ezFCF v1.2

Franck-Condon factors in the harmonic approximation

Paweł Wójcik, Samer Gozem, Vadim Mozhayskiy, and Anna I. Krylov

October 14, 2022

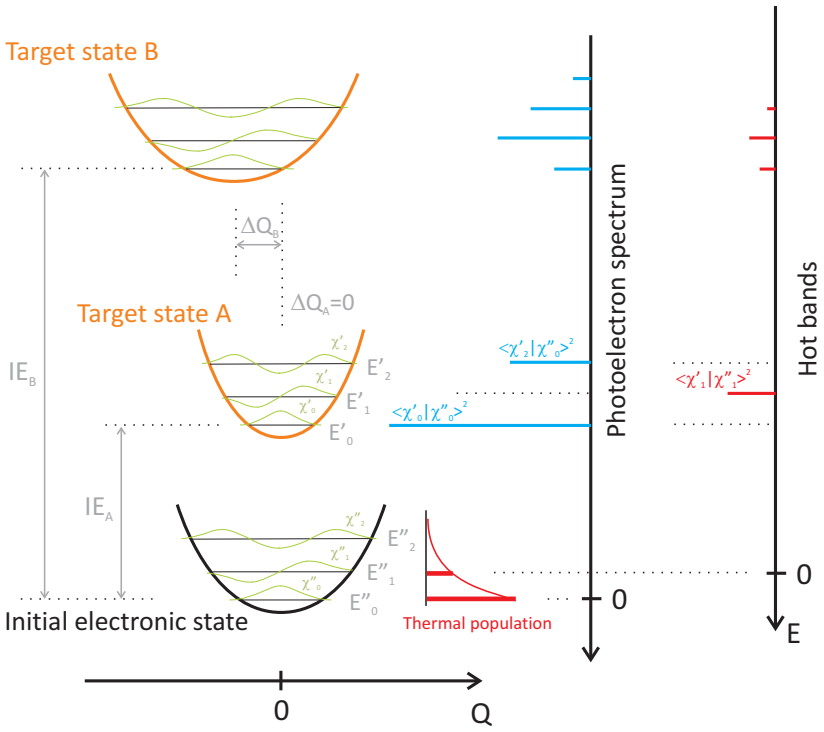
The program is available for download and at
`iopenshell.usc.edu/downloads/`

Contents

Introduction

`ezFCF` (formerly known as `ezSpectrum`) computes stick photoelectron/photodetachment spectra for polyatomics within the double-harmonic approximation.

Figure 1: 1D photoelectron spectrum in the harmonic approximation. χ'' and χ' are the initial and the target vibrational wavefunctions, respectively; E_i are the energies of the vibrational levels; IE is the adiabatic ionization energy; ΔQ is the displacement of the target state's equilibrium geometry along the normal coordinate; $\langle\chi'|\chi''\rangle$ are the Franck-Condon factors for $\chi' \leftarrow \chi''$ vibronic transitions.



Franck-Condon factors (FCFs), the overlaps between the initial and target vibrational wavefunctions, can be calculated:

- in *the parallel normal modes approximation* as products of one-dimensional harmonic wave functions (see Sec. 2);

- including *Duschinsky rotations* of the normal modes as full-dimensional integrals (see Sec. 3).

In both cases the overlap integrals are computed analytically. We note that in the literature it is often the squares of the overlaps that are called FCFs.

`ezFCF` calculations require equilibrium geometries, harmonic frequencies, and normal mode vectors for each electronic state. These data computed by *ab initio* packages. Shortcuts are possible by assuming that normal modes are the same in the initial and final states and that FCFs arise due to geometry changes—in this case, one needs to compute frequencies and normal modes for the initial state only and for the rest of the states only provide optimized geometries. Furthermore, one can even avoid doing optimizations for each target state by using vertical gradient approximation, which requires only calculations of nuclear gradient at the equilibrium geometry of the initial state.

The key feature of `ezFCF` is a program-independent `xml` input. It can be prepared either manually following the format description and examples, or by running the python script that processes `Q-Chem`, `ACESII`, `CFour`, `Molpro`, `GAMESS`, `Orca`, and `Gaussian` [1] outputs. The script can be easily modified to process outputs of other programs.

Please contact Professor Anna I. Krylov (`krylov@usc.edu`) if you have suggestions, questions, or bugs to report. Have fun!

What's new

ezFCF v.1.2 versus ezFCF v.1.1

This version includes the following new features:

- Vertical gradient approximation (see Section 2.4);
- Ability to compute FCFs for transitions originating from a single vibrational level (see Sec. 2.3.1 and 3.5);
- Fixes to the scripts mostly extending the range of accepted *ab initio* outputs.

ezFCF v.1.1 versus ezSpectrum v.3.0

`ezFCF` v.1 is an updated version of `ezSpectrum` v.3. It is a part of `ezSpectra` spectroscopy modeling suite[1]. The functionality and most of the compute

[1] We are Gaussian free zone, and encourage everyone to degauss their workplace as well.

kernels in `ezFCF` v.1 are the same as in `ezSpectrum` v.3, but the input parser was rewritten to remove the dependency on the outside library. In addition, we extended the support to include `Orca` (many thanks to Denis Tikhonov who provided the script). The `xml` parser works almost as before, but with the following (temporary) restrictions:

- Empty XML tags are not supported, i.e., this

```
<geometry
 n_of_atoms="3"
 text = "
      O      -0.0002310000    -0.1047090000     0.0000000000
      H      -0.7616800000     0.4751910000     0.0000000000
      H       0.7589890000     0.4781700000     0.0000000000

  "
/>
```

  should be changed to

```
<geometry
 n_of_atoms="3"
 text = "
      O      -0.0002310000    -0.1047090000     0.0000000000
      H      -0.7616800000     0.4751910000     0.0000000000
      H       0.7589890000     0.4781700000     0.0000000000

">
</geometry>
```

- The closing tags, such as

```
</geometry>
```

  above, should be separated by either a space or a newline from the previous text. For example

```
"></geometry>
```

  would result in an error.

- We changed

  `<ip>`

  to

  `<excitation_energy>`

  .

- We changed how the stick spectra is printed (use positive energy now).
- We added new plotting *Python* scripts that generate a convoluted spectrum from the output of `ezFCF`.

The scripts were rewritten to use *python3*. The current scripts take the above changes into account, but the old `xml` inputs need to be edited.

1 Quick start

This section explains how to create an input `xml` file from your electronic structure program outputs using `make_xml.py` script and how to run `ezFCF`. For brief overview of the `xml` syntax please see the Appendix; the detailed description of the input file format is given in the subsequent sections of the manual.

The distribution `ezFCF_v1.2.tar.gz` includes the following:

```
atomic masses:  xml file with atomic masses (needed for program to run)
LICENCE: lincence info (please read)
Manual: user manual (tex and pdf)
bin:  precompiled binaries for MacOS and Linux and python script
make_xml.py (python3 is required) for xml input generation
InputScripts: xml inputs from supported ab initio packages
      and examples of inputs and outputs
Plotting: python scripts for plotting spectra (both python2 and python3)
Samples: xml inputs (new format) and outputs
References: xml inputs (old format) and outputs (ezFCF v.1.0) for sample jobs
ezSpectrum_code: source code and makefiles
```

After you install it, you can add the path to the bin directory in your `.csh` or `.bsh` configuration file. Alternatively, you can create links to the executables in your working directory.

1.1 Create an input file: `make_xml.py` script

`make_xml.py` python script (located in `bin` directory) creates an `xml` input file for `ezFCF` from several *ab initio* outputs. All *ab initio* outputs should be for the same molecule (different electronic states and total charges are OK) and contain a frequency job at the optimized geometry. A resulting `xml` input file contains basic default job parameters and an `electronic_state` element for every output file provided.

The script `make_xml.py` executed with no arguments prints the usage format:

```
make_xml.py <filename.xml> <initial_state.out>
    <target_state_1.out> <target_state_2.out> etc...
```

`filename.xml` is the name of `xml` input file to be created, and other arguments are the file names of *ab initio* outputs. The first output is for the initial electronic state, and all sequential files are for the target electronic states.

`InputScripts` folder contains sample input and output files for *ab initio* packages supported by the script. Each file contains the in an H_2O frequency job using HF/6-311G**. Here is the summary of *ab initio* outputs' formats:

	Cartesian geometry units	Mass weighted normal modes?
`Q-Chem`	Å	Yes
`ACES II/CFour`	Bohr	No
`GAMESS` and `Molpro`	Bohr	Yes
`Gaussian`	Å	Yes
`Orca`	Å	Yes

For these packages `make_xml.py` script sets all necessary flags automatically, but if you plan to use other *ab initio* programs you can compare the format of your outputs with the ones provided.

Some tags in the created input `xml` file are "commented" by the `OPT_` prefix: when removed, the tags become visible to `ezFCF`; for more details on comments see Sec. 5.3.

To plot the spectra, use `plot_spectrum.py` script located in `Plotting` directory.

1.2 Run a sample job

`ezFCF` executable takes the input `xml` file as a single argument. There are several sample input `xml` files in the `Samples/` folder:

- `trans_hcoh.xml` —a basic input for both parallel approximation and Duschinsky rotations calculations; no hot bands (0 K temperature);
- `cis_hcoh.xml` —includes hot bands (600 K temperature) and energy thresholds for vibrational states;
- `formaldehyde.xml`—a basic input for both parallel approximation and Duschinsky rotations calculations; normal modes are reordered manually for the best match;
- `adenine.xml`—atoms and normal modes are reordered in the target state; single FCFs are requested in the Duschinsky rotations section;
- `thymine.xml`—only Duschinsky rotations with full set of optional keywords; it takes several minutes to complete this job;
- `vg_phenolate.xml`—an input for calculation using vertical gradient approximation;
- `the_only_initial_state.xml`—an input for calculation using only one vibrational state in the initial electronic state, see 2.3.1 and 3.5. The example uses the previously described formaldehyde model.

Note that the input file must contain `parallel_approximation` or `dushinsky_rotations` section. One may include both sections in a single input and compare the calculated spectra.

To run `ezFCF` for one of the examples type:

```
./ezFCF_mac.exe samples/cis_hcoh.xml > samples/cis_hcoh.xml.out
```

on Mac OS or

```
./ezFCF_linux.exe samples/cis_hcoh.xml > samples/cis_hcoh.xml.out
```

on a Linux machine. The working directory should also contain the `atomicMasses.xml` file (see Section 1.5). Note that the executables must be executable—you may need to run

```
chmod +x ezFCF*.exe
```

after download. In addition, on Mac OS you may need to go into the System Preferences to allow the program to run.

Calculated spectra in the parallel approximation and with the Duschinsky rotations will be saved in the `cis_hcoh.xml .spectrum_parallel` and `cis_hcoh.xml .spectrum_dushinsky` files, respectively as well as in the `cis_hcoh.xml.out` output file. The format of the spectrum and the output files is briefly described in Sec. 1.4

1.3 Job parameters

There are two general job parameters: the temperature in Kelvin and a minimum intensity threshold for spectral lines.

Here is an example of the input file structure:

```
<input
   job="harmonic_pes">

  <job_parameters
     temperature                    = "300"
     spectrum_intensity_threshold   = "0.001"
    />

  <parallel_approximation     ....   />

  <dushinsky_rotations  .... />

  <initial_state>   .... </initial_state>

  <target_state>    .... </target_state>

</input>
```

Job-specific parameters for the parallel approximation and Duschinsky rotations are described in Sec. 2 and Sec. 3, respectively. The initial and target state elements are created by the `make_xml.py` script and can be easily adjusted by the user (see Sec. 4).

Both temperature and intensity threshold are applied after FCFs are calculated (using either approximation). The intensity I of the vibronic transition is proportional to the square of the FCF ($\langle\chi'|\chi''\rangle^2$, where χ'' and χ' are the vibrational wavefunctions of the initial and target electronic states, respectively). For the hot bands (i.e., transitions from the vibrationally excited initial electronic state), the intensity is additionally multiplied by the Boltzmann population of the initial vibrational state at temperature T. The overall intensity I^{calc} for a given $\chi' \leftarrow \chi''$ transition is calculated as:

$$I \sim I^{calc} = \langle\chi'|\chi''\rangle^2 \cdot e^{-E''/kT}, \tag{1}$$

where E'' is the energy of the initial vibrational state relative to the ground vibrational state. If the calculated intensity is above the specified threshold, the transition is included in the spectrum. The Boltzmann distribution in the above equation is not normalized and the population of excited vibrational states of the initial electronic state is calculated relative to the population of the ground vibrational state (which is always one).

1.4 Output

A stick photoelectron spectrum is printed both at the end of each section of the output (parallel approximation and Duschinsky rotations sections) and in separate files: `*.spectrum_parallel` and/or `*.spectrum_dushinsky` .

Spectra are printed as five columns (from left to right):

- energy positions of the spectral lines: $E = E^a_{00} + (E' - E'')$, where E'' and E' are the energies of the vibrational states relative to the ground vibrational states of the initial and target electronic states respectively (see Fig. 1); E^a_{00} is a ZPE-adjusted adiabatic ionization or excitation energy (the energy difference between the ground vibrational states of the initial and the target electronic state);
- spectral intensities calculated by Eq. (1);
- Franck-Condon Factors $\langle \chi' | \chi'' \rangle$;
- E''—energies of initial vibrational states relative to the ground vibrational state of the initial electronic state;
- assignment of the vibronic transitions $\psi'' \to \psi'$; both vibronic states are printed as: *"electronic state number"*(*"list of vibrational excitations"*), where each vibrational excitation is in the following format: [*number of quanta*]v[*normal mode's number*]; zero in parentheses is the ground vibrational state; normal mode's numbering starts with zero;

For example:

```
Energy   Intensity    FC factor     E",K   Transition

1.000  1.1102e-02  +1.053e-01     0.000  0(0)->1(0)
1.030  5.2203e-04  -1.365e-01  2145.805  0(1v3)->1(1v2)
1.099  1.6786e-04  +1.522e-01  2956.964  0(2v0)->1(2v0,1v1)
1.122  2.3433e-02  +1.530e-01     0.000  0(0)->1(1v1)
```

Every output file of `ezFCF` starts with a copy of the input `xml` file followed by the molecular geometries of each electronic state after the automatic alignment procedure. It is recommended to check that all states were aligned correctly—a large value of the "The norm of the geometry difference from the initial state" in the output may indicate a mis-alignment of the geometries (see Sec. 5.2). The parallel approximation job's output also includes: (a) the normal modes' overlap matrices between the initial and each of the target electronic states (each matrix should be diagonal for the

parallel approximation; otherwise, the normal modes must be reordered to minimize the off diagonal elements—see Sec. 5.4); and (b) the table of the target states displacements, ΔQ, relative to the initial state geometry in the "initial coordinates" and in "target coordinates" (i.e., along the normal modes of the initial and the target electronic states, respectively). In the Duschinsky rotation section of the output, one should check the normal modes rotation matrix determinant, $Det(S)$, which should be close to one (see Sec. 5.5); also check that you have enough memory to store all the requested "layers".

A basic `python3` script for plotting the resulting spectra is provided in `Plotting`. To use it to plot a spectrum from `Samples/`, follow these steps:

```
cd Samples
ln -s ../Plotting/plot_spectrum_new.py .
./plot_spectrum_new.py trans_hcoh.xml.spectrum_parallel
```

The plot will be in `trans_hcoh.xml.spectrum_parallel.pdf`. Tweak the parameters in the script for the desired appearance. (Note that `python` executable path should be properly setup).

1.5 Atomic masses file

File `atomicMasses.xml` contains atomic masses in atomic mass units (amu) and must be present in the same folder as an `xml` input file (you can use `ln -s` command to create a link to the master copy from the directory where you intend to run `ezFCF`). If you need more atoms, e.g. isotopes, you can add new elements into the root element `masses` of this file. The user-defined atomic names must start with a letter and contain only letters and numbers. Here is an example of a short `atomicMasses.xml` file:

```
<masses>
  <H>    1.007820 </H>
  <C>   12.000000 </C>
  <C14> 14       </C14>
</masses>
```

2 Parallel approximation

2.1 Overview

Within the dipole approximation, the intensity of one-electron transition between two vibronic states is proportional to the square of the electric dipole

transition moment, which in adiabatic approximation can be expressed as:

$$\langle \chi'(Q) \cdot \psi'(q,Q) | M(q,Q) | \psi''(q,Q) \cdot \chi''(Q) \rangle = \mu \cdot \langle \chi'(Q) | \chi''(Q) \rangle, \tag{2}$$

where ψ'' and ψ' are the electronic wavefunctions, χ'' and χ'—the vibrational wavefunctions of the initial and target states, respectively; q and Q are the electronic and nuclear coordinates. If the dependence of an electronic transition moment μ on the nuclear coordinates is neglected, the intensity of the transition is proportional to the square of the Franck-Condon factor (FCF)[2, 3, 1] defined here as the overlap between the two vibrational wavefunctions:

$$\langle \chi'(Q) | \chi''(Q) \rangle. \tag{3}$$

Within the harmonic approximation vibrational wavefunctions are products of one dimensional harmonic oscillator wavefunctions. If normal coordinates of the initial and the target electronic states are assumed to be the same (the parallel approximation), multidimensional Franck-Condon factors reduce to products of one dimensional FCFs:

$$\begin{aligned} \langle \chi'(Q) | \chi''(Q) \rangle = \\ \langle \chi'_1(Q_1) \cdot \chi'_2(Q_2) \cdot ... | \chi''_1(Q_1) \cdot \chi''_2(Q_2) \cdot ... \rangle = \\ \langle \chi'_1(Q_1) | \chi''_1(Q_1) \rangle \cdot \langle \chi'_2(Q_2) | \chi''_2(Q_2) \rangle \cdot ... \end{aligned} \tag{4}$$

`ezFCF` computes each one-dimensional harmonic FCF analytically[4]:

$$\begin{aligned} \langle \chi'_{\nu'} | \chi''_{\nu''} \rangle = & \sqrt{\frac{2\alpha}{\alpha^2+1}} \cdot \sqrt{\frac{\nu''\nu'}{2^{(\nu''+\nu')}}} \cdot e^{\frac{-\delta^2}{2(\alpha^2+1)}} \sum_{L=0}^{L<min\{\nu'',\nu'\}} \sum_{i=0}^{i\leq\frac{\nu'-L}{2}-1} \sum_{j=0}^{j\leq\frac{\nu''-L}{2}-1} \\ & \left[\frac{1}{L!} \left(\frac{4\alpha}{1+\alpha^2} \right)^L \frac{1}{i!} \left(\frac{1-\alpha^2}{1+\alpha^2} \right)^i \frac{1}{j!} \left(\frac{\alpha^2-1}{1+\alpha^2} \right)^j \right. \\ & \left. \frac{1}{\nu'-2i-L} \left(\frac{-2\alpha\delta}{1+\alpha^2} \right)^{\nu'-2i-L} \frac{1}{\nu''-2j-L} \left(\frac{2\delta}{1+\alpha^2} \right)^{\nu''-2j-L} \right], \end{aligned} \tag{5}$$

where

$$\begin{aligned} \alpha &= \sqrt{\frac{\omega''}{\omega'}} \\ \delta &= \Delta Q \sqrt{\omega''}, \end{aligned} \tag{6}$$

ν'' and ν' are quantum numbers; ω'' and ω' are harmonic frequencies of the initial and the target electronic state, respectively; ΔQ is the displacement of the target electronic state relative to the initial one along a mass-weighted normal mode. ΔQ can be computed from the optimized geometries of the initial and final state or estimated from the gradient of the target state computed at the equilibrium geometry of the initial state (vertical gradient approximation).

2.2 Keywords

Here is an example of the `parallel_approximation` element with a full set of keywords (all sub elements of the `parallel_approximation` element are optional):

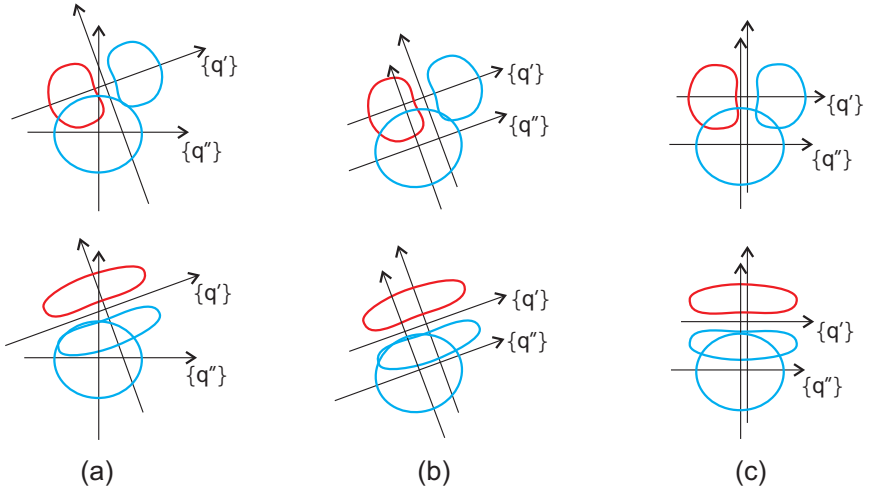
```
<parallel_approximation
   max_vibr_excitations_in_initial_el_state = "1"
   max_vibr_excitations_in_target_el_state  = "4"
   combination_bands                        = "true"
   use_normal_coordinates_of_target_states  = "true"
  >

   <print_franck_condon_matrices flag="true"/>

   <do_not_excite_subspace size="3" normal_modes="0 3 7" />

   <energy_thresholds  units="eV, K, cm-1">
      <initial_state   units="K">  1000 </initial_state>
      <target_state   units="eV">  0.25 </target_state>
   </energy_thresholds>

   <the_only_initial_state text = "1v0,1v2" >
   </the_only_initial_state>

</parallel_approximation>
```

Four attributes of the `parallel_approximation` element are always required:

- maximum number of the vibrational excitations in the initial and target electronic states (the total number of quanta in all normal modes);
- flag to include the combination bands (vibrational states with several normal modes excited at the same time);
- the flag whether to use the normal modes of the target (`true`) or the initial (`false`) state.

The latter requires clarification. In general, if there are no hot bands in the spectrum (0 K, and only the ground vibrational state of the initial electronic state is populated), using the normal modes of the target electronic state is preferred. Indeed, rotation of the initial ground states' vibrational wavefunction does not significantly change the overlap integrals with the target state (see Fig 2); on the other hand, rotation of target state's vibrational wavefunctions with many nodes strongly affects the overlap matrices.

Figure 2: The effect of normal coordinate's rotation on the FCFs within the parallel mode approximation: (a) the correct overlap between wavefunctions on lower (q") and upper (q') surfaces; (b) the overlap when lower normal coordinates are rotated to coincide with upper coordinates; (c) the overlap when upper normal coordinates are rotated to coincide with lower coordinates. More detailed discussion can be found in Ref. [5].



2.3 Reduced dimensionality calculations and energy thresholds

Commonly, some normal modes can be excluded from the model because they are not excited in the experiment, or, not active due to the symmetry restrictions. Sometimes it is desirable to exclude normal modes that have too large displacements or that are too anharmonic, in order to compute FCFs for all other normal modes and treat these problematic modes separately.

The time cost of combination band calculation grows exponentially with the number of the normal modes, but can be significantly reduced by ex-

cluding some of the normal modes from the "active subspace" by adding:

```
<do_not_excite_subspace size="2" normal_modes="3 38" />
```

into the `parallel_approximation` element. In this example, no vibrational quanta will be added to the normal modes number `3` and `38`. (Please remember that normal modes are numbered from `0`.)

The number of the combination bands may also be reduced by applying thresholds to the energies of the vibrational states (relative to the ground vibrational states):

```
<energy_thresholds  units="eV, K, cm-1">
   <initial_state   units="K">     1000    </initial_state>
   <target_state   units="eV">      0.25   </target_state>
</energy_thresholds>
```

The energies of the thresholds can be in electron volts, Kelvins and wavenumbers (the value of the `units` attribute must be `eV`, `K` or `cm-1`, respectively). In the parallel approximation lists of the initial and the target vibrational states below specified thresholds are formed first. Then the FCFs for all possible initial-target combinations are evaluated. This algorithm reduces the calculation time, but requires more memory to store the initial-target pairs of states (see Sec. 5.6).

Note that, if the energy threshold is lower than some of the vibrational frequencies, you may include these "high frequency" normal modes into the `do not excite subspace` for an additional speedup.

If normal modes of the target state are reordered to maximally diagonalize the normal modes' overlap matrix (see Sec. 4.3), the new order is used whenever the normal modes are referenced in the input. If normal modes of the target electronic state are used (`use normal coordinates of target states = "true"`), then the new order of normal modes will be used in the output as well. In particular the new order of the normal modes is used in: the transition assignments of the final spectrum printout and in the `do not excite subspace` element of the input.

2.3.1 Additional state for customized calculations

The reduced dimensionality calculations may exclude a state that has an important contribution to the spectrum. The keyword `the_only_initial_state` allows to generate the missing contribution. To use this feature add an `xml` node to the `parallel_approximation` node

```
<the_only_initial_state text = "1v0,1v2" >
</the_only_initial_state>
```

The argument of `text` contains a string representing one initial vibrational state. The state representation uses the regular convention: a normal mode `nm` with `q` excitations should be listed as `qvnm`; if more than one mode is excited use comma as a separator. If the parallel approximation is not enough, this functionality is available also in the calculations that include the Duschinsky effect (see Sec. 3.5).

2.4 Vertical gradient approximation

One can compute a full spectrum without target state geometry optimizations by instead calculating the excited states' nuclear gradient. The respective relaxed geometries can be extrapolated using both the gradients of the target state computed at the equilibrium geometry of the initial state and the initial state's Hessian. This approach called the vertical gradient approximation (VGA) is exact when both (i) the potential energy surfaces are harmonic and (ii) normal modes are the same in the initial and target states. VGA is also known as the linear coupling model.

VGA is triggered when the `gradient` node is detected inside the `target_state` node. In that case, the `geometry`, `normal_modes` and `frequencies` nodes are ignored by the parser and do not need to be specified. The corresponding quantities are copied from the `initial_state` node just as expected in the parallel mode approximation without frequency shifts, after which the gradient is used to find the VGA equilibrium geometry (as described below). The gradient must be given in atomic units `a.u.` (Hartree/Bohr).

If the VGA is used the `excitation_energy` node is not allowed (it is also not available as the target state optimization is omitted), instead one has to use the analog `vertical_excitation_energy` node.

Within VGA, the Cartesian geometry of the target state is given by

$$\vec{x_0'} = \vec{x_0''} - T^{-\frac{1}{2}} L \Omega^{-2} L^{-1} T^{-\frac{1}{2}} \vec{g}. \tag{7}$$

where Ω is an $N \times N$ diagonal matrix of harmonic frequencies, $\vec{g}$ is the gradient of the potential energy surface of the target state in Cartesian coordinates evaluated at the geometry of the ground state. The other symbols are explained in Sec. 3.1. The adiabatic excitation energy (bottom to bottom) extrapolated by the vertical gradient method is given by

$$E_{\mathrm{ad}} = E_{\mathrm{vert}} - \frac{1}{2} \sum_{i=1}^{N} \omega_i^2 d_i^2, \tag{8}$$

where ω_i is the ith harmonic frequency and d_i is the ith element of the displacement vector from Eq. (10).

An example of the `vertical_excitation_energy` and `gradient` nodes

```
<vertical_excitation_energy
    units="eV">
    2.32
</vertical_excitation_energy>
<gradient
    units = "a.u."
    text = "
      -0.00000000
       0.00000000
      -0.13653498
       0.00148260
      -0.00000000
       0.00362732
      -0.00148260
       0.00000000
       0.00362732
       0.00000000
      -0.00000000
       0.12928034"
>
</gradient>
```

The `text` argument contains $3 * (\#\text{ atoms})$ numbers – the gradient components. The order of the components must stick to the coordinate system used throughout the `geometry` and `normal_modes` nodes, i.e., it must be a list of the x, y, and z gradient components corresponding to the first atom, followed by the x, y, and z components of the second atoms and so on; the order of atoms must be the same as the order of atoms appearing in the `geometry` node. The value of the `text` argument is read as a series of floating point number separated by white-spaces.

3 Duschinsky rotations

3.1 Overview

The harmonic approximation is briefly described in Sec. 2.1. When normal modes of the initial and the target electronic state are significantly non-parallel, the nuclear coordinates of the initial and the target vibrational wavefunctions are different, and the full FCFs $\langle\chi'(Q')|\chi''(Q'')\rangle$ are not represented by the product of the one-dimensional integrals, Eq. (4). Multidimensional Franck-Condon factors between two harmonic vibrational wavefunctions can be evaluated analytically [6, 7]. For a molecule with K atoms and N normal modes ($N = 3K - 6$ or $N = 3K - 5$ for non-linear

and linear molecules, respectively), normal modes of the initial and target states are related by the Duschinsky transformation[8]:

$$\overrightarrow{Q'} = S \cdot \overrightarrow{Q''} + \overrightarrow{d}, \tag{9}$$

where the normal modes rotation matrix S $[N \times N]$ is:

$$S = L'^T L''$$

and the vector of displacements $\overrightarrow{d}$ $[N]$ along the normal coordinates is:

$$\overrightarrow{d} = L'^T \sqrt{T}(\overrightarrow{x''_0} - \overrightarrow{x'_0}) \tag{10}$$

where L'' $[N \times 3K]$ and L' $[N \times 3K]$ are rectangular matrices composed of N mass-weighted normal vectors (in Cartesian coordinates) of the initial and the target electronic states respectively; $\overrightarrow{x''_0}$ $[3K]$ and $\overrightarrow{x'_0}$ $[3K]$ are the Cartesian geometries vectors of the initial and the target states respectively; and matrix T $[3K \times 3K]$ is the diagonal matrix composed of atomic masses: $T = diag\{m_1, m_1, m_1, m_2, m_2, m_2, ..., m_K, m_K, m_K\}$.

The overlap integral (FCF) between the ground vibrational states of the initial and target electronic states is given by[7]:

$$\langle \chi'_0 | \chi''_0 \rangle = \frac{2^{N/2}}{\sqrt{det(S)}} \left[\prod_{\eta=1}^{N} \left(\frac{\omega'_\eta}{\omega''_\eta} \right) \right]^{1/4} \sqrt{det(Q)} \left[e^{-\frac{1}{2} \overrightarrow{\delta}^T (1-P) \overrightarrow{\delta}} \right] \tag{11}$$

(Note that there is a typo in this equation in Ref. [7]) FCFs for transition from the ground vibrational state of the initial electronic state are calculated recursively[7] from the $\langle \chi'_0 | \chi''_0 \rangle$ integral as:

$$\langle \chi'_{\nu'_1,...\nu'_\xi+1,...\nu'_N} | \chi''_0 \rangle = \sqrt{\frac{2}{\nu'_\xi + 1}} \left[(1-P) \overrightarrow{\delta} \right]_\xi \langle \chi'_{\nu'_1,...\nu'_\xi,...\nu'_N} | \chi''_0 \rangle + \sum_{\theta=1}^{N} \sqrt{\frac{\nu'_\theta}{\nu'_\xi + 1}} [2P-1]_{\xi\theta} \langle \chi'_{\nu'_1,...\nu'_\theta-1,...\nu'_N} | \chi''_0 \rangle \tag{12}$$

Hot bands (transitions from the vibrationally excited initial electronic state) are given by:

$$\langle \chi'_{\nu'_1,\ldots\nu'_N} | \chi''_{\nu''_1,\ldots\nu''_\eta+1,\ldots\nu''_N} \rangle =$$
$$-\sqrt{\frac{2}{\nu''_\eta+1}} \left[R\overrightarrow{\delta}\right]_\eta \langle \chi'_{\nu'_1,\ldots\nu'_N} | \chi''_{\nu''_1,\ldots\nu''_\eta,\ldots\nu''_N} \rangle +$$
$$\sum_{\theta=1}^{N} \sqrt{\frac{\nu''_\theta}{\nu''_\eta+1}} \left[2Q-1\right]_{\eta\theta} \langle \chi'_{\nu'_1,\ldots\nu'_N} | \chi''_{\nu''_1,\ldots\nu''_\theta-1,\ldots\nu''_N} \rangle +$$
$$\sum_{\xi=1}^{N} \sqrt{\frac{\nu'_\xi}{\nu''_\eta+1}} R_{\eta\xi} \langle \chi'_{\nu'_1,\ldots\nu'_\xi-1,\ldots\nu'_N} | \chi''_{\nu''_1,\ldots\nu''_\eta,\ldots\nu''_N} \rangle \qquad (13)$$

where ν_i is a number of vibrational quanta in the i-th normal mode. J, Q, P, and R are square $[N \times N]$ matrices:

$$J = \lambda' S \lambda''^{-1}$$
$$Q = (1 - J^T J)^{-1}$$
$$P = JQJ^T$$
$$R = QJ^T$$

$\overrightarrow{\delta}$ is a vector $[N]$:

$$\overrightarrow{\delta} = \lambda' \overrightarrow{d}$$

λ'' and λ' are $[N \times N]$ diagonal matrices:

$$\lambda'' = diag\{\sqrt{\omega''_1}, \sqrt{\omega''_2}, \ldots \sqrt{\omega''_N}\}$$
$$\lambda' = diag\{\sqrt{\omega'_1}, \sqrt{\omega'_2}, \ldots \sqrt{\omega'_N}\}$$

where ω''_i and ω'_i are the frequencies of the $i-$th normal mode in atomic units.

Total number of vibrational states with up to K quanta for the molecule with N normal modes is given by[9]:

$$\left[1 + \sum_{k=1}^{K} \binom{N+k-1}{N-1}\right]^2 = \left[1 + \sum_{k=1}^{K} \frac{(N+k-1)!}{(N-1)!k!}\right]^2 \qquad (14)$$

3.2 Keywords

Full dimensional harmonic FCFs (with Duschinsky rotations) are evaluated if the `dushinsky_rotations` element is present in the root of the input `xml` file:

```
<dushinsky_rotations target_state="1"
      max_vibr_excitations_in_initial_el_state = "3"
      max_vibr_excitations_in_target_el_state  = "8"
/>
```

All three arguments in the example above are required:

- target state from the input to include; if there is only one target state in the input, the attribute value must be `"1"`. Implementation of the Duschinsky rotation, for simplicity, allows only one target state. If you need to include several target states, please combine the results from several separate calculations.
- the maximum number of vibrational excitations in the initial and target electronic states (total number of excitations in all normal modes combined).

This minimal set of parameters is sufficient, however for larger systems it is often unrealistic to go beyond even few quanta of vibrational excitations. Additional keywords can be used to reduce the number of FCFs and the time of a calculation. Here is the example of the `dushinsky_rotations` element with all optional keywords included:

```
<dushinsky_rotations target_state="1"
      max_vibr_excitations_in_initial_el_state = "3"
      max_vibr_excitations_in_target_el_state  = "8"
      >

   <energy_thresholds  units="eV, K, cm-1">
     <initial_state   units="K">  1000  </initial_state>
     <target_state   units="eV">  0.25  </target_state>
   </energy_thresholds>

   <max_vibr_to_store  target_el_state="6"/>

   <do_not_excite_subspace size="2" normal_modes="3 38" />

   <single_excitation
        ini="0"
       targ="2v38"/>
   <single_excitation
```

```
        ini="2v3 3v4"
       targ="1v1 1v2 1v3 7v38"/>
   <single_excitation
        ini="1v0"
       targ="0"/>

</dushinsky_rotations>
```

The optional keywords are described below.

In general there is no need to reorder the normal modes (see Sec. 4.3) for the Duschinsky rotation calculations, and we advise you to use the default order (as in the *ab initio* input). If normal modes of the target state have to be reordered, please remember that the new order is to be used in all assignments in the output and in the input. In particular the new order of the normal modes is used in:

- transition assignments of the final spectrum printout (energies and intensities of the Duschinsky rotation spectrum are obviously unaffected by any normal modes reordering);
- `do_not_excite_subspace` element of the input (see Sec. 3.3).
- single `excitation_element` -s of the input (see Sec. 3.4);
- `the_only_initial_state` node (see Sec. 3.5).

3.3 Reduced dimensionality calculations

It is common that some normal modes can be excluded from the model: whether they are not excited in the experiment, or, for example, not active because of the symmetry restrictions.

The time cost of combination band calculations grows exponentially with the number of normal modes, and can be significantly reduced by excluding some of the normal modes from the "active subspace" by adding:

```
   <do_not_excite_subspace size="2" normal_modes="3 38" />
```

into the `dushinsky_rotations` element. In this example no vibrational quanta will be added to the normal modes number `3` and `38` (please note that normal modes are numbered from `0`).

3.4 Single FCFs for customized calculations

One can include FCFs for particular vibronic transitions by adding `single_excitation` elements to the `dushinsky_rotations` . For example:

```
<single_excitation
     ini="0"
    targ="2v2"/>
<single_excitation
     ini="2v0 3v2"
    targ="1v1/>
```

That may be useful, for example, if one want to include the "tail" of a long vibrational progression that goes beyond the maximum number of vibrational quanta value.

In the above example for a molecule with three normal modes, the following FCFs will be evaluated: $\langle 002|000\rangle$ and $\langle 010|203\rangle$. The vibrational state of the initial and target electronic states are described in the `ini` and `targ` attributes, respectively. The values of these attributes are either `"0"` for the ground vibrational state, or a list of excited normal modes (space and/or end-of-line separated). The first number in each word (before the letter `v`) is the number of quanta in a particular normal mode, and the second number is the normal mode number (normal modes are numbered from zero; also see a note in Sec. 3.2 regarding the normal modes reordering).

All requested single FCFs are evaluated and printed in the output. However only transitions with an intensity above the requested threshold are added to the spectrum.

You can also include excitations to normal modes from the `do_not excite_subspace` .

3.5 Additional state for customized calculations

If one is interested in computing all transitions from a selected initial state, manual addition of the `single_excitation` nodes can become too tedious. The keyword `the_only_initial_state` is available also for Duschinsky calculations (see Sec. 2.3.1 for the equivalent keyword in the parallel approximation).

The keyword `the_only_initial_state` allows one to generate transitions originating from a selected initial state. To use this feature add an `xml` node to the `dushinsky_rotations` node

```
<the_only_initial_state text = "1v0,1v2" >
</the_only_initial_state>
```

The argument of `text` contains a string representing one initial vibrational state. The state representation uses the regular convention: a normal mode

`nm` with `q` excitations should be listed as `qvnm`, if more than one mode is excited use comma as a separator.

3.6 Other keywords: CPU and memory use

As it was mentioned in Sec 3.1, `ezFCF` stores in memory only FCFs for the transitions from the ground vibrational state of the initial electronic state (i.e., no hot bands are stored). All hot bands are evaluated recursively, which may require considerable CPU time. To reduce the number of hot bands, one can apply thresholds to the energies of the vibrationally excited states (above the ground vibrational state) in both initial and target electronic states:

```
<energy_thresholds  units="eV, K, cm-1">
   <initial_state   units="K">     1000    </initial_state>
   <target_state   units="eV">      0.25   </target_state>
</energy_thresholds>
```

Note that in Duschinsky rotation calculations *only hot bands* calculations will be accelerated by the use of the energy thresholds. The energy of the thresholds can be in electron volts, Kelvins and wavenumbers (the value of the `units` attribute must be `eV`, `K` or `cm-1`, respectively).

If the energy threshold is lower than some of the vibrational frequencies, you may include these "high frequency" normal modes into the `do_not excite_subspace` for an additional speedup.

If the memory is insufficient to store all "layers" (sets of FCFs with the same number of vibrational quanta) up to the `max vibr excitations in target el state` value, you may limit the maximum number of "layers" to store by adding:

```
<max_vibr_to_store  target_el_state="6"/>
```

into the `dushinsky_rotations` element. In the example above only layers with up to six vibrational excitations will be stored.

Sizes of the layers for a molecule with 39 normal modes (15 atoms) are shown in the table below (printed in the output before the FCFs evaluation):

Layer	Size in bytes
K'=0	8 b
K'=1	312 b
K'=2	6.09 Kb
K'=3	83.28 Kb
K'=4	874.45 Kb
K'=5	7.34 Mb
K'=6	53.86 Mb
K'=7	346.22 Mb
K'=8	1.94 Gb
K'=9	10.15 Gb
K'=10	48.73 Gb
K'=11	217.08 Gb

The next table shows the time benchmarks for different number of layers stored in the memory for the same molecule [2].

Layers stored (max number of vibr. quanta=8)	Memory used, Gb	No hot bands included	11,505,823 hot bands (up to 3 vibr. quanta in the initial state)
0	0.00	14h 4min	25h 28min
6	0.06	40 min	2h 5min
8	2.40	29 min	1h 54 min

Note that recursive evaluation of two highest layers (second row) is done in almost the same time as when all layers are stored (third row), which is obvious from the recursion equations in Sec. 3.1. However full recursive calculations (first row) are much slower.

4 `initial_state` and `target_state` elements

For each *ab initio* output `make_xml.py` script adds an electronic state element to the input `xml` file; the `initial_state` element is created from the first provided output, and `target_state` elements are created from all consecutive outputs. In many cases there is no need to change anything except the `excitation_energy` element (see below). However the optional elements described in this section may be used for the manual reordering

[2] These times were obtained on Intel Xeon 3.2GHz processor

of normal modes and/or atoms, if necessary. Full description of the electronic state element format may be also useful if the data is taken from unsupported *ab initio* packages.

Each electronic state element must have the following elements: (a) `geometry` (Sec. 4.1); (b) `normal_modes` (Sec. 4.3); and (c) `frequencies` (Sec. 4.4). In addition, each `target_state` element must contain the `excitation_energy` element. The value of the adiabatic excitation or ionization energy should be the ZPE-corrected (0-0) adiabatic energy (i.e. the energy difference between lowest vibrational states of initial and target electronic states). `make_xml.py` script fills the `excitation_energy` elements with a default value of 1.0 eV for the first target state.

The optional elements of the electronic state element are:

(a) `manual_atoms_reordering` (Sec. 4.1);

(b) `manual_coordinates_transformation` (Sec. 4.2);

(c) `manual_normal_modes_reordering` (Sec. 4.3);

(d) `gradient` (allowed only in target states) (Sec. 2.4).

4.1 `geometry` element

The `geometry` element contains the molecular geometry of an electronic state and has the following format:

- attribute `text` equals to the molecular geometry in Cartesian coordinates, i.e. `"AtomName1 X1 Y1 Z1 AtomName2 X2 Y2 Z2 ..."` ; atomic names and coordinates can be separated by any number of spaces, tabs and/or end-of-line symbols;
- coordinates can be in Angstroms or atomic units (Bohrs), and `units` attribute must be set to `"angstr"` or `"au"` respectively (see Sec. 1.1 for the geometry format in different *ab initio* packages);
- masses for every atomic name are listed in the `atomicMasses.xml` file (see Sec. 1.5);
- `number_of_atoms` attribute;
- attribute `linear` is `"true"` or `"y"` for a linear molecule and `"false"` or `"n"` for a non-linear molecule.

Here is an example of a `geometry` element:

```
<geometry
   number_of_atoms="4"
   linear="false"
   units="au"
   text="
     C    1.39149214     0.22982244   0.00
     O   -1.06145138    -0.15710283   0.00
     H    2.14668108    -1.72045361   0.00
     H   -1.86894079     1.47733232   0.00 "
 />
```

If your *ab initio* outputs have different order of atoms for different electronic states, you can reorder atoms in both `geometry` and `normal_modes` by adding the `manual atoms reordering` element in the electronic state element:

```
<manual_atoms_reordering
   new_order="0 3 1 2" />
```

The numbering of atoms starts with zero. Please pay special attention to the order of the same-type atoms, e.g., hydrogens in this example.

4.2 Automatic states alignment

The molecular orientation of *ab initio* outputs is not always well defined. The geometries of initial and target electronic states must be correctly aligned before the FCF calculations. To address the problem of possible misalignment, `ezFCF` uses a simple automatic alignment algorithm, which aligns translational and rotational degrees of freedom to ensure that the subspaces of the normal modes is the same for all electronic states.

Each electronic state is realigned in three steps:

- atomic coordinates are shifted so that the center of mass coincides with the coordinates origin;
- molecular geometry and normal modes are rotated so that the principal axes of the moment of inertia tensor are aligned with the coordinate axes;
- molecular geometry and normal modes of each `target_state` are additionally rotated by $k_i \cdot \pi/2$ (k_i is an integer, $i = x, y, z$) around x, y, and z axes so that each target state has the same orientation as the initial state (the norm of the geometries difference is minimized).

This "automatic alignment algorithm" is pretty robust, but if it fails to align the states, each state can be reoriented manually by adding the `manual coordinates transformation` element to the electronic state element. The shift and rotation around axis i ($i = x, y, z$) is defined in `shift along` i attribute in Angstroms and `rotate around` i attribute in units of π, respectively. When the manual transformation is performed, first the geometry is shifted and then it is rotated around x, then y, and then z axis. Note that the coordinate system is "right-handed" and the rotation is CCW if you look along the direction of the axis of rotation. Here is the example of the `manual coordinates transformation` element:

```
<manual_coordinates_transformation
  shift_along_x="0.07" shift_along_y="0" shift_along_z="0"
  rotate_around_x="1" rotate_around_y="0.5" rotate_around_z="0"/>
```

The automatic reorientation is skipped for states with the manual reorientation. So, if an automatic alignment does not work for your system it is recommended to add the manual coordinate transformation to *each* electronic state in the `xml` input.

4.3 `normal_modes` element

The matrix of normal modes' vectors must be listed in the `"text"` attribute of the `normal_modes` element and formatted as it is shown in Fig. 3.

Figure 3: The format of the normal modes' vectors in the input file: (a) normal modes are grouped in sets of three; (b) every next set of three normal modes is below the previous one; (c) within each set, the first line is the list of coordinates for the first atom, the second—for the second atom, and so on.

Normal mode #1

	X	Y	Z
Atom #1	0.813	0.078	0.000
Atom #2	-0.406	-0.039	0.000
Atom #3	-0.406	-0.039	0.000

Normal mode #2

```
-0.078  0.813 0.000
 0.039 -0.406 0.000
 0.039 -0.406 0.000
```

Normal mode #3

```
0.000 0.000  0.816
0.000 0.000 -0.408
0.000 0.000 -0.408
```

Normal mode #4

```
0.000 0.000  0.000
0.000 0.000  0.707
0.000 0.000 -0.707
```

If normal modes' vectors are mass weighted (Å/$\sqrt{amu}$, e.g., as in a `Q-Chem` or `MOLPRO` outputs) the argument `if mass weighted` must be `"true"`;

otherwise (e.g., as in an `ACESII` outputs) it must be set to `"false"` (see Sec. 1.1 for the normal modes format in different *ab initio* packages).

Atomic names in the `atoms` attribute of the `normal_modes` element define atomic masses for which normal modes were calculated (used to "unmassweight" the normal modes). Note, that atomic names in the `geometry` element can be different, which may be used for the isotope substitutions: one can rerun `ezFCF` for a molecule with the isotope substitutions using the set of normal modes from an *ab initio* output for the non-substituted molecule.

Here is an example of the minimal `normal_modes` element for a water molecule:

```
<normal_modes
   if_mass_weighted="y"
   atoms = "O   H   H"
   text = "
     0.000 0.0 -0.072    0.000 0.0  0.049   -0.073 0.0  0.000
    -0.417 0.0  0.569   -0.592 0.0 -0.386    0.576 0.0  0.407
     0.417 0.0  0.569    0.592 0.0 -0.386    0.576 0.0 -0.407 "
 />
```

Script `make_xml.py` keeps the normal modes order the same as in an *ab initio* output, i.e., in the frequency ascending order. This does not guarantee that the particular normal mode (e.g. the symmetric stretch) has the same number in all electronic states, which is required for the *parallel mode approximation*. One can reorder normal modes (only in the `target_state` -s) with the optional `manual normal modes reordering` element:

```
<manual_normal_modes_reordering new_order="0 1 3 2 4 5"/>
```

The numbering of the normal modes starts with zero. The `formaldehyde.xml` sample job illustrates modes reordering (try to run it without and see what happens).

`ezFCF` prints the overlap matrix between normal modes of the initial and each target state, which helps to assign the normal modes of the target states to the normal modes of the initial one. By default only a non-diagonal part of the overlap matrix is printed, but one can print a full overlap matrix by adding an optional `<print_normal_modes flag="true"/>` element to the root `input` element of the input file (see Sec. 5.2 for some additional details).

Note that the reordering of the normal modes may affect the output assignment of the vibrational transitions and the input keywords, which refer to the normal modes. For more details, please see notes in the end of the of the parallel approximation (Sec. 2.2) and Duschinsky rotation (Sec. 3.2)

keywords sections.

Normal modes vectors are printed after the automatic or manual transformations, if an optional `<print normal modes flag="true"/>` element is in the root element of the input file.

4.4 `frequencies` element

Frequencies must be in the wavenumbers and listed in the `text` attribute of the `frequencies` element separated by any number of spaces, tabs, or/and end-of-line symbols. For example:

```
<frequencies
   text="   1101.42   1217.81   1339.50
            1520.23   2882.19   3772.87  "/>
```

The order of the frequencies should be the same as the order of the normal modes in the `normal_modes` element.

5 Common problems and FAQ

5.1 It does not run.

Make sure your executables have proper permissions—use `chmod` command if necessary.

5.2 My spectrum is empty!

There could be many causes for this. Try the following:

- Check the intensity threshold (make it 0) and the energy thresholds (remove). Switch off combination bands for a faster diagnostics of the parallel approximation.
- Check that dQ-s (displacements along normal modes) in the parallel approximation are small (generally less than 1.0 and usually in the order of 0.1). Large dQ-s may indicate too large changes in the geometries (you may try to go higher in the vibrational excitations by increasing the total number of vibrations limit for the target state). However, often this means that the normal modes are not parallel, geometries are not aligned, or the order of atoms is different in different states. See the items below for more details.
- Check that normal modes' overlap matrix is diagonal if you use the parallel approximation (see Sec. 5.4) and that det(S) is close to one (see Sec. 5.5) if you use the Duschinsky rotations.
- Check that geometries of all states are similarly aligned after the automatic alignment procedure: you can plot "New molecular geometries" for each state from the `ezFCF` output with your favorite visualization software. A large value of the "Norm of the geometry difference" is an indicator of miss-aligned geometries. The following can cause problems in the alignment: (a) geometries of electronic states are very different (e.g. cyclization of a linear molecule); (b) you have two chiral isomers which can not be aligned by rotation; (c) the order of atoms is not the same in all electronic states (if not—reorder, see Sec. 4.1, pay a special attention to the order of the like atoms, e.g, hydrogens). If the automatic alignment does not work and all of the above is checked, try to align your molecules manually (see Sec. 4.2).

5.3 How do I comment parts of the input?

The formal rule is to enclose comments in `<!--` and `-->` tags (see Appendix). However, any tag which is not recognized by `ezFCF` is simply ignored. So you can modify both an open and a respective close tags of the element to make it a comment. For example in the inputs created with `make_xml.py` script optional elements (keywords) are preceded with `OPT_` and are invisible for `ezFCF`.

You also can add comments to the elements by adding any attributes which are well formatted. For example, the attribute `units` of the `energy_threshold` element is a comment and ignored by `ezFCF`.

5.4 Normal modes are not parallel...

Or even worse—you get a "Normal modes' overlap matrix is non diagonal" message in your output.

Parallel approximation assumes that normal modes of the initial and target states are parallel. In many cases this works very well even for a slightly rotated normal modes. However the normal modes in the *ab initio* outputs are in the frequency ascending order, which may be different since the frequency of a particular normal mode changes from state to state.

The goal is to make the overlap matrix as diagonal as possible by re-ordering the normal modes of each target state using the `manual normal modes reordering` element (see Sec. 4.3). If the number of normal modes is large, you can try to use `do not excite subspace` to reorder a small subset of normal modes at a time (especially if the molecule is symmetric and normal modes can be naturally separated). If you want to see the full normal modes' overlap matrix when you use a subspace add

```
<print_normal_modes flag="true"/>
```

into the root element `input` This will also print normal modes after all coordinate transformations.

Also note that Duschinsky rotations automatically eliminate the normal modes' reordering problem, but for large molecules it may be problematic to include high vibrational excitations with Duschinsky rotations.

5.5 $Det(S)$ is less than one (or even zero)

S is the normal modes' rotation matrix in a Duschinsky transformation, Eq. (9). If all normal modes are in the "excite subspace", $Det(S)$ should be almost 1.0, which corresponds to a true rotation and therefore both sets

of normal modes span the same space. Use of the "do not excite subspace" keyword can make two "active" subspaces different and not alignable by a rotation (therefore $Det(S)$ is not 1.0). So, either your subspace is not uncoupled from the rest of the normal modes, or, more likely, you use different normal modes in your initial and target subspaces. The latter requires reordering of the normal modes of the target electronic state (see Secs. 4.3 and 5.4). Once again, if you use a full space of normal modes, there is no need to reorder normal modes for Duschinsky rotations section.

5.6 `ezFCF` crashes with no error message!

There is one known problem: `ezFCF` crashes when too many FCFs are requested. We will try to address this in the future, but basically in the case of a crash it would be unrealistic anyway to evaluate all requested combination bands in a finite time. So, the only solution is to reduce the number of FCFs:

- apply intensity and energy thresholds;
- limit maximum number of vibrational excitations;
- reduce the "excite normal modes' subspace" by excluding normal modes along which the geometry change is small.

Another possibility is that you just found a new bug in `ezFCF` :)—please let us know and forward us the problematic input. We will try to update the program asap!

5.7 FCFs are very large or infinite

We are aware of one problem that causes this, which is due to hidden line break characters introduced by Windows. This can easily be solved by using `dos2unix` (which is already available on Linux or can be downloaded online) to convert line breaks back to unix format. Try applying `dos2unix` to the input `xml` file and on `atomicMasses.xml`.

5.8 `ezFCF` ends with a format error

This problem can be caused by several issues related to the format of the `xml` input. However, one common problem occurs when Raman intensities are requested in `Q-Chem`. As a result, the `Q-Chem` output is not read correctly by the `make_xml.py` script and this causes the "format error" to show up

in `ezFCF`. Re-run the `Q-Chem` job without requesting Raman to avoid this problem or compose your `xml` input manually.

Bibliography

References

[1] S. Gozem and A. I. Krylov, WIREs: Comput. Mol. Sci. **12**, e1546 (2022).

[2] J. Franck, Trans. Faraday Society **21**, 536 (1926).

[3] E. Condon, Phys. Rev. **28**, 1182 (1926).

[4] E. Hutchisson, Phys. Rev. **36**, 410 (1930).

[5] L. Koziol, V. Mozhayskiy, B. J. Braams, J. M. Bowman, and A. I. Krylov, J. Phys. Chem. A **113**, 7802 (2009).

[6] H. Kupka and P.H. Cribb, J. Chem. Phys. **85**, 1303 (1986).

[7] R. Berger, C. Fischer, and M. Klessinger, J. Phys. Chem. A **102**, 7157 (1998).

[8] F. Duschinsky, Acta Physicochim. USSR **7**, 551 (1937).

[9] Please refer to the "stars and bars" combinatorial problem elsewhere, e.g. http://en.wikipedia.org/wiki/Stars_and_ bars_(probability).

Appendix. An overview of the `xml` format

Extensible Markup Language (XML) is widely used for storage of a structured data. Any data, which is representable as a tree structure, can be saved in an .xml file. Tables and databases are the simplest examples of the possible applications.

Every node in a tree (including all sub-nodes) is called an element. Every element in the `xml` file should be placed within "tags" as in HTML (a special case of XML). XML tags can have any names, which are defined by the particular application.

Here is the set of rules for creating a "well formatted" `xml` file:

- Document must have only one root element. In the case of `ezFCF`, the input file has the `<input> ... </input>` root element, and the `atomicMasses.xml` file has the `<masses> ... </masses>` root element.

- Comments can appear anywhere in the document and must be enclosed by `<!--` and `-->`. Also any valid element or argument (see below) can be considered as a comment if it is not used by the program (see Section 5.3).

- Every element must be enclosed by the *start* and *end* tags: `<sample_tag>` and `</sample_tag>`.

- If an element does not have any sub-elements the *start* and *end* tags can be combined in a single tag: `<sample_tag />`, which is equivalent to `<sample_tag> </sample_tag>`.

- Every element may contain any number of elements.

- *Start* tags can have any number of attributes in the following format: `<sample_tag attribute1_name = "attr1_value" attribute2_name = "attr2_vailue">`. Values of attributes must be quoted by single or double quotes.

- The names of elements (tags) and attributes are case sensitive.

- Elements can not overlap. Here is an example of an incorrect format: `<tag1> <tag2> </tag1> </tag2>`. The error message will be that no end `</tag2>` tag was found in the `<tag1>...</tag1>` element.