3 Measurements

3.1 Equilibrium

One of the important aspects when simulating fluid systems is characterizing equilibrium, since, before attempting to perform any measurement, the system should be allowed to reach equilibrium. Characterization of the equilibrium state is a quite delicate task, especially for small systems, whose properties fluctuate considerably in time. Physical quantities generally relax exponentially in time to their equilibrium averages, but the relaxation rates can be rather different for different quantities.

An useful indication for the equilibration is the rate at which the velocity distribution tends to the expected Maxwell distribution:

$$f(\mathbf{v}) = \rho \left(\frac{m}{2\pi k_B T} \right)^{3/2} \exp \left(-\frac{mv^2}{2k_B T} \right).$$

Upon integration over the angles, the distribution becomes:

$$f(v) \propto v^2 \exp \left(-\frac{mv^2}{2k_B T} \right). \tag{3.1}$$

The distribution function can be approximated numerically by the velocity histogram $\{f_n\}$, $n = 1, \ldots, N_{\text{hist}}$, where f_n represents the number of particles with the velocity magnitude in the interval $[(n-1)\triangle v, n\triangle v]$, $\triangle v = v_{\max}/N_{\text{hist}}$ is the velocity subinterval width, and $v_{\max}$ is an upper limit for the considered velocities. The normalized histogram represents a discrete approximation to $f(v)$.

The Fortran routine constructing the velocity histogram is listed below:

```
!===========================================================================
subroutine Vdistrib(vx,vy,vz,natm,vHist,fvHist,nvHist,vHistMax)
!---------------------------------------------------------------------------
!  Builds the normalized velocity distribution function (histogram)
!
!  vx,vy,vz - velocities of atoms
!  natm     - no. of atoms
!  vHist    - limits of velocity subintervals (bins)
!  fvHist   - relative no. of velocities in individual subintervals
!  nvHist   - no. of velocity subintervals
!  vHistMax - upper limit for considered velocities
!---------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   real(8) vx(natm), vy(natm), vz(natm)
   real(8) vHist(nvHist), fvHist(nvHist)

   delta = vHistMax / nvHist                                ! initialization
   do iHist = 1,nvHist
      vHist(iHist) = iHist * delta
      fvHist(iHist) = 0.d0
   end do

   do iatm = 1,natm                                              ! summation
      v = sqrt(vx(iatm)**2 + vy(iatm)**2 + vz(iatm)**2)
      if (v < vHistMax) then
         iHist = int(v/delta) + 1                             ! index of bin
         fvHist(iHist) = fvHist(iHist) + 1.d0
      end if
   end do

   sum = 0.d0                                                ! normalization
   do iHist = 1,nvHist
      sum = sum + fvHist(iHist)
   end do
   do iHist = 1,nvHist
      fvHist(iHist) = fvHist(iHist) / sum
   end do
end
```

A variant of this subroutine, constructing the *averaged* velocity histogram over a certain number of simulation steps, is the following:

```
!===========================================================================
subroutine VdistrAvg(vx,vy,vz,natm,vHist,fvHist,nvHist,vHistMax,iopt)
!---------------------------------------------------------------------------
!  Builds the normalized velocity distribution function (histogram)
!  averaged over an arbitrary number of simulation steps
!
!  vx,vy,vz - velocities of atoms
!  natm     - no. of atoms
!  vHist    - limits of velocity subintervals (bins)
!  fvHist   - relative no. of velocities in individual subintervals
!  nvHist   - no. of velocity subintervals
!  vHistMax - upper limit for considered velocities
!  iopt     - option: 0 - initialization
!                     1 - summation
!                     2 - normalization
!----------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   real(8) vx(natm), vy(natm), vz(natm)
   real(8) vHist(nvHist), fvHist(nvHist)

   select case (iopt)
      case (0)
         delta = vHistMax / nvHist                          ! initialization
         do iHist = 1,nvHist
           vHist(iHist) = iHist * delta
           fvHist(iHist) = 0.d0
         end do
      case (1)
         do iatm = 1,natm                                        ! summation
            v = sqrt(vx(iatm)**2 + vy(iatm)**2 + vz(iatm)**2)
            if (v < vHistMax) then
               iHist = int(v/delta) + 1                       ! index of bin
               fvHist(iHist) = fvHist(iHist) + 1.d0
            end if
         end do
      case (2)
         sum = 0.d0                                          ! normalization
         do iHist = 1,nvHist
            sum = sum + fvHist(iHist)
         end do
         do iHist = 1,nvHist
            fvHist(iHist) = fvHist(iHist) / sum
         end do
   end select
end
```

Equilibrium can be globally characterized by the Boltzmann H-function:

$$H(t) = \int f(\mathbf{v},t) \log f(\mathbf{v},t) d\mathbf{v}. \tag{3.2}$$

It can be demonstrated that, in general,

$$\left\langle \frac{dH}{dt} \right\rangle \leq 0$$

and the particular case

$$\left\langle \frac{dH}{dt} \right\rangle = 0$$

holds only for the Maxwell distribution.

To compute $H(t)$ numerically, the velocity histogram $\{f_n\}$ can be employed:

$$H(t) \simeq \sum_n f_n \log(f_n/v_n^2), \tag{3.3}$$

where constants have been neglected. The corresponding routine is:

```
!===========================================================================
function HBoltzmann(vHist,fvHist,nvHist)
!---------------------------------------------------------------------------
!  Evaluates the Boltzmann H-function from the normalized velocity histogram
!
!  vHist    - limits of velocity subintervals (bins)
!  fvHist   - relative no. of velocities in individual subintervals
!  nvHist   - no. of velocity subintervals
!---------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   real(8) vHist(nvHist), fvHist(nvHist)

   sum = 0.d0
   do iHist = 1,nvHist
      if (fvHist(iHist) > 0.d0) &
         sum = sum + fvHist(iHist) * log(fvHist(iHist) / vHist(iHist)**2)
   end do

   HBoltzmann = sum
end
```

3.2 Structure of simple fluids

Even though fluids are generally characterized by the absence of any permanent structure, they feature nevertheless well-defined structural correlations, reflecting the average molecular organization. These correlations are experimentally (directly or indirectly) accessible and provide a valuable link to theory.

Real space correlation functions are typically of the form:

$$\langle A(\mathbf{r})A(0)\rangle$$

and are straightforward to obtain by molecular dynamics: one has to compute the quantity of interest $A(\mathbf{r})$ starting from the atomic positions and velocities for several configurations, construct the correlation function for each configuration, and average over the available configurations.

The simplest example of a real space correlation function is the *pair correlation function* $g(\mathbf{r})$, which is essentially a *density-density correlation.* For spatially homogeneous systems only relative separations are meaningful and $g(\mathbf{r})$ has the form:

$$\rho g(\mathbf{r}) = \frac{1}{N}\left\langle \sum_{i=1}^{N}\sum_{j\neq i}^{N} \delta(\mathbf{r}-\mathbf{r}_{ij}) \right\rangle .$$

If the system is *isotropic*, the pair correlation function can be averaged over the angles and the result is the *radial distribution function* $g(r)$, which describes the averaged local organization around the atoms:

$$\rho g(r) = \frac{2}{N}\left\langle \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \delta(r-r_{ij}) \right\rangle \tag{3.4}$$

$g(r)$ is proportional to the probability to find a pair a distance r apart. More precisely, it is defined such that $\rho g(r)d\mathbf{r}$ is proportional to the probability of finding an atom in the volume element $d\mathbf{r}$ at a distance r from a given atom. With this definition, the quantity $4\pi\rho g(r)r^2\Delta r$ specifies the average number of atoms in the shell of radius r and thickness Δr centered at a given atom.

A useful numerical approximation for the radial distribution function may be obtained by using the histogram of the pair separations $\{f_n\}$, with f_n representing the number of atom pairs (i,j) with the interatomic distance r_{ij} contained within the nth bin, $(n-1)\triangle r \leq r_{ij} \leq n\triangle r$. If N_{hist} is the number of histogram bins (distance subintervals) and $r_{\max}$ is the upper limit for the considered distances, then the bin width is $\triangle r = r_{\max}/N_{\text{hist}}$ and the following approximation for $g(r)$ results:

$$4\pi\rho g(r_n)r_n^2\Delta r = \frac{2}{N}f_n,$$

or, defining the average distance in interval $[(n-1)\triangle r, n\triangle r]$ as $r_n = (n-1/2)\Delta r$,

$$g(r_n) = \frac{V}{2\pi N^2(n-1/2)^2(\Delta r)^3}\, f_n,\ n = 1,\ldots,N_{\text{hist}}. \tag{3.5}$$

The straightforward implementation of this procedure can be found in the following subroutine.

```
!==========================================================================
subroutine Rdistrib(x,y,z,natm,rHist,frHist,nrHist,rHistMax,dcell)
!--------------------------------------------------------------------------
!  Builds the radial distribution function (histogram)
```

```
!
!  x, y, z  - coordinates of atoms
!  natm     - no. of atoms
!  rHist    - limits of interatomic distance subintervals (bins)
!  frHist   - no. of distances in individual subintervals
!  nrHist   - no. of distance subintervals
!  rHistMax - upper limit for considered interatomic distances
!-----------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   parameter (pi = 3.14159265358979323850d0)

   real(8) x(natm), y(natm), z(natm)
   real(8) rHist(nrHist), frHist(nrHist)

   delta = rHistMax / nrHist                                  ! initialization
   do iHist = 1,nrHist
      rHist(iHist) = iHist * delta
      frHist(iHist) = 0.d0
   end do

   do iatm = 1,natm-1                                              ! summation
      do jatm = iatm+1,natm
         dx = x(iatm) - x(jatm)
         dy = y(iatm) - y(jatm)
         dz = z(iatm) - z(jatm)
                                                       ! minimum image criterion
         if (abs(dx) > 0.5d0*dcell) dx = dx - sign(dcell,dx)
         if (abs(dy) > 0.5d0*dcell) dy = dy - sign(dcell,dy)
         if (abs(dz) > 0.5d0*dcell) dz = dz - sign(dcell,dz)

         r = sqrt(dx*dx + dy*dy + dz*dz)
         if (r < rHistMax) then
            iHist = int(r/delta) + 1                                ! index of bin
            frHist(iHist) = frHist(iHist) + 1.d0
         end if
      end do
   end do

   fact = dcell**3 / (2.d0*pi*natm**2*delta**3)
   do iHist = 1,nrHist
      frHist(iHist) = fact * frHist(iHist)/(iHist-0.5)**2
   end do
end
```

For a crystal, the radial distribution function exhibits a sequence of peaks at positions corresponding to shells around a given atom. The positions and magnitudes of the peaks are a "signature" of the crystal structure (fcc, bcc etc.). For a liquid, $g(r)$ exhibits a major peak close to the average atomic separation and oscillates with exponentially decaying peaks at larger distances. In all cases, $g(r)$ vanishes below a certain distance, where atomic repulsion prevents atoms from approaching any further, and the normalization factor ensures that $g(r \to \infty) = 1$, even though periodic boundary conditions limit

$r_{\max}$ to no more than half the smallest edge of the simulation region, with wraparound in evaluating interatomic distances.

The central role played by the radial distribution function in liquid state physics is due to the fact that all functions depending on the pair separation solely (potential energy, pressure etc.) can be expressed as integrals involving $g(r)$.

The pair correlation function $g(\mathbf{r})$ is experimentally not directly accessible, but it is related through the Fourier transform to the experimentally measurable structure factor $S(\mathbf{k})$,

$$S(\mathbf{k}) = 1 + \rho \int g(\mathbf{r}) \exp(-\mathbf{i}\mathbf{k} \cdot \mathbf{r})\ d\mathbf{r},$$

which is central to the interpretation of X-ray scattering measurements. For isotropic liquids, the relation involves the radial distribution function $g(r)$:

$$S(k) = 1 + 4\pi\rho \int \frac{\sin kr}{kr} g(r) r^2 dr. \tag{3.6}$$

3.3 Diffusion

In a continuous system, the diffusion coefficient D is defined by Fick's law, relating mass flow to density gradient:

$$\mathbf{j} = -D\nabla\rho, \tag{3.7}$$

where $\mathbf{j}(\mathbf{r}, t)$ is the local current density and $\rho(\mathbf{r}, t)$ the local density (concentration). Having in view the continuity equation,

$$\frac{\partial\rho}{\partial t} + \nabla \cdot \mathbf{j} = 0, \tag{3.8}$$

the time-evolution of ρ is described by the equation:

$$\frac{\partial\rho}{\partial t} = D\nabla^2\rho. \tag{3.9}$$

This equation applies both to self-diffusion of a single species and to diffusion of different species through one another.

For a "fluid" made up of N discrete particles, the density ρ reads:

$$\rho(\mathbf{r}, t) = \sum_{i=1}^{N} \delta(\mathbf{r} - \mathbf{r}_i(t)).$$

For large time intervals (compared to the typical interatomic collision times) the Einstein expression for the diffusion coefficient holds:

$$D = \lim_{t\to\infty} \frac{1}{6Nt} \left\langle \sum_{i=1}^{N} [\mathbf{r}_i(t) - \mathbf{r}_i(0)]^2 \right\rangle \tag{3.10}$$

Since this expression implies the "true" atomic displacements, when periodic boundary conditions are applied, the real displacements have to be used, from which the effects of wraparound have been removed.

The alternative Green-Kubo expression of the diffusion coefficient,

$$D = \lim_{t\to\infty} \frac{1}{3N} \int_0^t \left\langle \sum_{i=1}^{N} \mathbf{v}_i(t) \cdot \mathbf{v}_i(0) \right\rangle dt \tag{3.11}$$

involves the integrated velocity autocorrelation function.

The next Fortran routine implements the Green-Kubo formula for the diffusion coefficient.

```
!============================================================================
function DiffCoef(vx,vy,vz,vx0,vy0,vz0,natm,dt)
!----------------------------------------------------------------------------
!  Evaluates the diffusion coefficient from the Green-Kubo formula:
!               1  inf   N
!  D =  lim   --- Int <Sum v(t)v(0)> dt
!      t->inf 3N   0   i=1
!
!  vx,vy,vz - velocities of atoms
!  natm     - no. of atoms
!----------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   real(8)  vx(natm),  vy(natm),  vz(natm)
   real(8) vx0(natm), vy0(natm), vz0(natm)
   logical init /.true./

   if (init) then
      Diff = 0.d0
      fact = dt/(3.d0*natm)
      init = .false.
   end if

   do iatm = 1,natm
      Diff = Diff + fact*(vx(iatm)*vx0(iatm) + &
                          vy(iatm)*vy0(iatm) + &
                          vz(iatm)*vz0(iatm))
   end do
   DiffCoef = Diff
end
```

3.4 Vibrational spectra of clusters

In order to characterize the overall vibration of a cluster, the *velocity autocorrelation function* may be employed:

$$C_{\mathbf{v}}(t) = \frac{\sum_{i=1}^{N} \langle \mathbf{v}_i(t) \cdot \mathbf{v}_i(0) \rangle}{\sum_{i=1}^{N} (\mathbf{v}_i(0))^2},$$

where $\mathbf{v}_i(t)$ is the velocity of atom i at time t. This is basically a normalized projection of the atomic velocity pattern at a particular time t onto the initial pattern.

An alternative to $C_{\mathbf{v}}(t)$ is the *displacement autocorrelation function*:

$$C_{\delta\mathbf{r}}(t) = \frac{\sum_{i=1}^{N} \langle \delta\mathbf{r}_i(t) \cdot \delta\mathbf{r}_i(0) \rangle}{\sum_{i=1}^{N} (\delta\mathbf{r}_i(0))^2},$$

with $\delta\mathbf{r}_i(t)$ representing the displacement of atom i at time t. If the initial preparation of the vibrational state is to obey some predefined pattern, the latter autocorrelation function may be sometimes more convenient. Aside from this aspect, the information contained in the two autocorrelation functions is absolutely equivalent.

The vibration of molecules with an inversion point can be additionally characterized by the *parity*, defined as:

$$P = \frac{\sum_{I=1}^{N} \langle \delta\mathbf{r}_I \cdot (-\delta\mathbf{r}_{\mathrm{sym}(I)}) \rangle}{\sum_{I=1}^{N} (\delta\mathbf{r}_I)^2},$$

where sym(I) designates the symmetric atom for atom I with respect to the inversion point.

The following routine implements the velocity autocorrelation function:

```
!===========================================================================
function FuncCorr(vx,vy,vz,vx0,vy0,vz0,natm)
!---------------------------------------------------------------------------
!  Velocity autocorrelation function
   implicit real(8) (a-h,o-z)

   real(8)  vx(natm),  vy(natm),  vz(natm)
   real(8) vx0(natm), vy0(natm), vz0(natm)
   logical :: init = .true.

   if (init) then
      sum0 = 0.d0
      do iatm = 1,natm
         vx0(iatm) = vx(iatm)
         vy0(iatm) = vy(iatm)
         vz0(iatm) = vz(iatm)

         sum0 = sum0 + vx0(iatm) * vx0(iatm) &
                     + vy0(iatm) * vy0(iatm) &
                     + vz0(iatm) * vz0(iatm)
      end do

      init = .false.
   end if

   sum = 0.d0
   do iatm = 1,natm
      sum = sum + vx0(iatm) * vx(iatm) &
                + vy0(iatm) * vy(iatm) &
                + vz0(iatm) * vz(iatm)
   end do

   FuncCorr = sum / sum0
end
```

The Fourier-transform (power spectrum) of $C_{\mathbf{v}}(t)$ (or $C_{\delta\mathbf{r}}(t)$) can be straightforwardly identified with the *vibrational spectrum* of the cluster.

```
!==============================================================================
subroutine Spectrum(CorVel,Intens,freq,freqMin,freqMax,dt,nstep,nfreq)
!------------------------------------------------------------------------------
!  Power spectrum of velocity auto-correlation function
   implicit real(8) (a-h,o-z)

   real(8), parameter :: psec = 1.d-12                              ! time unit
   real(8), parameter :: c = 2.99792458d10                     ! speed of light

   real(8) CorVel(nstep), Intens(nstep), freq(nstep)
   real(8) IntensMax

   if (nstep <= 2) stop 'Fourier: nstep <= 2'

   call FTcos(CorVel,Intens,nstep)                        ! Fourier transform

   dfreq = 1.d0 / (2.d0*(nstep-1)*dt*c*psec)            ! frequency increment

   IntensMax = 0d0
   nfreq = 0
   do i = 1,nstep
      freqi = (i-1) * dfreq
      if ((freqMin <= freqi).and.(freqi <= freqMax)) then
         nfreq = nfreq + 1
         freq(nfreq) = freqi
         Intens(nfreq) = abs(Intens(i))
         IntensMax = max(Intens(nfreq),IntensMax)
      end if
   end do

   do i = 1,nfreq                                      ! normalize intensities
      Intens(i) = Intens(i) / IntensMax
   end do
end
```

Bibliografie

[1] Rapaport, D.C., *The Art of Molecular Dynamics Simulation*, Cambridge University Press, Cambridge, 1995

[2] Gould, H. and Tobochnik, J., *An Introduction to Computer Simulation Methods. Theory, Algorithms and Object-Orientation*, Addison-Wesley Publishing Company, Reading, 1996.

[3] Sadus, R.J., *Molecular Simulation of Fluids. Applications to Physical Systems. Second Edition*, Elsevier, Amsterdam, 1999.

[4] Ercolessi, F., *A molecular dynamics primer*, http://www.sissa.it/furio/