fc.f90

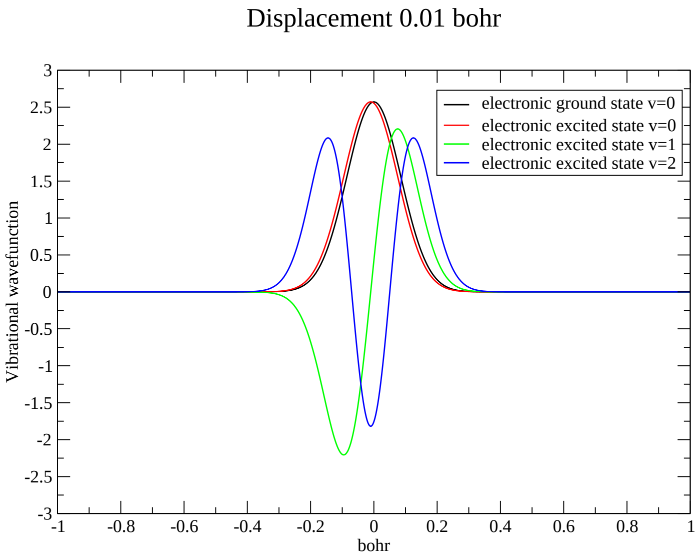
Franck-Condon factors

- Given the same frequencies, displacement, reduced masses, how the FC factor changes with the vibrational quantum number ν (0, 1 or 2) of the electronic excited state?

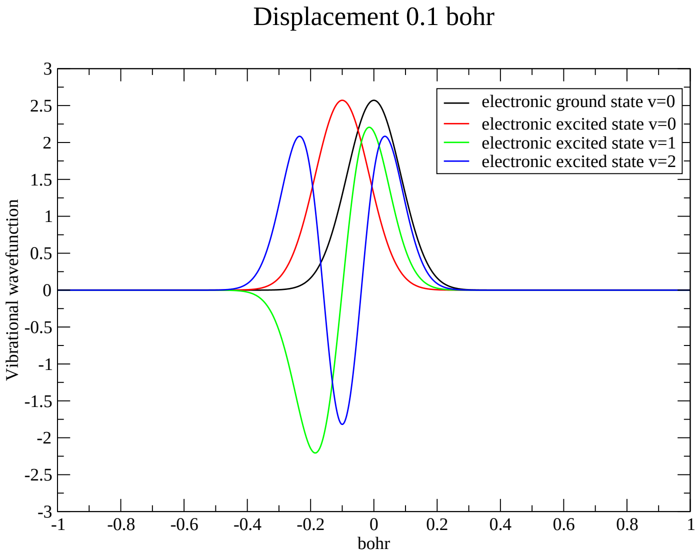
Franck-Condon factors

- Given the same frequencies, displacement, reduced masses, how the FC factor changes with the vibrational quantum number ν (0, 1 or 2) of the electronic excited state?
- Given the same frequencies and reduced masses, how the FC factor changes with the displacement Δq (for a chosen ν)?

“Pseudo” Br_2 results



"Pseudo" Br_2 results



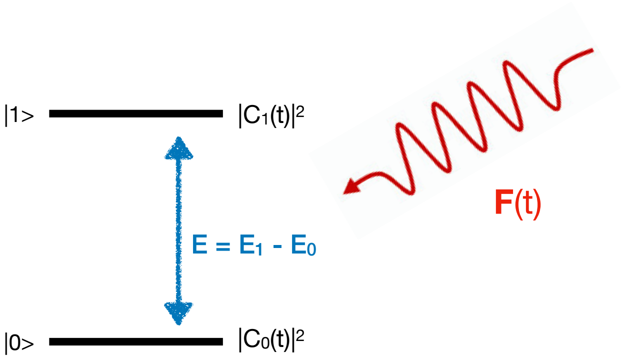
Simulating real-time dynamics

- Light-matter interaction, simulating time-resolved spectroscopies

Simulating real-time dynamics

- Light-matter interaction, simulating time-resolved spectroscopies
- Two-level system

$$|\Psi(t)\rangle = C_0(t)|0\rangle + C_1(t)|1\rangle$$



Simulating real-time dynamics

Time-dependent Schrödinger equation (TDSE)

$$
\begin{aligned}
i\frac{\partial|\Psi(t)\rangle}{\partial t} &= \hat{H}(t)|\Psi(t)\rangle \\
\hat{H}(t) &= \hat{H}_0 - \hat{\mu}\cdot\mathbf{F}(t) \\
i\frac{\partial C_0(t)}{\partial t} &= C_0(t)E_0 - \mathbf{F}(t)\left(\langle 0|\hat{\mu}|0\rangle + \langle 0|\hat{\mu}|1\rangle\right) \\
i\frac{\partial C_1(t)}{\partial t} &= C_1(t)E_1 - \mathbf{F}(t)\left(\langle 1|\hat{\mu}|1\rangle + \langle 1|\hat{\mu}|0\rangle\right) \\
\mathbf{F}(t) &= \mathbf{F}_{max}\exp\left(-\frac{(t-t_{mid})^2}{2\sigma^2}\right)\sin(\omega t),
\end{aligned}
$$

Simulating real-time dynamics

- Executable wavet.x in /home/tpcc/2022/WaveT

Simulating real-time dynamics

- Executable wavet.x in /home/tpcc/2022/WaveT
- input file, ci_ini.inp ci_energy.inp and ci_mut.inp in /home/tpcc/2022/WaveT
  - input: parameters for propagating TDSE
  - ci_ini.inp: initial populations
  - ci_energy.inp contains excitation energy *E*
  - ci_mut.inp contains dipoles and transition dipoles

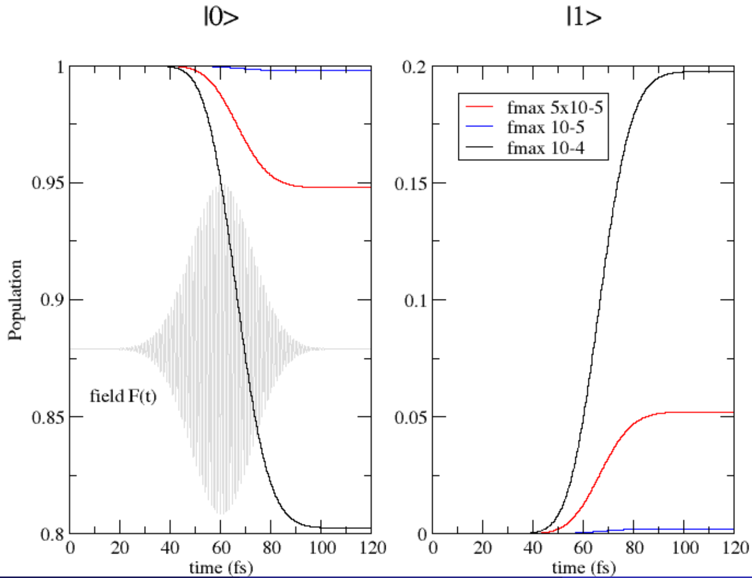
Simulating real-time dynamics

- Executable wavet.x in /home/tpcc/2022/WaveT
- input file, ci_ini.inp ci_energy.inp and ci_mut.inp in /home/tpcc/2022/WaveT
    - input: parameters for propagating TDSE
    - ci_ini.inp: initial populations
    - ci_energy.inp contains excitation energy E
    - ci_mut.inp contains dipoles and transition dipoles
- To run the simulation, copy the files in your working directory, and type ./wavet.x < input > output
- Many files are produced, focus on c_t_1.dat
    - c_t_1.dat: time evolution of $|C_0(t)|^2$ and $|C_1(t)|^2$
      step time Population $|0\rangle$, $|1\rangle$

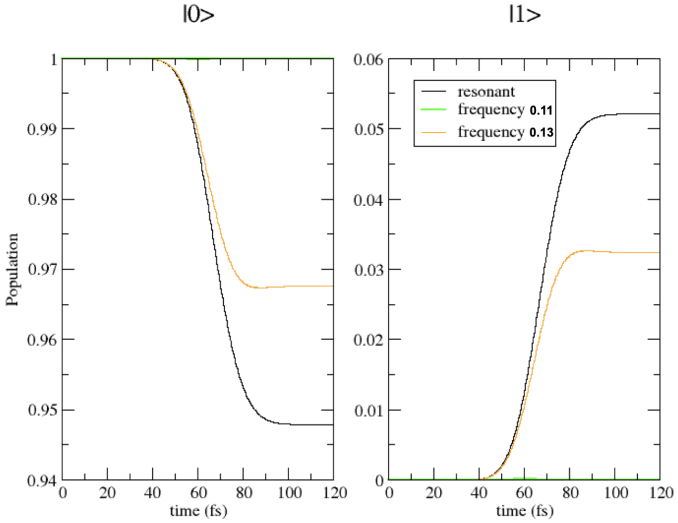
Simulating real-time dynamics

1. Check the last value of $|0\rangle$ and $|1\rangle$ populations by changing the amplitude of the pulse (fmax = 10^{-4}, 5x10^{-5}, 10^{-5}, resonant frequency)
2. Check the last value of $|0\rangle$ and $|1\rangle$ populations by changing the frequency of the pulse (fmax= 5x10^{-5}, omega=0.11, 0.13 and resonant)

Simulating real-time dynamics



Simulating real-time dynamics


|0>
|1>
Population
time (fs)
resonant
frequency 0.11
frequency 0.13