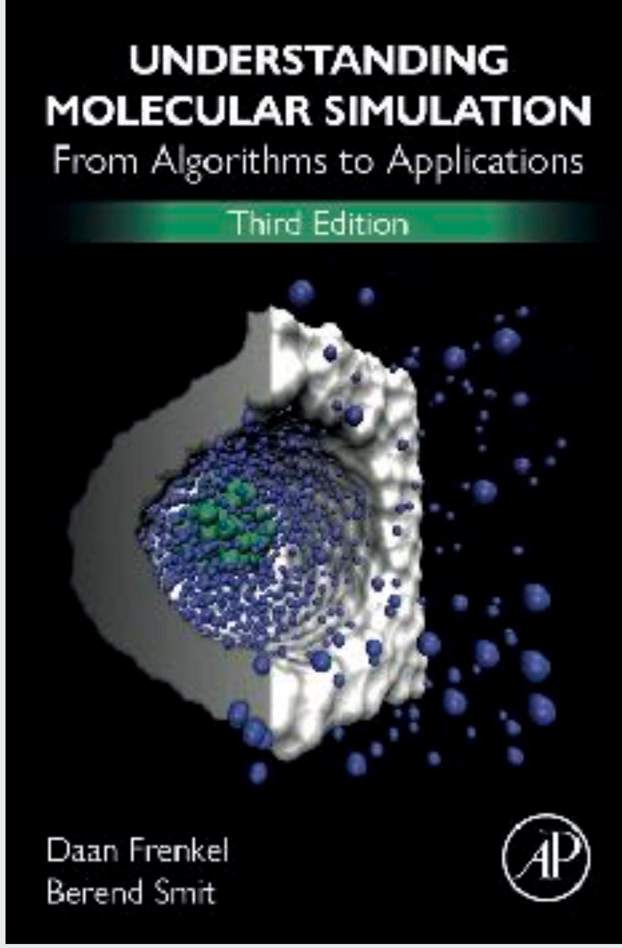
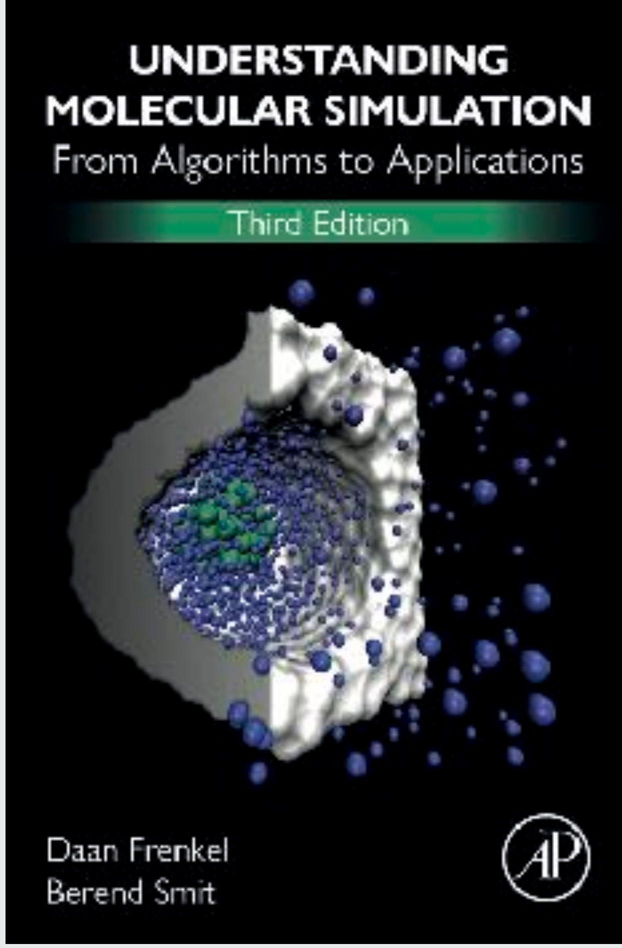
4. Molecular Dynamics



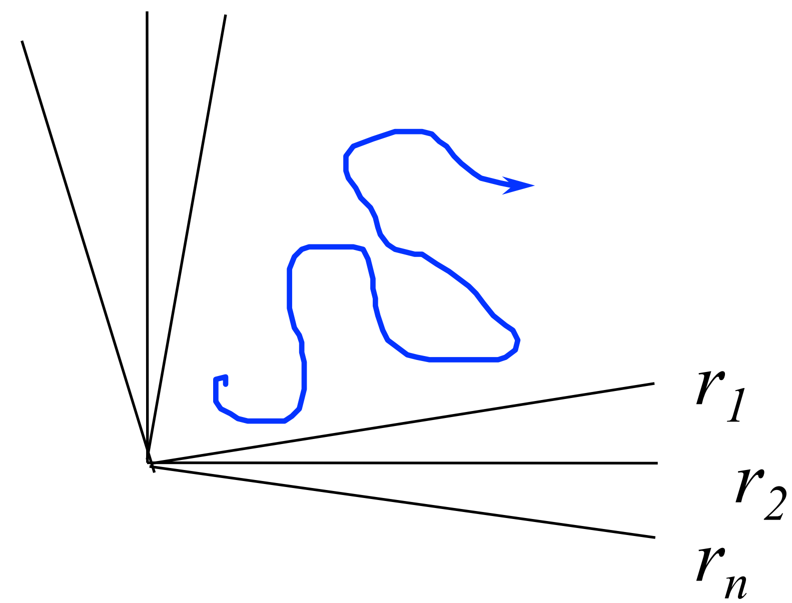


4. Molecular Dynamics

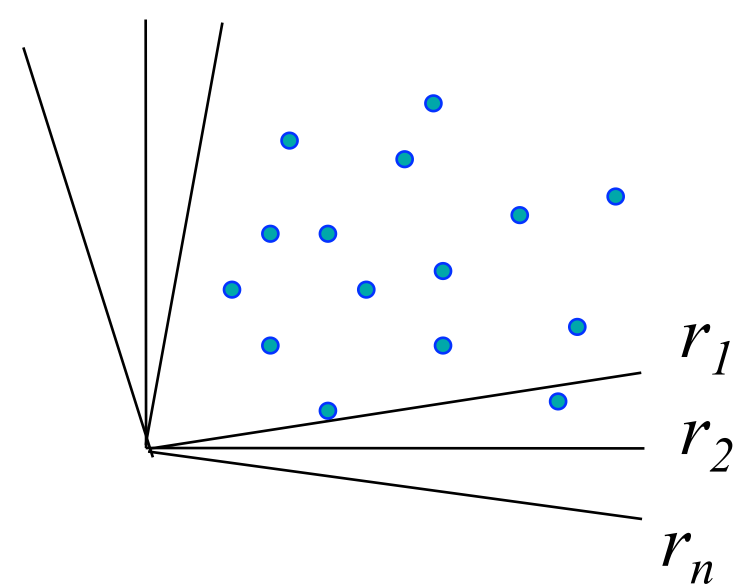
4.1 Basics

Molecular Simulations

- ➡ **Molecular dynamics**: solve equations of motion



- ➡ **Monte Carlo**: importance sampling



Molecular Dynamics

4. Molecular Dynamics
    4.1. Basics
    4.2. Liouville formulation
    4.3. Multiple time steps

“Fundamentals”

Theory:

$$F = m\frac{d^2r}{dt^2}$$

- Compute the forces on the particles
- Solve the equations of motion
- Sample after some # of time steps

Molecular Dynamics

Algorithm 3 (Core of Molecular Dynamics program)

program MD	basic MD code
`[...]`	
setlat	function to initialize positions x
initv`(temp)`	function to initialize velocities vx
`t=0`	
while `(t < tmax)` **do**	main MD loop
FandE	function to compute forces and total energy
Integrate-V	function to integrate equations of motion
`t=t+delt`	update time
sample	function to sample averages
enddo	
end program	

Molecular Dynamics

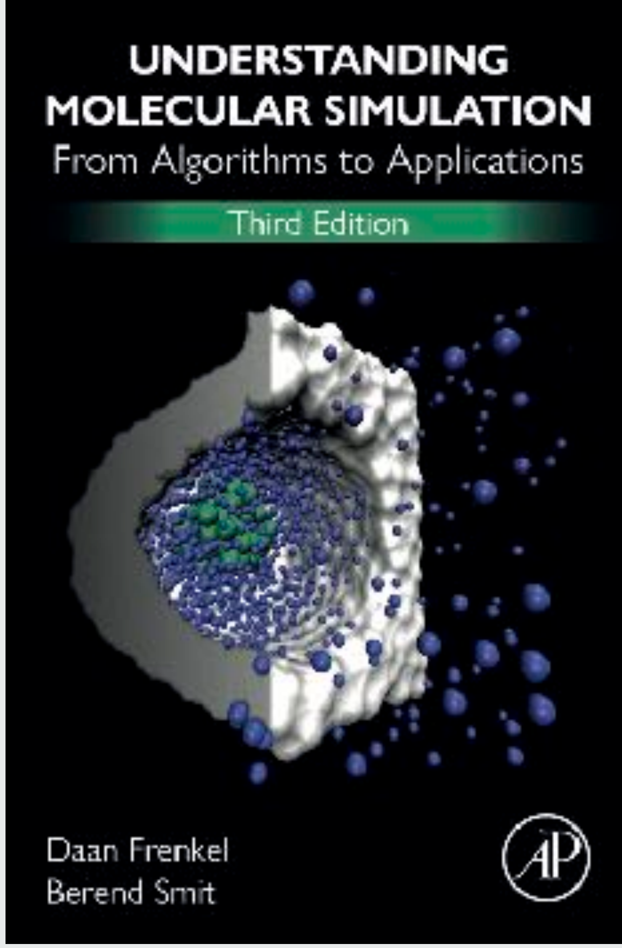
Initialization

Force calculations

- Periodic boundary conditions
- Order NxN and order N algorithms,
- Truncation and shift of the potential

Integrating the equations of motion

- integration schemes



4. Molecular Dynamics

4.1.1 Basics: Initialization

Algorithm 4 (Initialization of a Molecular Dynamics program)

function initv(*temp*)	initializes velocities for MD program
`sumv=0`	
`sumv2=0`	
for $1 \le i \le$ npart **do**	
`x(i) = lattice_pos(i)`	Place the particle on a lattice
`vx(i) =` $\sqrt{-\ln(\mathcal{R})}\cos(2\pi\mathcal{R})$	Generate 1D normal distribution
`sumv=sumv+v(i)`	center of mass momentum ($m = 1$)
enddo	
`sumv=sumv/npart`	center of mass velocity
for $1 \le i \le$ npart **do**	set desired kinetic energy and set
`vx(i) = vx(i) - sumv`	Center of Mass velocity to zero
`sumv2=sumv2+vx(i)**2`	kinetic energy
enddo	
`fs=`$\sqrt{\texttt{temp}/(\texttt{sumv2}/\texttt{nf})}$	`temp` = desired initial temperature
for $1 \le i \le$ npart **do**	
`vx(i)=vx(i)*fs`	set initial kinetic temperature
`xm(i)=x(i)-vx(i)*dt`	position previous time step
enddo	
end function	

Molecular Dynamics

Initialization

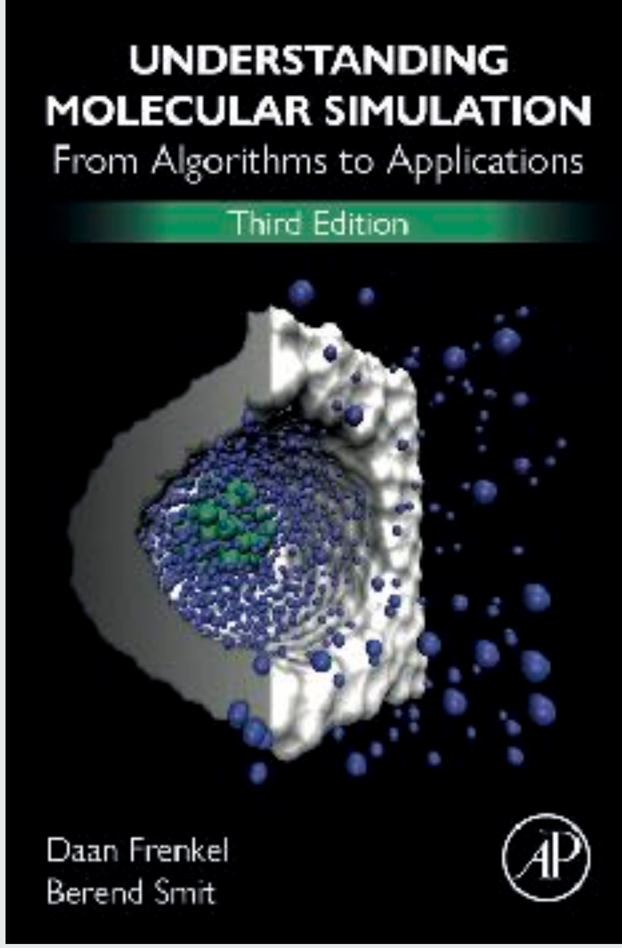
- Total momentum should be zero (no external forces)
- Temperature rescaling to desired temperature
- Particles start on a lattice

Force calculations

- Periodic boundary conditions
- Order NxN algorithm,
- Order N: neighbor lists, linked cell
- Truncation and shift of the potential

Integrating the equations of motion

- Velocity Verlet
- Kinetic energy



4. Molecular Dynamics

4.1.2 Basics: Force Calculation

Molecular Dynamics

Initialization

- Total momentum should be zero (no external forces)
- Temperature rescaling to desired temperature
- Particles start on a lattice

Force calculations

- Periodic boundary conditions
- Order NxN algorithm,
- Order N: neighbor lists, linked cell
- Truncation and shift of the potential

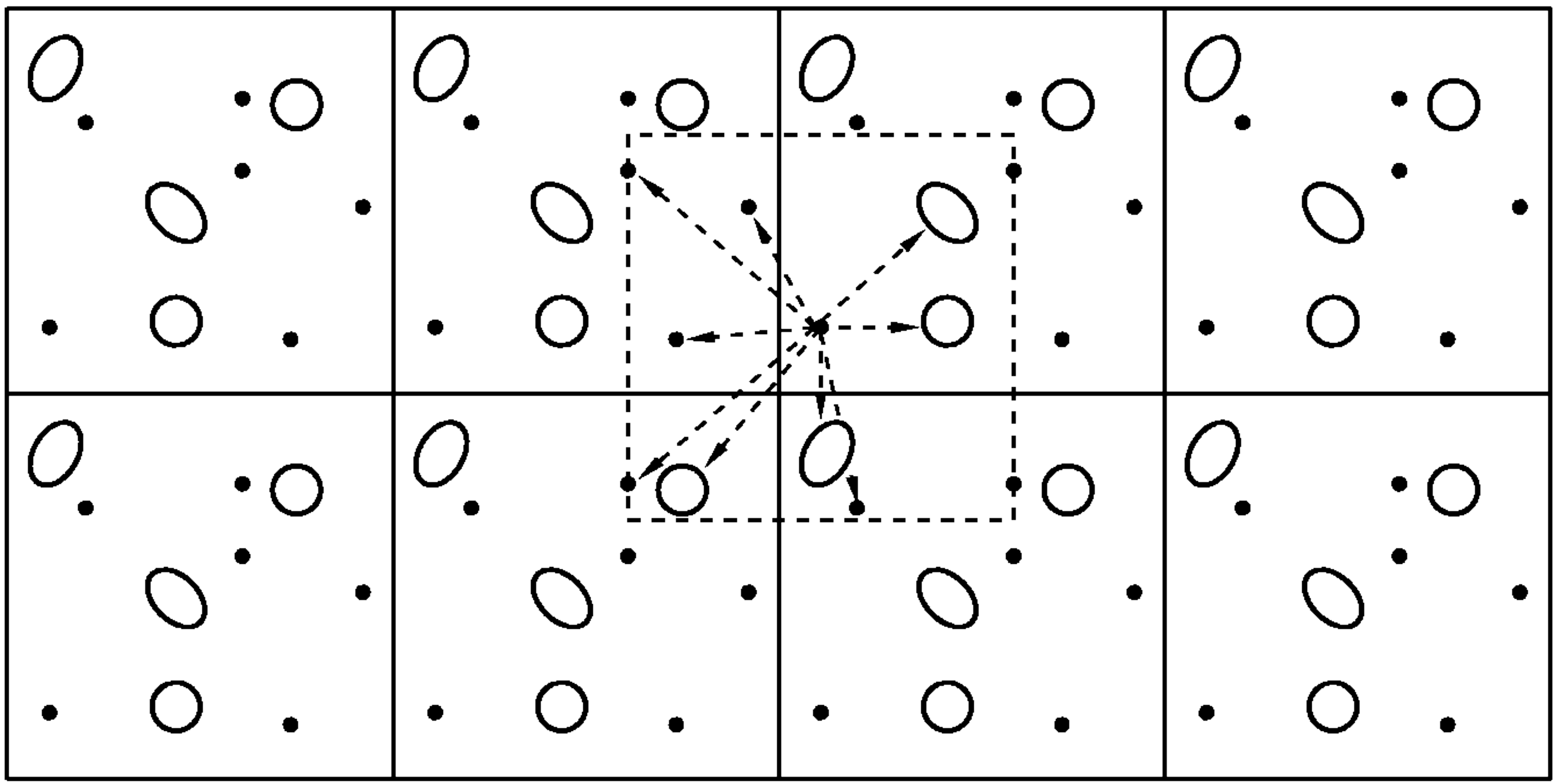
Integrating the equations of motion

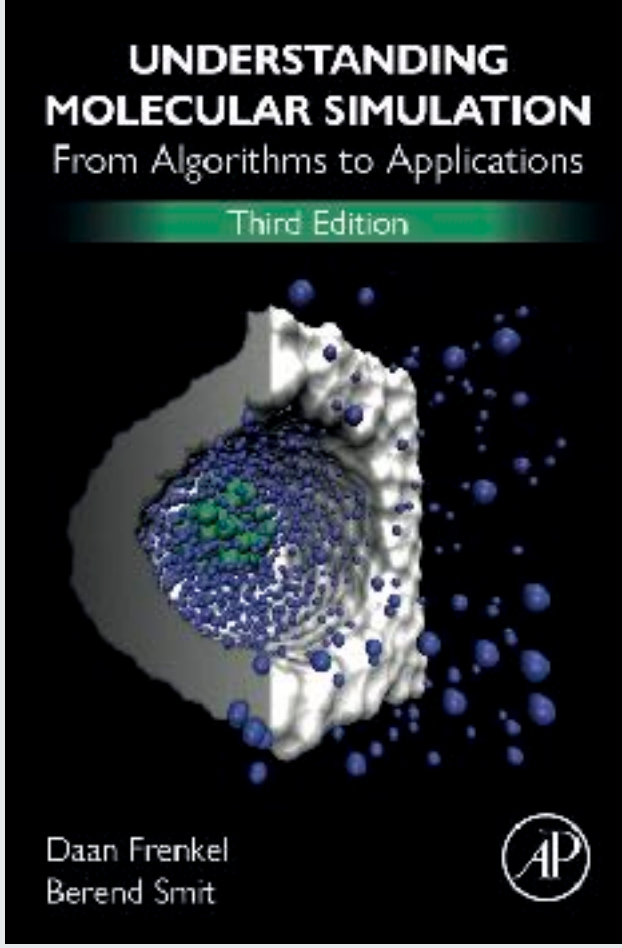
- Velocity Verlet
- Kinetic energy

Algorithm 5 (Calculation of pair forces and energy forces)

```
function FandE                               determine forces and energy
 rc2=rc**2                                   rc=2 is the default cut-off
en=0                                         set energy to zero
for 1 ≤ i ≤ npart do
    fx(i)=0                                  set forces to zero
enddo
 for 1 ≤ i ≤ npart-1 do
 for i+1 ≤ j ≤ npart do                      loop over all pairs
    xr=x(i)-x(j)
    xr=xr-box*round(xr/box)                  nearest image distance
    r2=xr**2
    if r2 <rc2 then                          test cutoff
        r2i=1/r2
        r2im1=r2i-1.0
        rc2r2im1=rc2*r2i-1.0
        en=en+r2im1*rc2r2im1**2              pair energy
        ff=6.0*r2i**2*rc2r2im1
             *(rc2r2im1-2)                   pair force
        fx(i)=fx(i)+ff*xr
        fx(j)=fx(j)-ff*xr
    endif
 enddo
enddo
end function
```

Periodic boundary conditions





4. Molecular Dynamics

4.1.2 Basics: Force Calculation - The Lennard Jones potential

The Lennard-Jones potentialS

- The Lennard-Jones potential

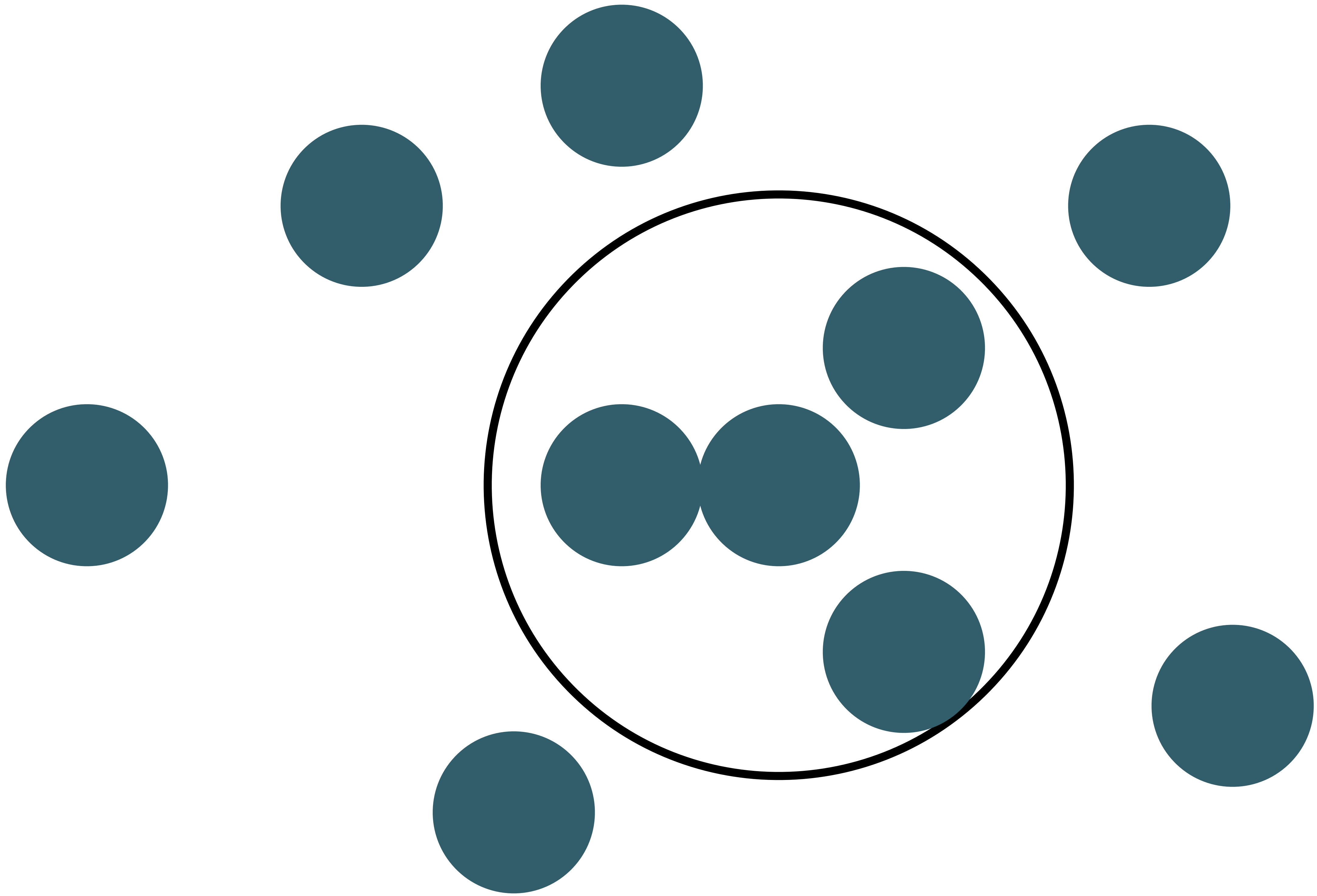
$$U^{LJ}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

- The truncated Lennard-Jones potential

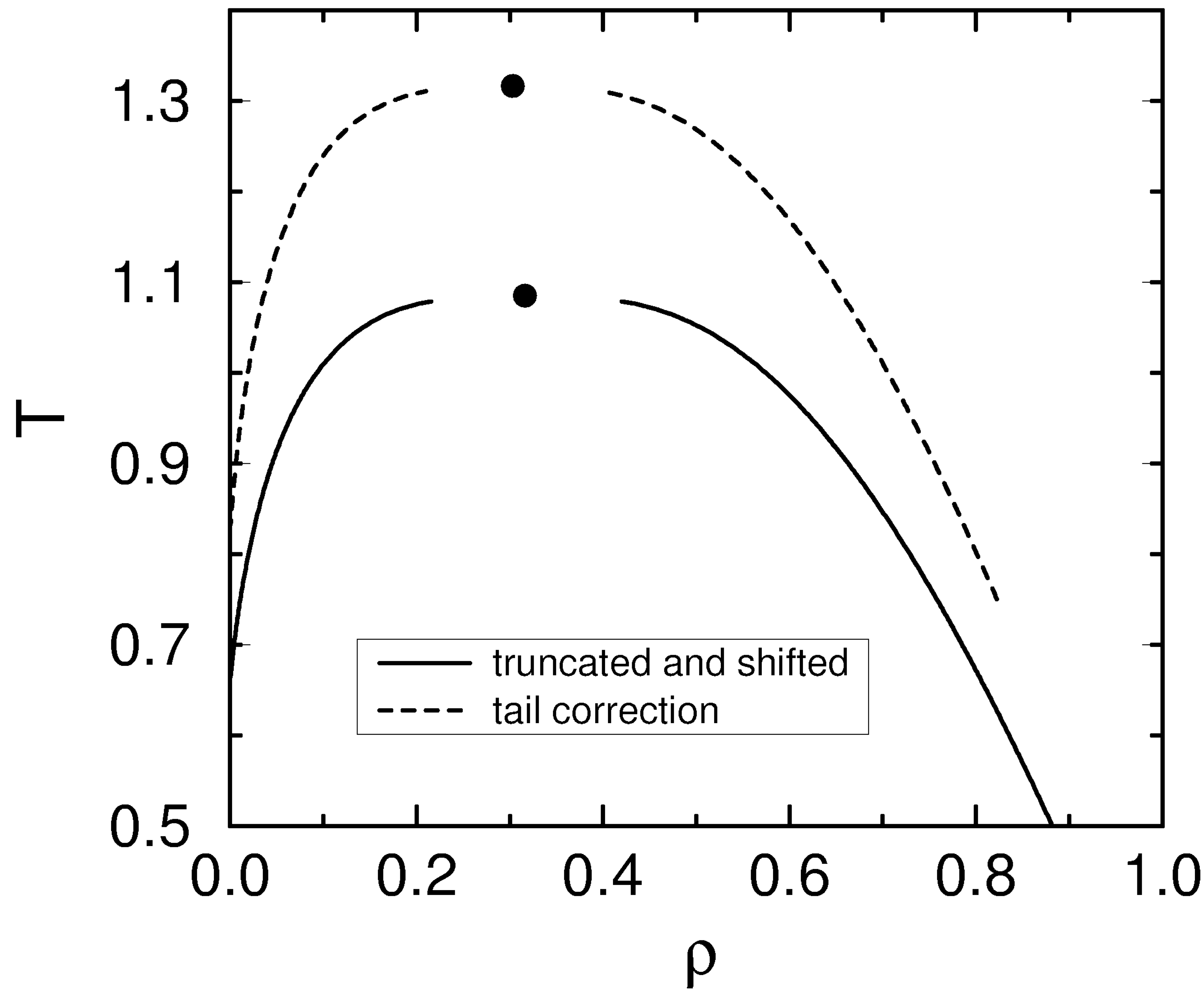
$$U^{LJ}_{TR}(r) = \begin{cases} U^{LJ}(r) & r \le r_c \\ 0 & r > r_c \end{cases}$$

- The truncated and shifted Lennard-Jones potential

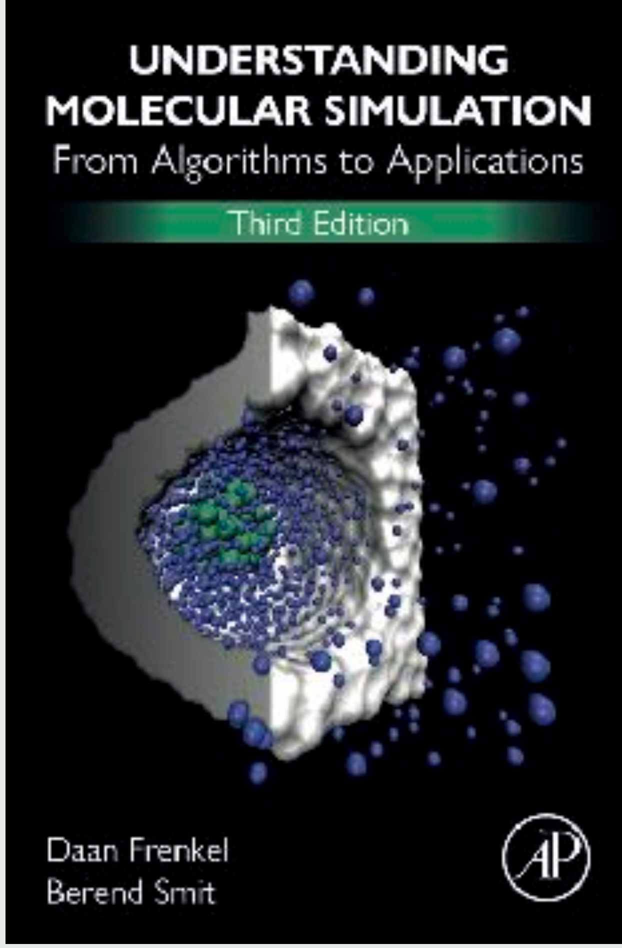
$$U^{LJ}_{TR-SH}(r) = \begin{cases} U^{LJ}(r) - U^{LJ}(r_c) & r \le r_c \\ 0 & r > r_c \end{cases}$$



The Lennard-Jones potentials



Understanding Molecular Simulation

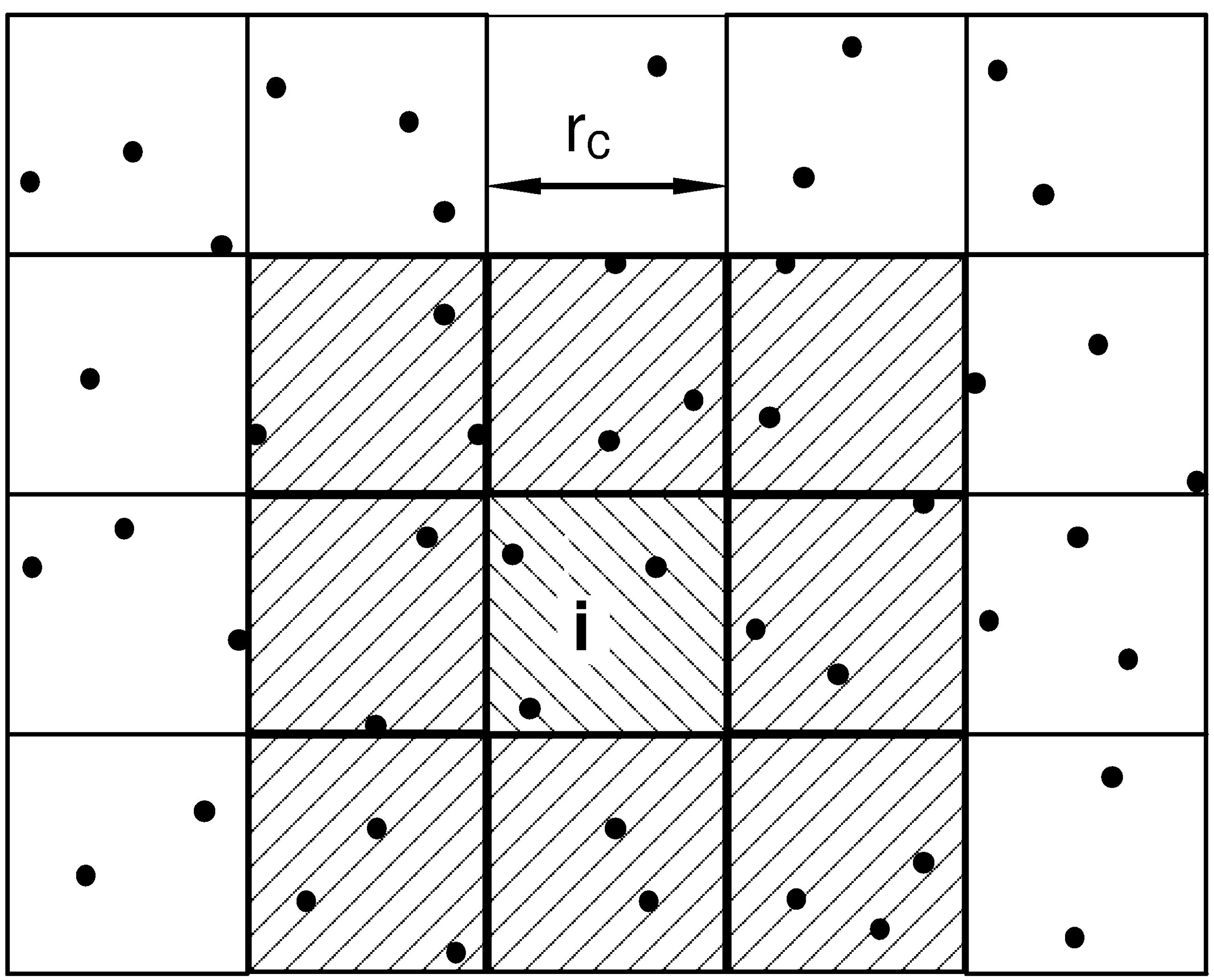


4. Molecular Dynamics

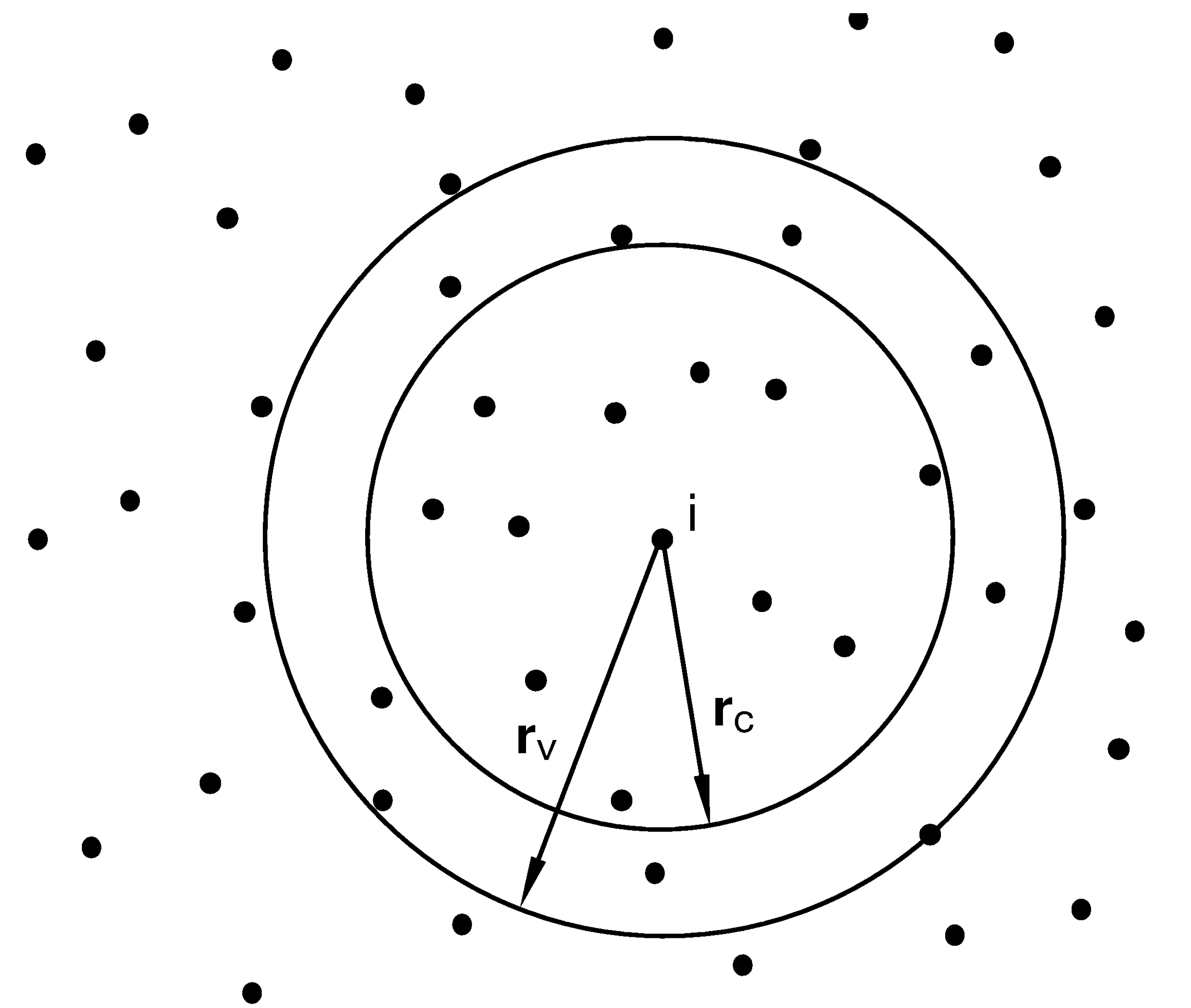
4.1.2 Basics: Force Calculation - saving CPU time

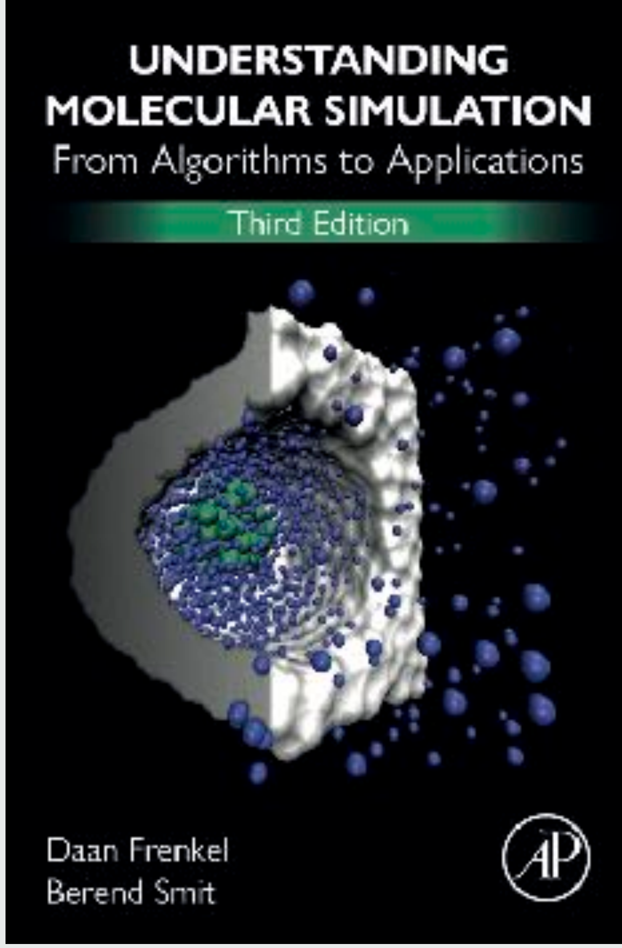
Saving CPU-time

Cell list



Verlet-list





4. Molecular Dynamics

4.1.3 Basics: Equations of Motion

Algorithm 6 (Integrating the equations of motion)

```
function Integrate-V                          integrate equations of motion
sumv=0
sumv2=0
for 1 ≤ i ≤ npart do                          MD loop
    xx=2*x(i)-xm(i)+delt**2*fx(i)             Verlet algorithm (4.2.3)
    vi=(xx-xm(i))/(2*delt)                    velocity (4.2.4)
    sumv=sumv+vi                              velocity center of mass
    sumv2=sumv2+vi**2                         total kinetic energy
    xm(i)=x(i)                                update "old" positions
    x(i)=xx                                   update "current" positions
enddo
temp=sumv2/(nf)                               current temperature
etot=(en+0.5*sumv2)/npart                     and total energy per particle
end function                                  may be used elsewhere
```

Equations of motion

We can make a Taylor expansion for the positions:

$$r(t+\Delta t) = r(t) + \frac{dr(t)}{dt}\Delta t + \frac{d^2 r(t)}{dt^2}\frac{\Delta t^2}{2!} + O(\Delta t^3)$$

The simplest form (Euler):

$$r(t+\Delta t) = r(t) + v(t)\Delta t + O(\Delta t^2)$$

$$v(t+\Delta t) = v(t) + m\frac{df(t)}{dt}\Delta t$$

We can do better!

We can make a Taylor expansion for the positions:

$$r(t+\Delta t) = r(t) + \frac{dr(t)}{dt}\Delta t + \frac{d^2r(t)}{dt^2}\frac{\Delta t^2}{2!} + \frac{d^2r(t)}{dt^2}\frac{\Delta t^3}{3!} + O(\Delta t^4)$$

$$r(t-\Delta t) = r(t) - \frac{dr(t)}{dt}\Delta t + \frac{d^2r(t)}{dt^2}\frac{\Delta t^2}{2!} - \frac{d^2r(t)}{dt^2}\frac{\Delta t^3}{3!} + O(\Delta t^4)$$

When we add the two:

$$r(t+\Delta t) + r(t-\Delta t) = 2r(t) + \frac{d^2r(t)}{dt^2}\Delta t^2 + O(\Delta t^4)$$

Verlet algorithm

$$r(t+\Delta t) = 2r(t) - r(t-\Delta t) + f(t)\frac{\Delta t^2}{m} + O(\Delta t^4)$$

numerically not ideal

no need for velocities

Verlet algorithm: $$r(t+\Delta t)=2r(t)-r(t-\Delta t)+f(t)\frac{\Delta t^2}{m}+O(\Delta t^4)$$

Velocity Verlet algorithm

$$\boxed{\begin{aligned} r(t+\Delta t)&=r(t)+v(t)\Delta t+f(t)\frac{\Delta t^2}{2m}+O(\Delta t^4) \\ v(t+\Delta t)&=v(t)+\frac{\Delta t}{2m}\left[f(t+\Delta t)+f(t)\right] \end{aligned}}$$

to see the equivalence:

$$r(t+2\Delta t)=r(t+\Delta t)+v(t+\Delta t)\Delta t+f(t+\Delta t)\frac{\Delta t^2}{2m}$$

$$r(t)=r(t+\Delta t)-v(t)\Delta t-f(t)\frac{\Delta t^2}{2m}$$

adding the two

$$r(t+2\Delta t)=2r(t+\Delta t)-r(t)+\left[v(t+\Delta t)-v(t)\right]\Delta t+\left[f(t+\Delta t)-f(t)\right]\frac{\Delta t^2}{2m}$$

with $$v(t+\Delta t)=v(t)+\frac{\Delta t}{2m}\left[f(t+\Delta t)+f(t)\right]$$

$$r(t+2\Delta t)=2r(t+\Delta t)-r(t)+f(t+\Delta t)\frac{\Delta t^2}{m}$$

Lyaponov instability

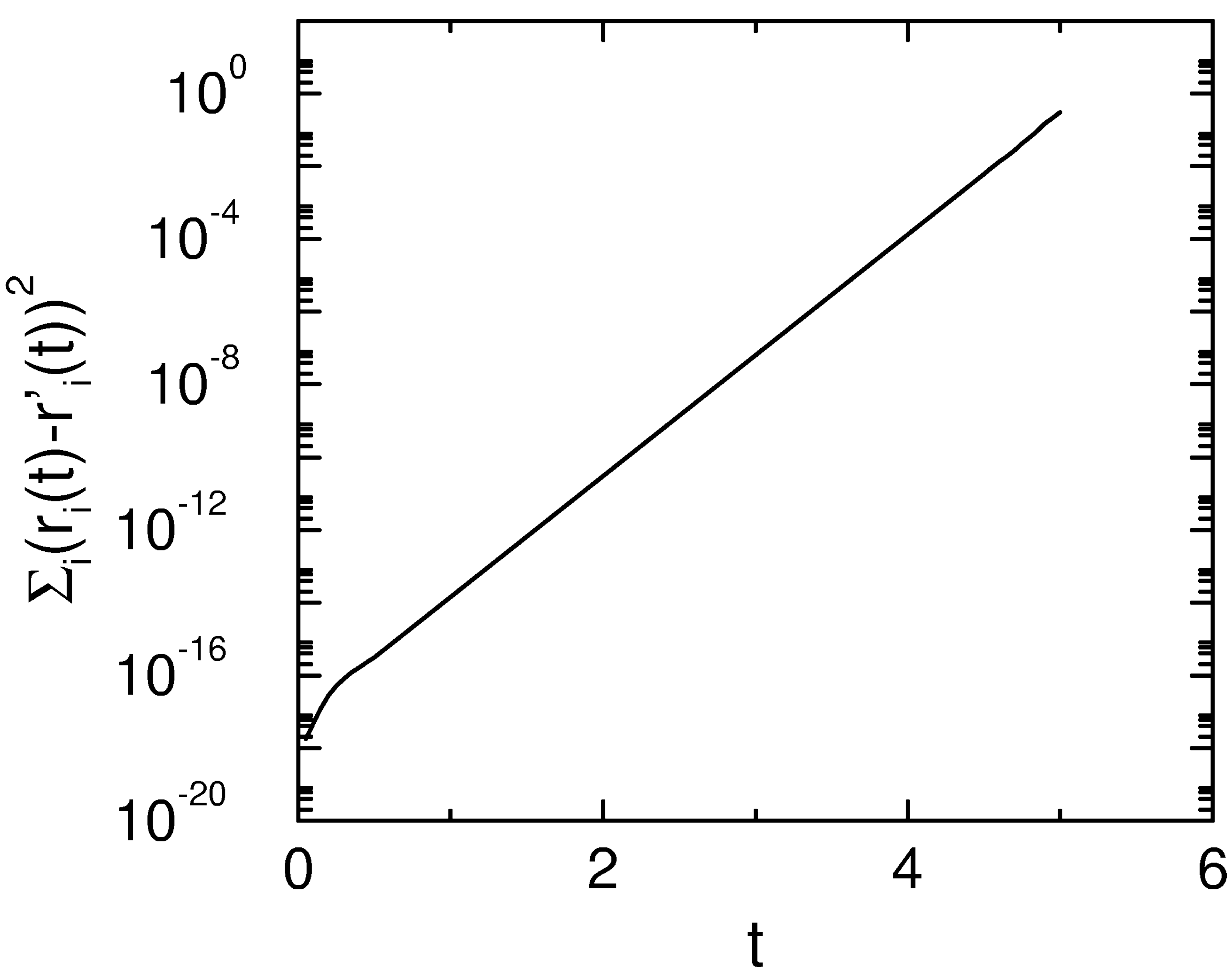
MD: reference trajectory with initial condition:

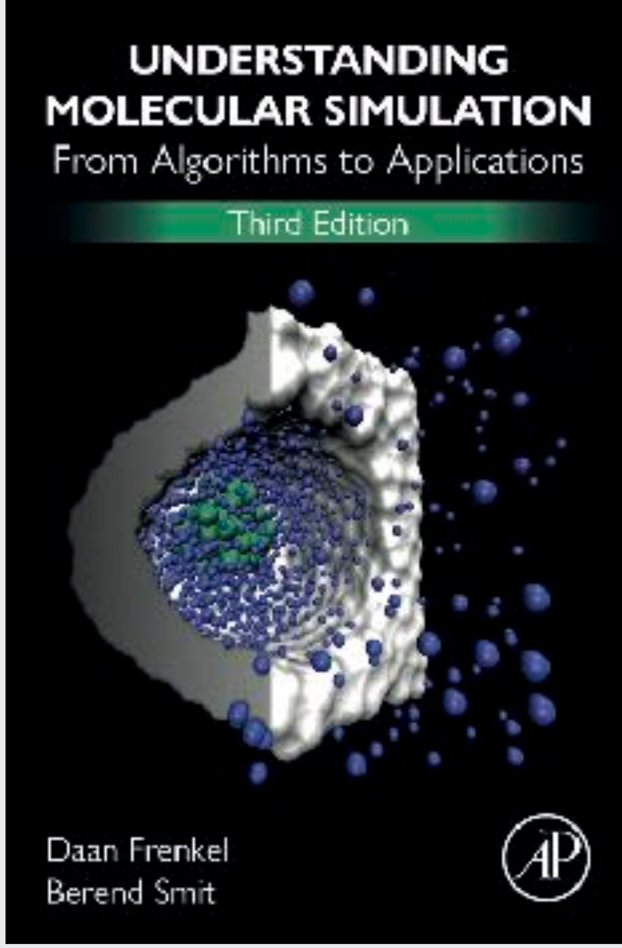
$$\left(r_1(0),\cdots,r_N(0),p_1(0),\cdots,p_N(0)\right)$$

MD: compare:

$$\left(r_1(0),\cdots,r_N(0),p_1(0),\cdots,p_i(0)+\varepsilon,p_j(0)-\varepsilon,\cdots,p_N(0)\right)$$

$$\varepsilon = 10^{-10}$$





4. Molecular Dynamics

4.2 Liouville Formulation

Liouville formulation

the dot above, $\dot{f}$,
implies time derivative

Let us consider a function that *f* which depends on the positions and momenta of the particles:

$$f\left(p^N, r^N\right)$$

We can "solve" how f depends on time:

$$\dot{f} = \left(\frac{\partial f}{\partial r}\right)\dot{r} + \left(\frac{\partial f}{\partial p}\right)\dot{p}$$

Define the Liouville operator:

$$iL \equiv \dot{r}\left(\frac{\partial}{\partial r}\right) + \dot{p}\left(\frac{\partial}{\partial p}\right)$$

the time dependence follows from:

$$\frac{df}{dt} = iLf$$

with solution:

beware: the solution is equally useless as the differential equation

$$f = e^{iLt} f\left(0\right)$$

In an ideal world it would be less useless:

$$iL \equiv \dot{r}\left(\frac{\partial}{\partial r}\right) + \dot{p}\left(\frac{\partial}{\partial p}\right)$$

Let us look at half the equation

$$iL_r \equiv \left(\frac{\partial}{\partial r}\right)\dot{r}$$

which has as solution:

$$f = e^{iL_r t} f(0)$$

Taylor expansion:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

$$e^{iL_r t} f(0) = \left[1 + iL_r t + \frac{1}{2}\left(iL_r t\right)^2 + \frac{1}{3!}\left(iL_r t\right)^3 + \ldots\right] f(0)$$

$$e^{iL_r t} f(0) = \left[1 + \dot{r}(0)t\left(\frac{\partial}{\partial r}\right) + \frac{1}{2}\left(\dot{r}(0)t\right)^2\left(\frac{\partial}{\partial r}\right)^2 + \ldots\right] f(0)$$

$$f\left(0 + \dot{r}(0)t\right) = f(0) + \dot{r}(0)t\left(\frac{\partial f(0)}{\partial r}\right) + \frac{1}{2}\left(\dot{r}(0)t\right)^2\left(\frac{\partial f(0)}{\partial r}\right)^2 + \cdots$$

the operator iL_r gives a shift of the positions

Hence: $\boxed{e^{iL_r t} f(0) = f\left(0 + \dot{r}(0)t\right)}$

The operation iL_r gives a shift of the positions

$$iL \equiv \dot{r}\left(\frac{\partial}{\partial r}\right) + \dot{p}\left(\frac{\partial}{\partial p}\right)$$

Similarly for the operator iL_p

$$iL_p \equiv \left(\frac{\partial}{\partial p}\right)\dot{p}$$

which has as solution:

$$f = e^{iL_p t} f(0)$$

Taylor expansion:

$$e^{iL_p t} f(0) = \left[1 + iL_p t + \frac{1}{2}\left(iL_p t\right)^2 + \frac{1}{3!}\left(iL_p t\right)^3 + \ldots\right] f(0)$$

$$e^{iL_p t} f(0) = \left[1 + \dot{p}(0) t \left(\frac{\partial}{\partial p}\right) + \frac{1}{2}\left(\dot{p}(0) t\right)^2 \left(\frac{\partial}{\partial p}\right)^2 + \ldots\right] f(0)$$

$$f\left(0 + \dot{p}(0) t\right) = f(0) + \dot{p}(0) t \left(\frac{\partial f(0)}{\partial p}\right) + \frac{1}{2}\left(\dot{p}(0) t\right)^2 \left(\frac{\partial f(0)}{\partial p}\right)^2 + \cdots$$

the operator iL_p gives a shift of the momenta

Hence:

$$\boxed{e^{iL_p t} f(0) = f\left(0 + \dot{p}(0) t\right)}$$

The operation iL_r gives a shift of the positions:

$$e^{iL_r t} f(0,0) = f\left(0, 0 + \dot{r}(0)t\right)$$

… and the operator iL_p a shift of the momenta:

$$e^{iL_p t} f(0,0) = f\left(0 + \dot{p}(0)t, 0\right)$$

This would have been useful if the operators would commute

$$e^{iLt} f(0,0) = e^{(iL_r + iL_p)t} f(0,0) \neq e^{iL_r t} e^{iL_p t} f(0,0)$$

Trotter expansion:

we have the non-commuting operators A and B:

then the following expansion holds:

$$e^{A+B} \neq e^A e^B$$

$$e^{A+B} = \lim_{P\to\infty} \left(e^{\frac{A}{2P}} e^{\frac{B}{P}} e^{\frac{A}{2P}} \right)^P$$

$$e^{iL_r t} f(0,0) = f(0, 0 + \dot{r}(0)t)$$

$$e^{iL_p t} f(0,0) = f(0 + \dot{p}(0)t, 0)$$

We can apply the Trotter expansion:

$$e^{A+B} = \lim_{P\to\infty}\left(e^{\frac{A}{2P}} e^{\frac{B}{P}} e^{\frac{A}{2P}}\right)^P$$

$$\Delta t = \frac{t}{P} \qquad \frac{iL_r t}{P} = iL_r \Delta t \qquad \frac{iL_p t}{2P} = iL_p \frac{\Delta t}{2}$$

These give as operations:

$$e^{iL_r \Delta t} f(p(t), r(t)) = f(p(t), r(t) + \dot{r}(t)\Delta t)$$

gives us a shift of the position:

$$r(t+\Delta t) \to r(t) + \dot{r}(t)\Delta t$$

$$e^{iL_p \Delta t/2} f(p(t), r(t)) = f\left(p(t) + \dot{p}(t)\frac{\Delta t}{2}, r(t)\right)$$

gives us a shift of the momenta:

$$p(t+\Delta t) \to p(t) + \dot{p}(t)\frac{\Delta t}{2}$$

$$iL_r \Delta t \qquad r(t+\Delta t) \to r(t) + \dot{r}(t)\Delta t$$

$$iL_p \frac{\Delta t}{2} \qquad p\left(t+\frac{\Delta t}{2}\right) \to p(t) + \dot{p}(t)\frac{\Delta t}{2}$$

We can apply the Trotter expansion to integrate M time steps: *t=M x Δt*

$$f(t) = e^{iLt} f(0) = \left(e^{iL_p \frac{\Delta t}{2}} e^{iL_r \Delta t} e^{iL_p \frac{\Delta t}{2}}\right)^M f(0)$$

These give as operations:

$$e^{iL_p \frac{\Delta t}{2}} \qquad p\left(\frac{\Delta t}{2}\right) \to p(0) + \dot{p}(0)\frac{\Delta t}{2}$$

$$e^{iL_r \Delta t} \qquad r(\Delta t) \to r(0) + \dot{r}\left(\frac{\Delta t}{2}\right)\Delta t$$

$$e^{iL_p \frac{\Delta t}{2}} \qquad p(\Delta t) \to p\left(\frac{\Delta t}{2}\right) + \dot{p}(\Delta t)\frac{\Delta t}{2}$$

which gives after one step

$$p(0) \to p(0) + \left[f(0) + f(\Delta t)\right]\frac{\Delta t}{2}$$

$$r(0) \to r(0) + \dot{r}\left(\frac{\Delta t}{2}\right)\Delta t = r(0) + v(0)\Delta t + f(0)\frac{\Delta t^2}{2m}$$

which gives after one step

$$r(0) \to r(0) + \dot{r}\left(\frac{\Delta t}{2}\right)\Delta t = r(0) + v(0)\Delta t + f(0)\frac{\Delta t^2}{2m}$$

$$p(0) \to p(0) + \left[f(0) + f(\Delta t)\right]\frac{\Delta t}{2}$$

Velocity Verlet algorithm

$$r(t+\Delta t) = r(t) + v(t)\Delta t + f(t)\frac{\Delta t^2}{2m}$$

$$v(t+\Delta t) = v(t) + \frac{\Delta t}{2m}\left[f(t+\Delta t) + f(t)\right]$$

Velocity Verlet algorithm:

$$e^{iL_p\frac{\Delta t}{2}}e^{iL_r\Delta t}e^{iL_p\frac{\Delta t}{2}}$$

$$iL_r\Delta t: \quad r(t+\Delta t)\rightarrow r(t)+v(t)\Delta t$$

$$iL_p\frac{\Delta t}{2}: \quad v\left(t+\frac{\Delta t}{2}\right)\rightarrow v(t)+f(t)\frac{\Delta t}{2}$$

```
Call force(fx)
Do while (t<tmax)
```

$$iL_p\frac{\Delta t}{2}: \quad v\left(t+\frac{\Delta t}{2}\right)\rightarrow v(t)+f(t)\frac{\Delta t}{2}$$

```
vx=vx+delt*fx/2
```

$$iL_r\Delta t: \quad r(t+\Delta t)\rightarrow r(t)+v(t)\Delta t$$

```
x=x+delt*vx
Call force(fx)
```

$$iL_p\frac{\Delta t}{2}: \quad v(t+\Delta t)\rightarrow v\left(t+\frac{\Delta t}{2}\right)+f(t+\Delta t)\frac{\Delta t}{2}$$

```
vx=vx+delt*fx/2
```

```
enddo
```

Liouville formulation

Velocity Verlet algorithm

$$r(t+\Delta t) = r(t) + v(t)\Delta t + f(t)\frac{\Delta t^2}{2m}$$

$$v(t+\Delta t) = v(t) + \frac{\Delta t}{2m}\left[f(t+\Delta t) + f(t)\right]$$

Transformations:

$$iL_p \Delta t/2: \quad r(t) \to r(t)$$

$$v(t) \to v(t) + f(t)\Delta t/2m$$

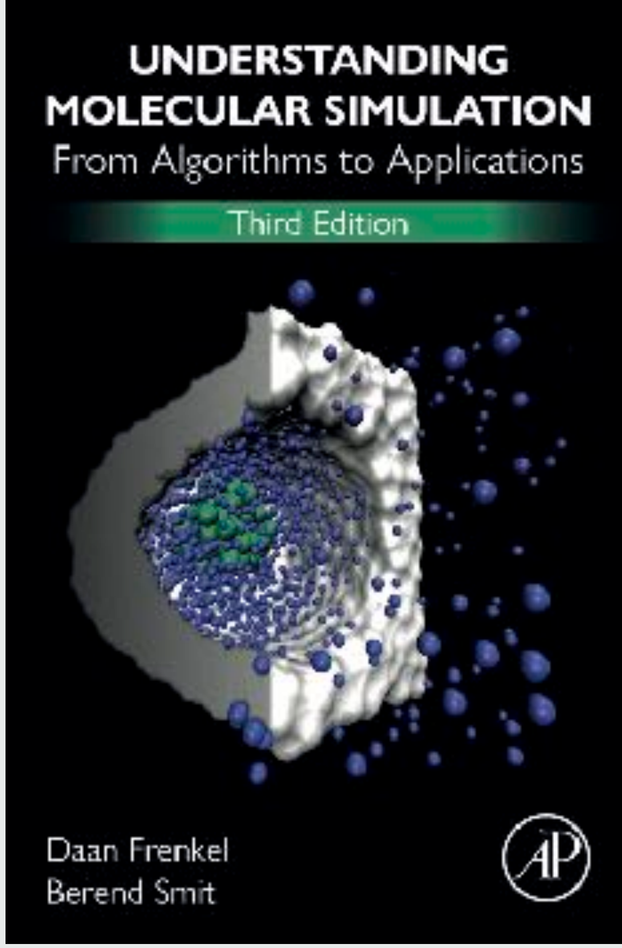
$$J_p = Det\begin{vmatrix} 1 & 0 \\ \left(\frac{\partial f}{\partial r}\right)\frac{\Delta t}{2m} & 1 \end{vmatrix} = 1$$

$$iL_r \Delta t: \quad r(t+\Delta t) \to r(t) + v(t)\Delta t$$

$$v(t) \to v(t)$$

$$J_r = Det\begin{vmatrix} 1 & \Delta t \\ 0 & 1 \end{vmatrix} = 1$$

Three subsequent coordinate transformations in either r or r of which the Jacobian is one: Area preserving

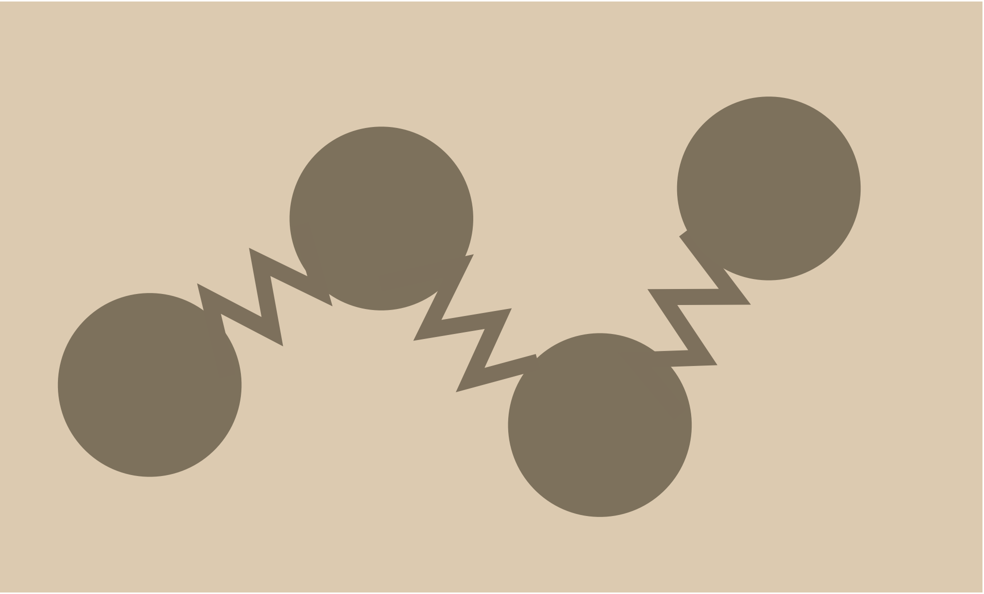


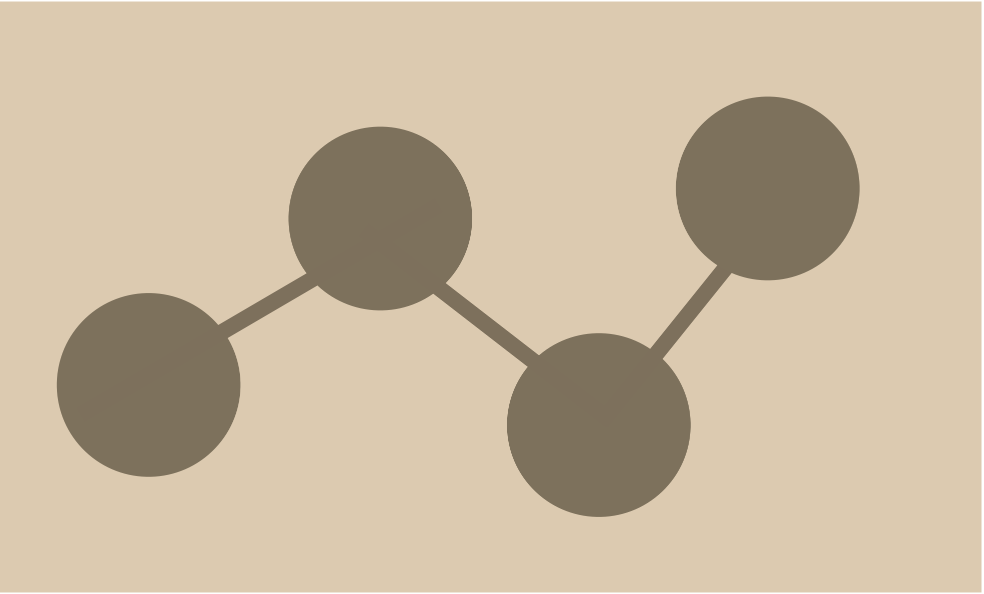
4. Molecular Dynamics

4.3 Multiple Time Steps

Multiple time steps

What to do with “stiff” potentials?





- Fixed bond-length: constraints (Shake)
- Very small time step

$$iL_r \Delta t: \quad r(t+\Delta t) \rightarrow r(t) + v(t)\Delta t$$

$$iL_p \frac{\Delta t}{2}: \quad v\left(t+\frac{\Delta t}{2}\right) \rightarrow v(t) + f(t)\frac{\Delta t}{2}$$

We can split the force is the stiff part and the more slowly changing rest of the forces:

$$f(t) = f_{Short}(t) + f_{Long}(t)$$

This allows us to split the Liouville operator:

$$iLt = iL_r t + iL_{pShort} t + iL_{pLong}$$

The conventional Trotter expansion:

$$iLt = \left[iL_{pLong}\, \Delta t/2 \left[iL_r + iL_{pShort} \right] \Delta t \; iL_{pLong}\, \Delta t/2 \right]^M$$

Now we can make another Trotter expansion: δt=Δt/m

$$\left[iL_r + iL_{pShort} \right] \Delta t = \left[iL_{pShort}\, \delta t/2 \; iL_r \delta t \; iL_{pShort}\, \delta t/2 \right]^m$$

The algorithm to solve the equations of motion

$$f(t) = f_{Short}(t) + f_{Long}(t)$$

$$iLt = \left[iL_{pLong}\, \Delta t/2 \left[iL_r + iL_{pShort} \right] \Delta t \; iL_{pLong}\, \Delta t/2 \right]^M$$

$$\left[iL_r + iL_{pShort} \right] \Delta t = \left[iL_{pShort}\, \delta t/2 \; iL_r \delta t \; iL_{pShort}\, \delta t/2 \right]^m$$

We now have 3 transformations:

$$iL_{pLong} \frac{\Delta t}{2}: \quad v\left(t + \frac{\Delta t}{2} \right) \rightarrow v(t) + f_{Long}(t) \frac{\Delta t}{2}$$

$$iL_{pShort} \frac{\delta t}{2}: \quad v\left(t + \frac{\delta t}{2} \right) \rightarrow v(t) + f_{Short}(t) \frac{\delta t}{2}$$

$$iL_r \delta t: \quad r(t + \delta t) \rightarrow r(t) + v(t) \delta t$$

The steps are first iL_{pLong} then m times iL_{pShort}/iL_r followed by iL_{pLong} again

$$iL_{pLong}\frac{\Delta t}{2}: \quad v\left(t+\frac{\Delta t}{2}\right) \rightarrow v(t)+f_{Long}(t)\frac{\Delta t}{2}$$

```
Call force(fx_long,f_short)

vx=vx+delt*fx_long/2
```

Algorithm 28 (Multiple-time-step MD)

function multi(fl,fs)	input:
	`fl`: long-range part of the force
	`fs`: short-range part of the force
`vx=vx+0.5*delt*fl`	velocity Verlet with time step $\Delta t/2$
for `1` $\leq$ `it` $\leq$ `n` **do**	loop for the short time steps
`vx=vx+0.5*(delt/n)*fs`	velocity Verlet with short timestep $\Delta t/n$
`x=x+(delt/n)2*vx`	
`fs =` **force_short**	short-range forces
`vx=vx+0.5*(delt/n)*fs`	
enddo	
`fl =` **force_long**	all long-ranged forces
`vx=vx+0.5*delt*fl`	velocity Verlet with time step $\Delta t/2$
end function	