■ Concepts and Methods of 2D Infrared Spectroscopy

Peter Hamm and Martin T. Zanni

This progam was used to produce Figs. 10.3, 10.4, and 10.5, i.e. the 2D IR lineshape of the OH stretch vibration in water, using the Cumulant expansion

Set working directory (you will have to change it)

```
In[1]:= SetDirectory["C:\Dokumente und Einstellungen\p.hamm\
            Eigene Dateien/PCI/projekte/2010/2D-IR Book/SecSimul/SimulWater"];
```

Constants and proportionaltiy factor that translates forces into a frequency shift (Eq. 10.4)

```
In[2]:= NA = 6.023 * 10^23;
        w0 = 3550;
        mass = 1.66 * 10^-27;
        mH = 1.;
        mO = 16.;
        mu = mH * mO / (mH + mO);
        c = 3 * 10^10;
        h = 6.626 * 10^-34;
        Δ = 240;
        fact = Sqrt[9 / 4 (Δ / w0) 10^18 * 10^6 / NA^2 / (w0 * c * h) / (mu * mass) * 10^-24] / .188;
```

Read MD output files generated by Gromacs (see README file)

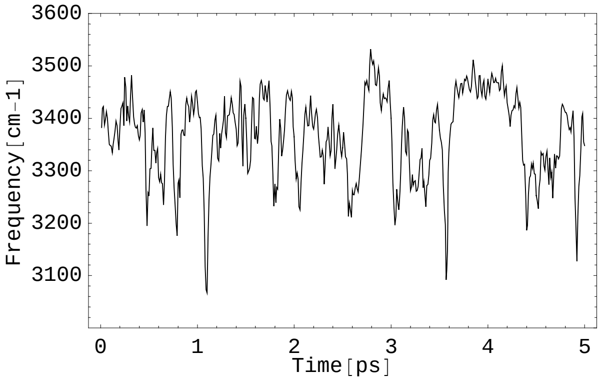
```
In[12]:= coord = ReadList["coord.xvg", {Real, Real, Real, Real, Real, Real, Real}];
         force = ReadList["force.xvg", {Real, Real, Real, Real, Real, Real, Real}];
```

Calculate frequency shift according to Eqs. 10.2 and 10.4

```
In[14]:= n = Length[force];
         dt = (force[[2, 1]] - force[[1, 1]]);
         bondOH = Table[{coord[[i, 5]] - coord[[i, 2]],
             coord[[i, 6]] - coord[[i, 3]], coord[[i, 7]] - coord[[i, 4]]} , {i, 1, n}];
         length = Sqrt[bondOH[[1, 1]]^2 + bondOH[[1, 2]]^2 + bondOH[[1, 3]]^2];
         forceOH = Table[{(force[[i, 5]] / mH - force[[i, 2]] / mO) * mu,
             (force[[i, 6]] / mH - force[[i, 3]] / mO) * mu,
             (force[[i, 7]] / mH - force[[i, 4]] / mO) * mu} , {i, 1, n}];
         w = w0 - Table[(forceOH[[i, 1]] * bondOH[[i, 1]] + forceOH[[i, 2]] * bondOH[[i, 2]] +
                forceOH[[i, 3]] * bondOH[[i, 3]]), {i, 1, n}] / length * fact;
```

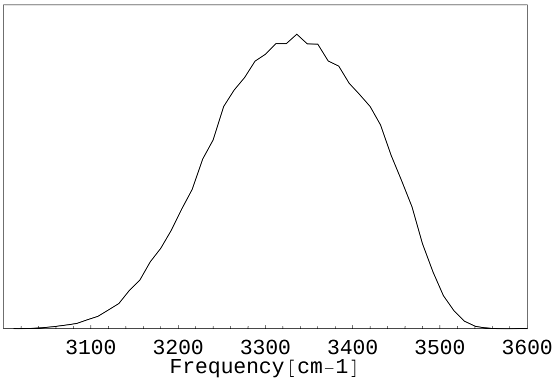
Plot a piece of the frequency trajectory (Fig. 10.3a)

```
In[20]:= wmin = 3000;
         wmax = 3600;
         Table[{i dt, w[[i]]}, {i, 1, 500}];
         ListPlot[%, PlotJoined → True, PlotRange → {wmin, wmax},
           Frame → True, FrameLabel → {"Time[ps]", "Frequency[cm-1]"}];
```



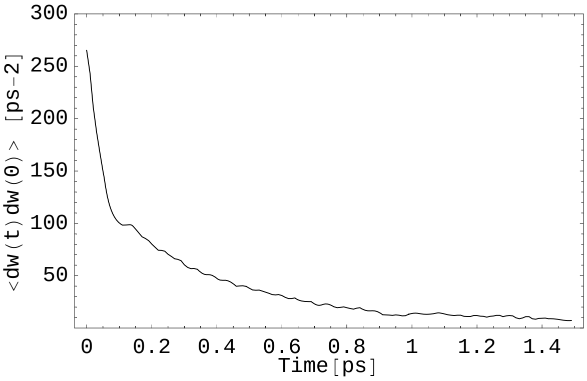
Calculate frequency distribution (Fig. 10.3b)

```
In[24]:= nd = 50;
         dist = Table[0., {i, 1, nd}];
         For[i = 1, i ≤ n, i++, dist[[Floor[(w[[i]] - wmin) / (wmax - wmin) * nd]]] += 1];
         dist = Table[{wmin + i (wmax - wmin) / nd, dist[[i]]}, {i, 1, nd}];
         ListPlot[dist, PlotRange → {{wmin, wmax}, {0, Max[dist] * 1.1}},
           PlotJoined → True, Frame → True, FrameLabel → {"Frequency[cm-1]", None},
           FrameTicks → {True, False, False, False}];
```



Calculate Frequency Fluctuation Correlation Function (FFCF, Fig. 10.3.c)

```
In[29]:= ncorr = 150;
         wmean = Sum[w[[i]], {i, 1, n}] / n;
         corr =
           Table[{i * dt, Sum[(w[[iti]] - wmean) * (w[[iti + i]] - wmean), {iti, 1, n - ncorr, 5}] /
               (n - ncorr) * 5 * .188^2}, {i, 0, ncorr - 1}];
         ListPlot[corr, PlotJoined → True, PlotRange → {0, 300}, Frame → True,
          FrameLabel → {"Time[ps]", "<dw(t)dw(0)> [ps-2]"}]
```

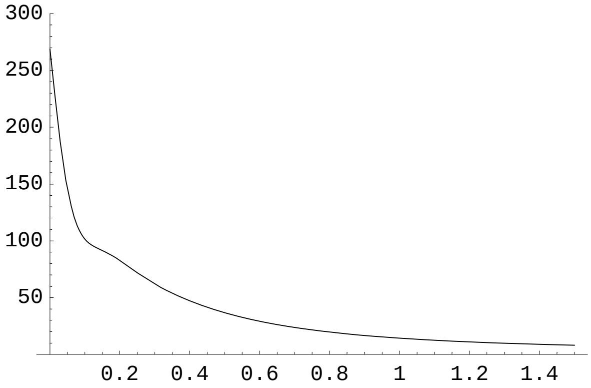


```
Out[32]= - Graphics -
```

Fit the FFCF to an analytic function in order to calculate a line shape function

```
In[33]:= << Statistics`NonlinearFit`
         fit = NonlinearFit[corr, a1 Cos[w00 t] Exp[-t / τ1] + a2 Exp[-t / τ2] + a3 Exp[-t / τ3],
           t, {{a1, 100}, {w00, 3}, {τ1, .05}, {a2, 100}, {τ2, .1}, {a3, 50}, {τ3, 1}}]
         g[tau2_] = Simplify[
           Chop[Integrate[Integrate[fit, {t, 0, tau1} ], {tau1, 0, tau2}] ]];
         Plot[fit, {t, 0, 1.5}, PlotRange → {0, 300}]
```

Out[34]= $129.313\, e^{-3.76619\, t} + 25.5684\, e^{-0.811311\, t} + 113.851\, e^{-20.6157\, t} \operatorname{Cos}[24.1127\, t]$



```
Out[36]= - Graphics -
```

Define response functions for rephasing and non-rephasing diagrams for a three-level system (Equ. 7.40). The factor 0.188 is to convert a frequency in cm-1 into one in ps-1.

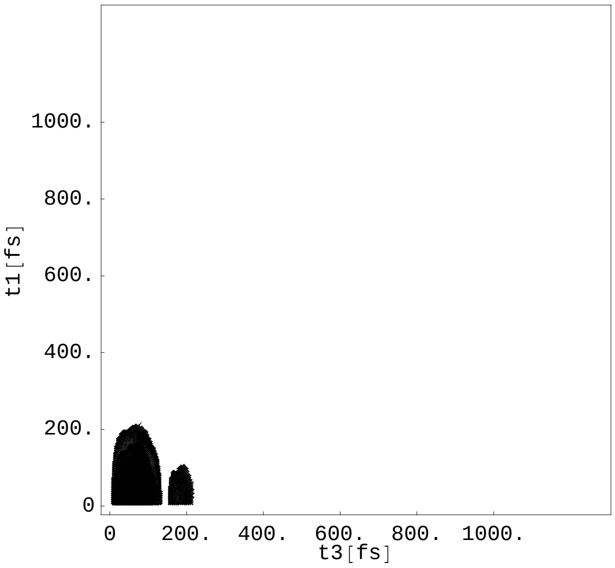
```
In[37]:= Rnr =
            (Exp[-I (t3 + t1) wmean * .188] - Exp[-I (t3 (wmean * .188 - Δ * .188) + t1 wmean * .188)] ) *
             Exp[-g[t1] - g[t2] - g[t3] + g[t1 + t2] + g[t2 + t3] - g[t1 + t2 + t3]];
         Rr = (Exp[-I (t3 - t1) wmean * .188] - Exp[-I (t3 (wmean * .188 - Δ * .188) - t1 wmean * .188)] )
             Exp[-g[t1] + g[t2] - g[t3] - g[t1 + t2] - g[t2 + t3] + g[t1 + t2 + t3]];
```

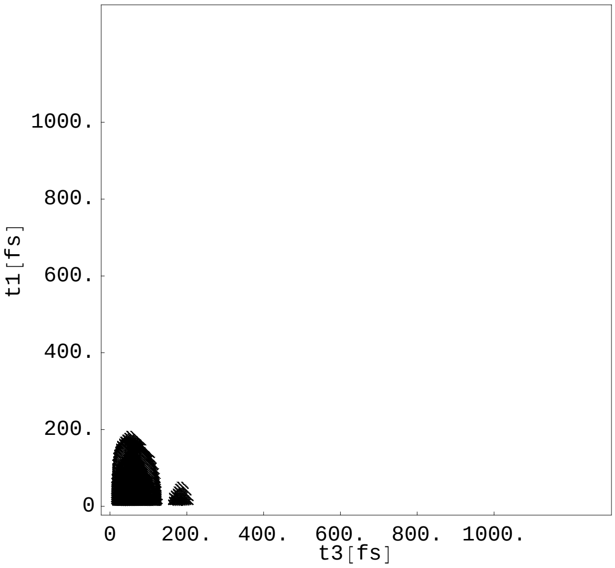
Collect rephasing and non-rephasing data on a grid with stepsize dt and nt data points with population time t2.The first time-point needs to be halved (Sect.9.5.3).

```
In[39]:= t2 = .5;
         nt = 512;
         dt = .0025;
         Rnrlist = Table[Rnr, {t1, 0, (nt - 1) * dt, dt}, {t3, 0, (nt - 1) * dt, dt}];
         Rrlist = Table[Rr, {t1, 0, (nt - 1) * dt, dt}, {t3, 0, (nt - 1) * dt, dt}];
         For[i = 1, i <= nt, i++, Rnrlist[[i, 1]] /= 2; Rrlist[[i, 1]] /= 2];
         For[i = 2, i <= nt, i++, Rnrlist[[1, i]] /= 2; Rrlist[[1, i]] /= 2];
```

Plot time-domain data (Fig. 10.4)

```
In[46]:= ticks = Table[{t / dt, t * 1000}, {t, 0, 1, .2}];
         ListContourPlot[Re[Rrlist], PlotRange → All, Contours → 20, ContourShading → False,
           FrameTicks → {ticks, ticks, None, None}, FrameLabel → {"t3[fs]", "t1[fs]"}];
         ListContourPlot[Re[Rnrlist], PlotRange → All, Contours → 20, ContourShading → False,
           FrameTicks → {ticks, ticks, None, None}, FrameLabel → {"t3[fs]", "t1[fs]"}];
```





Perform 2D Fourier transform and re-order data so that w1=w3=0 is centerd in the middle.Frequency axis w1 is inverted. Only the upper-right quadrant is kept

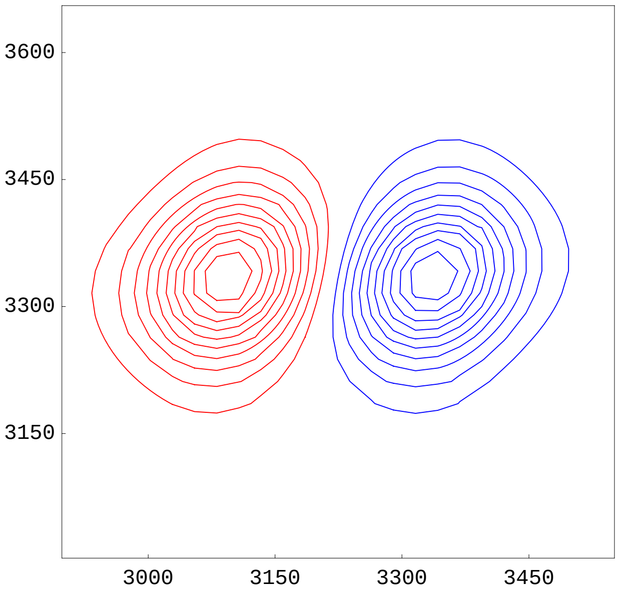
```
In[49]:= spectrum2Dr = Fourier[Rrlist];
         spectrum2Dr = Reverse[Drop[RotateRight[spectrum2Dr, {nt / 2, nt / 2}], 1, 1]];
         spectrum2Dr = Drop[spectrum2Dr, nt / 2 - 1, nt / 2 - 1];

         spectrum2Dnr = Fourier[Rnrlist];
         spectrum2Dnr = Drop[RotateRight[spectrum2Dnr, {nt / 2, nt / 2}], 1, 1];
         spectrum2Dnr = Drop[spectrum2Dnr, nt / 2 - 1, nt / 2 - 1];

         spectrum2Dabs = Re[spectrum2Dr + spectrum2Dnr];
```

Cut out the spectral range of interest

```
In[56]:= nminy = 115;
         nmaxy = 115;
         nminx = 111;
         nmaxx = 119;
         spectrum2Dabs = Drop[Drop[spectrum2Dabs, nminy, nminx], -nmaxy, -nmaxx];
```

Plot purely absorptive spectrum

```
In[61]:= max = Max[Max[spectrum2Dabs], -Min[spectrum2Dabs]];
         ticksy = Table[{ w * .188 * nt * dt / 2 / Pi + 1 - nminy, w}, {w, 0, 6000, 150}];
         ticksx = Table[{ w * .188 * nt * dt / 2 / Pi + 1 - nminx, w}, {w, 0, 6000, 150}];
         p1 = ListContourPlot[spectrum2Dabs, PlotRange → {0, max},
             ContourShading → False, Contours → 10, Ticks → None,
             ContourStyle → {RGBColor[0, 0, 1]}, DisplayFunction -> Identity];
         p2 = ListContourPlot[spectrum2Dabs, PlotRange → {-max, 0},
             ContourShading → False, Contours → 10, Ticks → None,
             ContourStyle → {RGBColor[1, 0, 0]}, DisplayFunction -> Identity];
         np = Length[spectrum2Dabs];
         Show[p1, p2, PlotRange → {{1, np}, {1, np}},
           FrameTicks → {ticksx, ticksy, None, None}, DisplayFunction → $DisplayFunction];
```


3600
3450
3300
3150
3000
3150
3300
3450