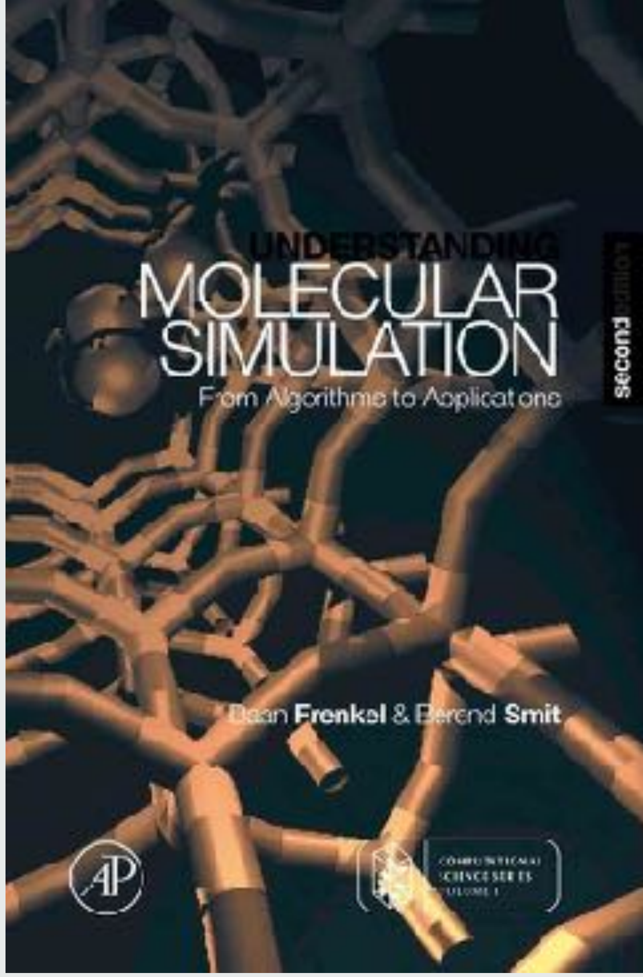
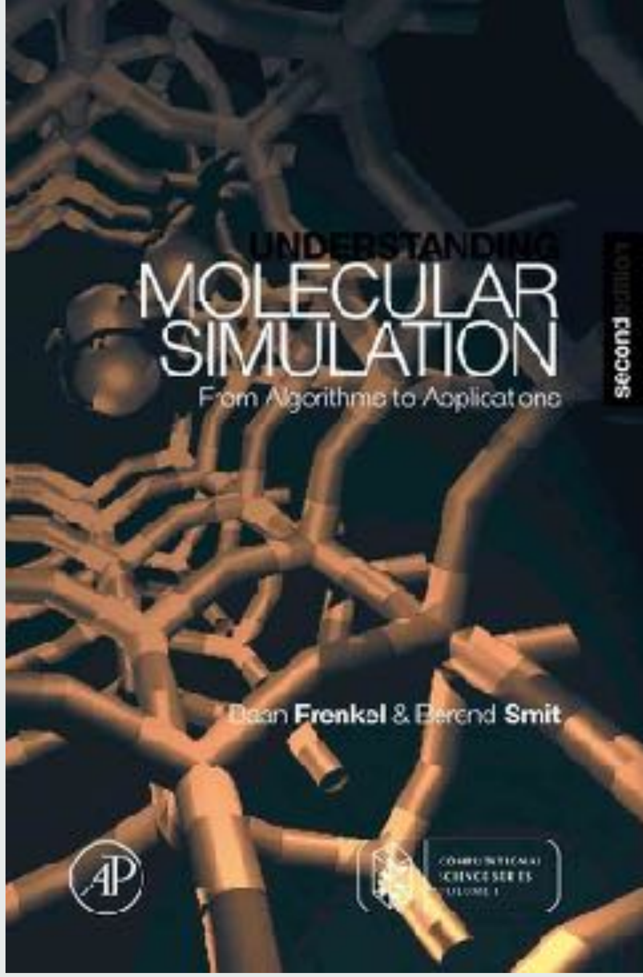
4. Molecular Dynamics


UNDERSTANDING
MOLECULAR
SIMULATION
From Algorithms to Applications
second edition
Daan Frenkel & Berend Smit
AP
COMPUTATIONAL SCIENCE SERIES VOLUME 1

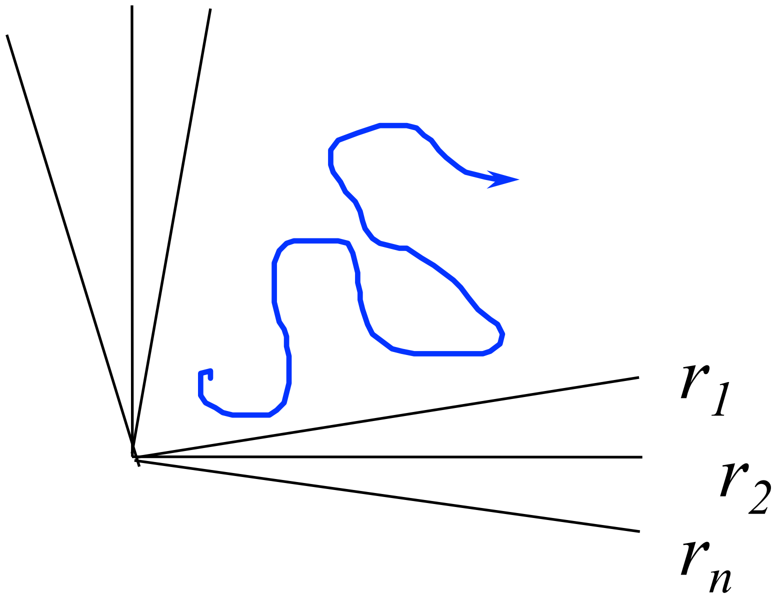


4. Molecular Dynamics

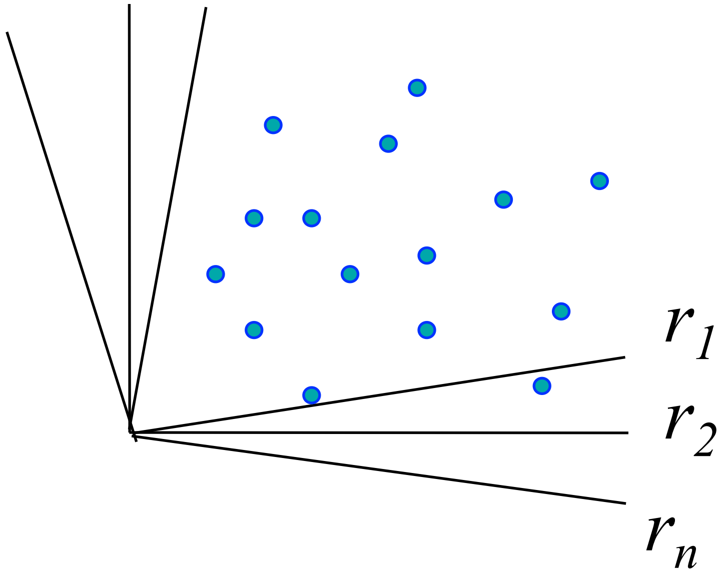
4.1 Basics

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”

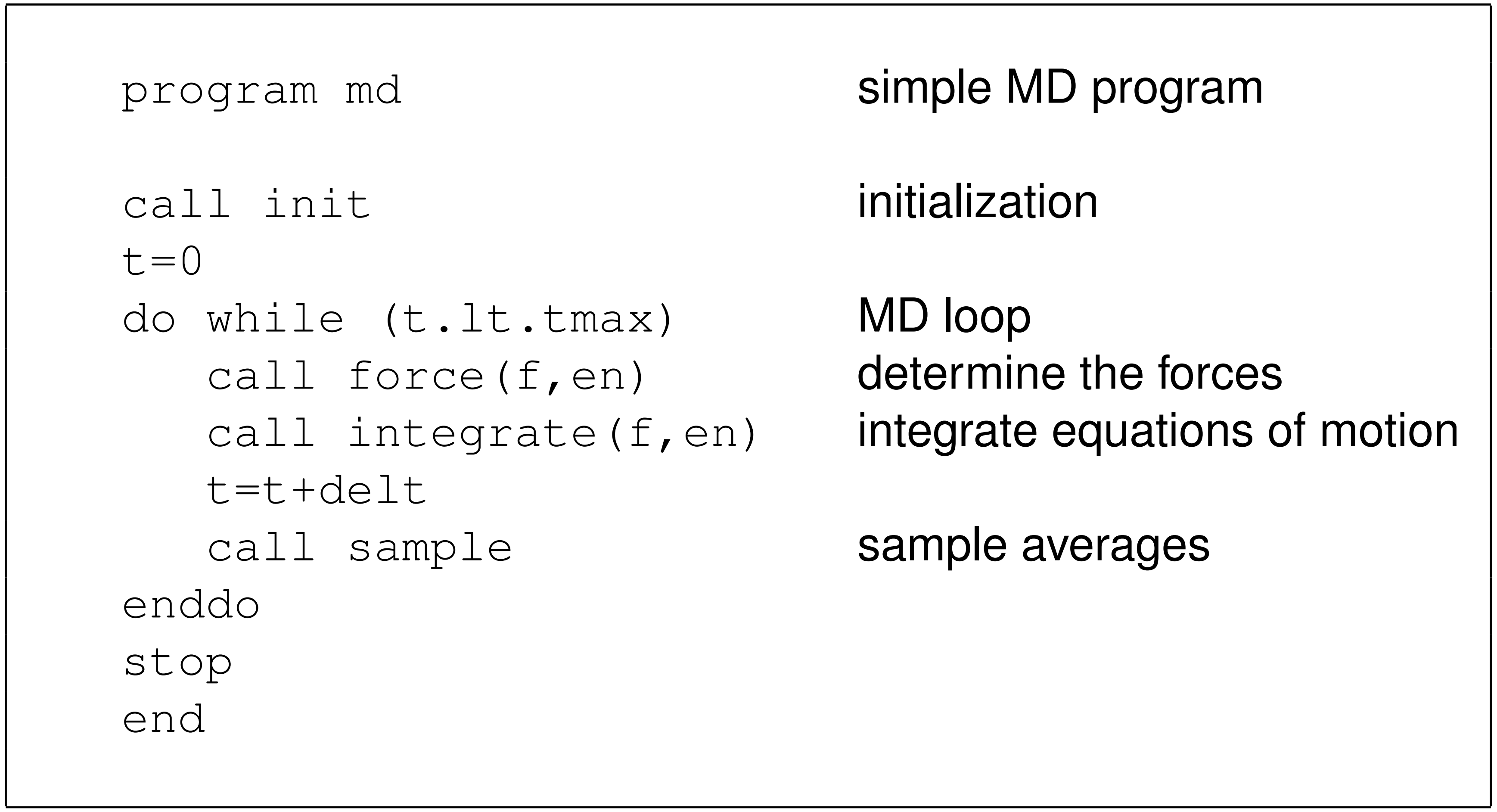
Theory:

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

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

Molecular Dynamics

Algorithm 3 (A Simple Molecular Dynamics Program)



```
program md                        simple MD program

call init                         initialization
t=0
do while (t.lt.tmax)              MD loop
   call force(f,en)               determine the forces
   call integrate(f,en)           integrate equations of motion
   t=t+delt
   call sample                    sample averages
enddo
stop
end
```

Molecular Dynamics

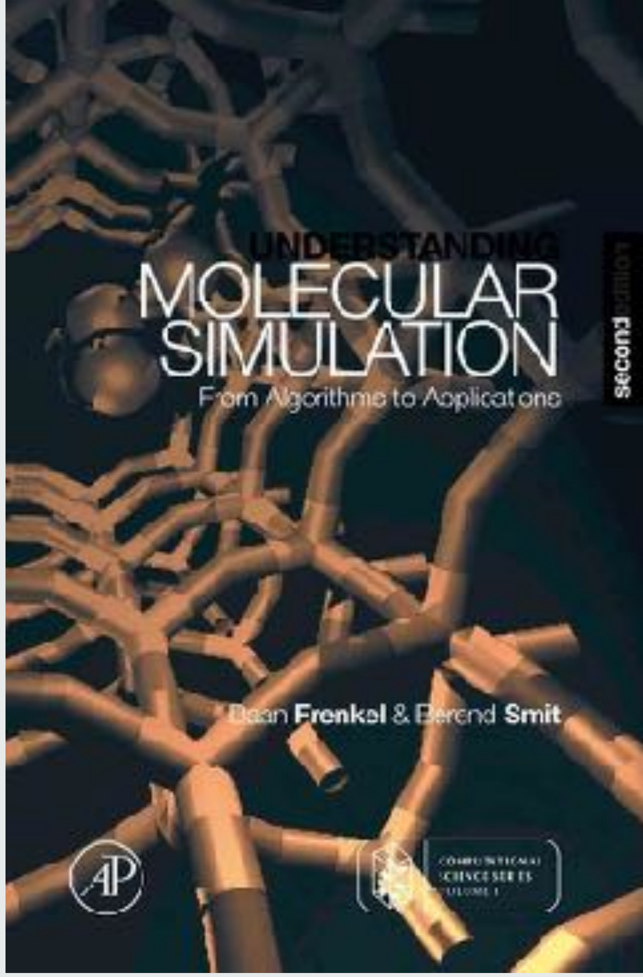
Initialization

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)

```
subroutine init              initialization of MD program
sumv=0
sumv2=0
do i=1,npart
  x(i)=lattice_pos(i)        place the particles on a lattice
  v(i)=(ranf()-0.5)          give random velocities
  sumv=sumv+v(i)             velocity center of mass
  sumv2=sumv2+v(i)**2        kinetic energy
enddo
sumv=sumv/npart              velocity center of mass
sumv2=sumv2/npart            mean-squared velocity
fs=sqrt(3*temp/sumv2)        scale factor of the velocities
do i=1,npart                 set desired kinetic energy and set
  v(i)=(v(i)-sumv)*fs        velocity center of mass to zero
  xm(i)=x(i)-v(i)*dt         position previous time step
enddo
return
end
```

Molecular Dynamics

Initialization

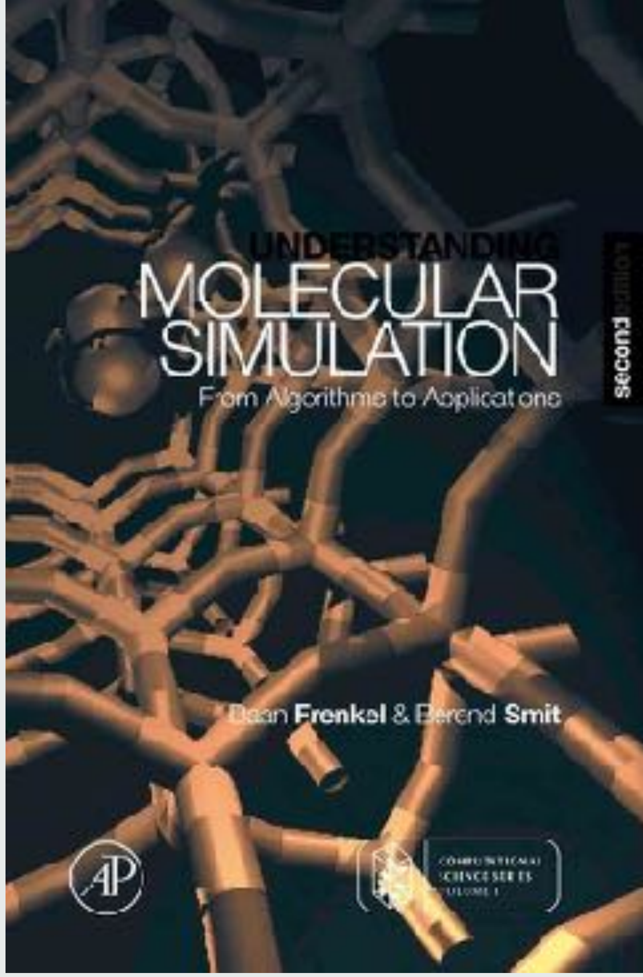
- Total momentum should be zero (no external forces)
- Temperature rescaling to desired temperature
- Particles start on a lattice

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

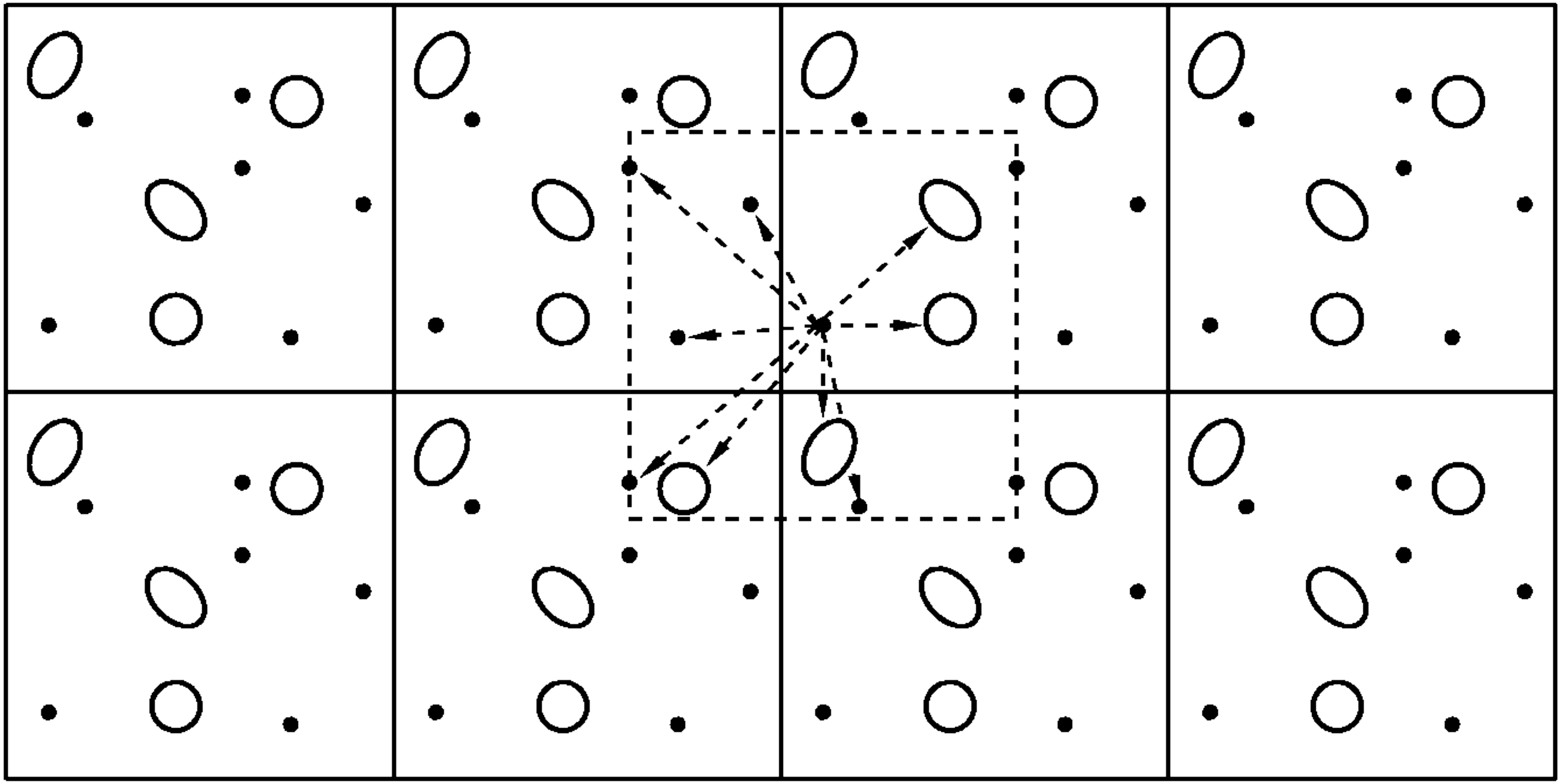
Integrating the equations of motion

- Velocity Verlet
- Kinetic energy

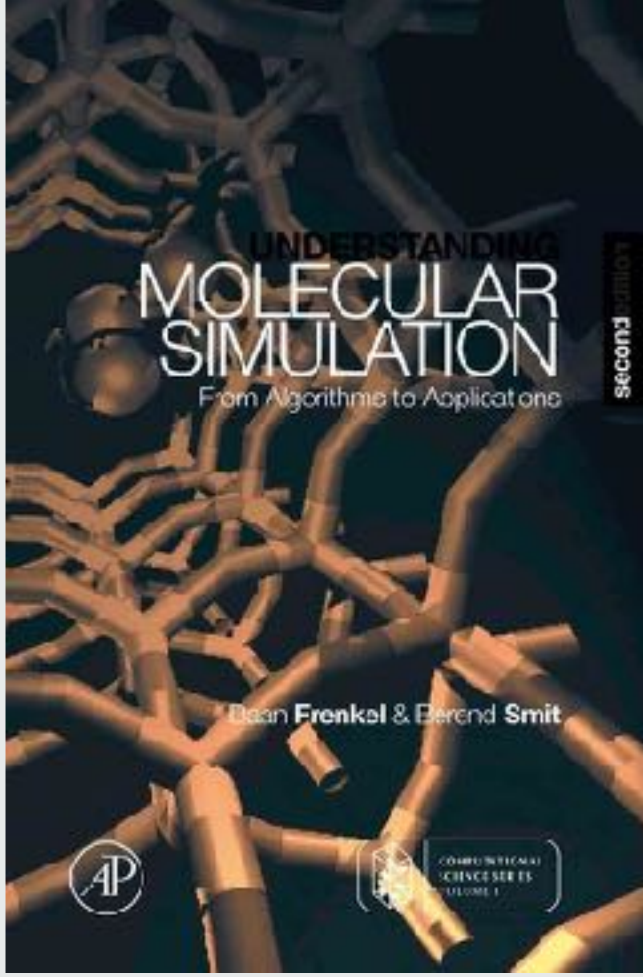
Algorithm 5 (Calculation of the Forces)

```
subroutine force(f,en)                  determine the force
en=0                                    and energy
do i=1,npart
   f(i)=0                               set forces to zero
enddo
do i=1,npart-1
  do j=i+1,npart                        loop over all pairs
     xr=x(i)-x(j)
     xr=xr-box*nint(xr/box)             periodic boundary conditions
     r2=xr**2
     if (r2.lt.rc2) then                test cutoff
       r2i=1/r2
       r6i=r2i**3
       ff=48*r2i*r6i*(r6i-0.5)          Lennard-Jones potential
       f(i)=f(i)+ff*xr                  update force
       f(j)=f(j)-ff*xr
       en=en+4*r6i*(r6i-1)-ecut         update energy
     endif
  enddo
enddo
return
end
```

Periodic boundary conditions



Understanding Molecular Simulation



4. Molecular Dynamics

4.1.2 Basics: Force Calculation - The Lennard Jones potential

The Lennard-Jones potentialS

- The Lennard-Jones potential

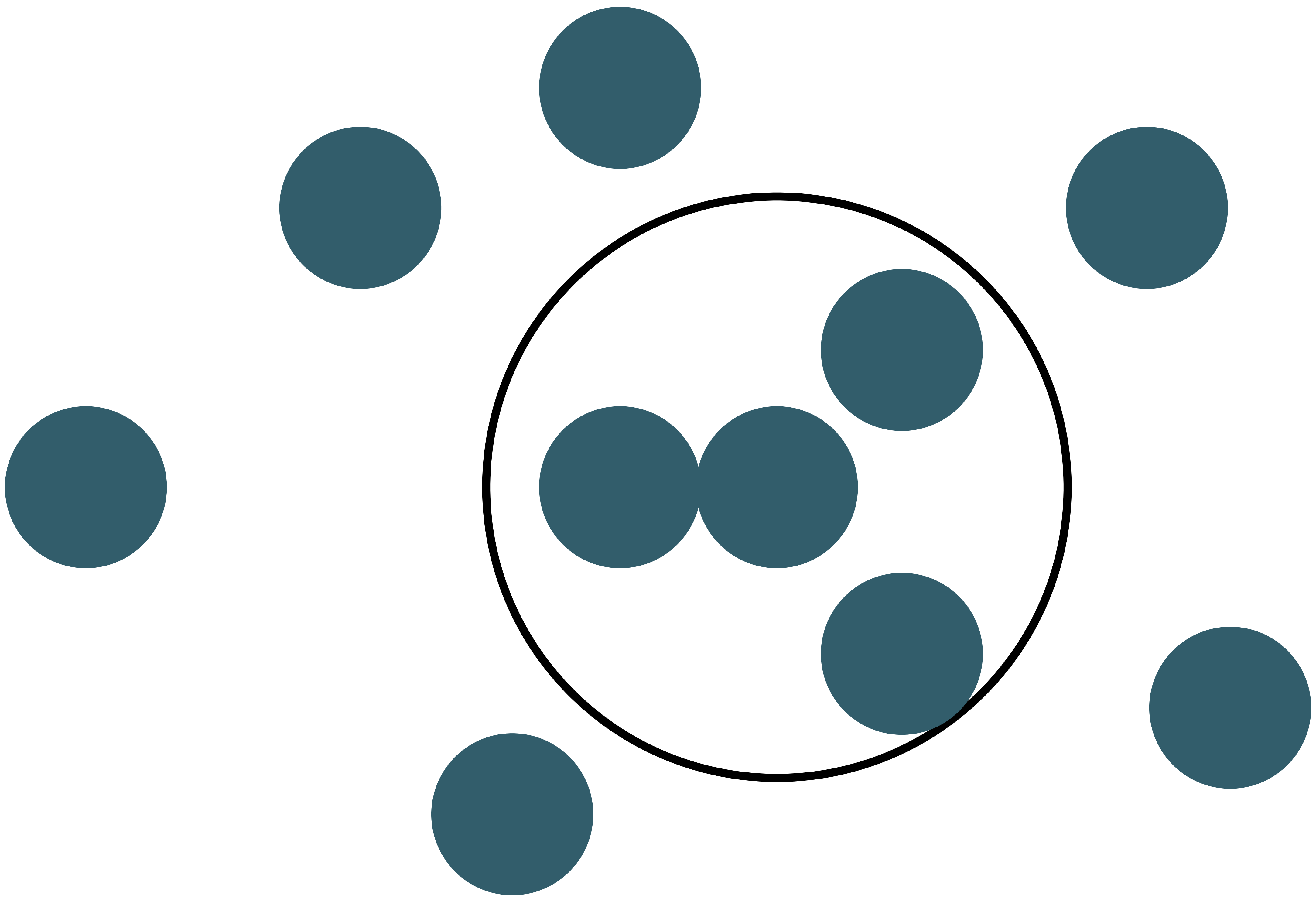
$$U^{LJ}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

- The truncated Lennard-Jones potential

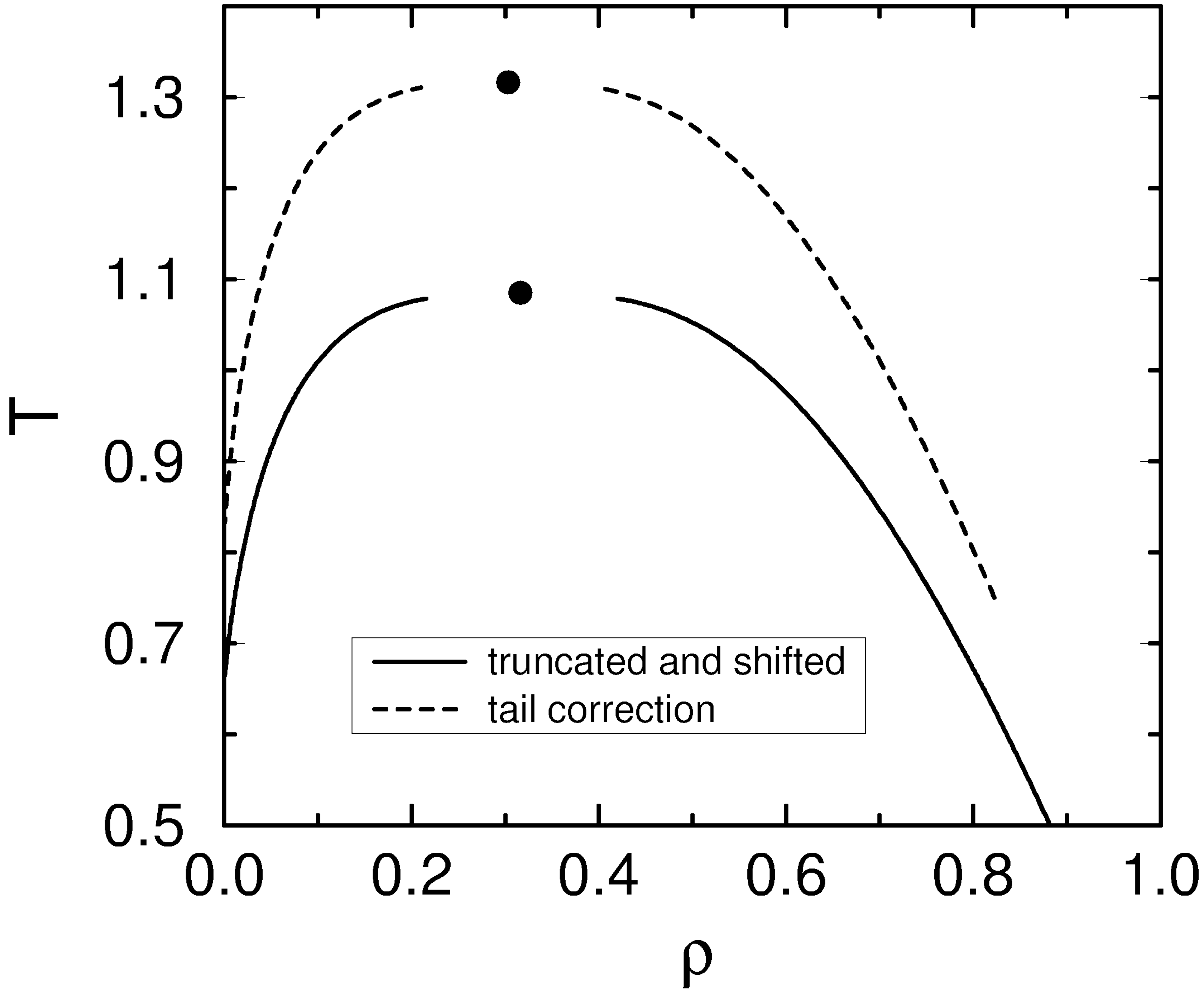
$$U^{LJ}_{TR}(r) = \begin{cases} U^{LJ}(r) & r \leq r_c \\ 0 & r > r_c \end{cases}$$

- The truncated and shifted Lennard-Jones potential

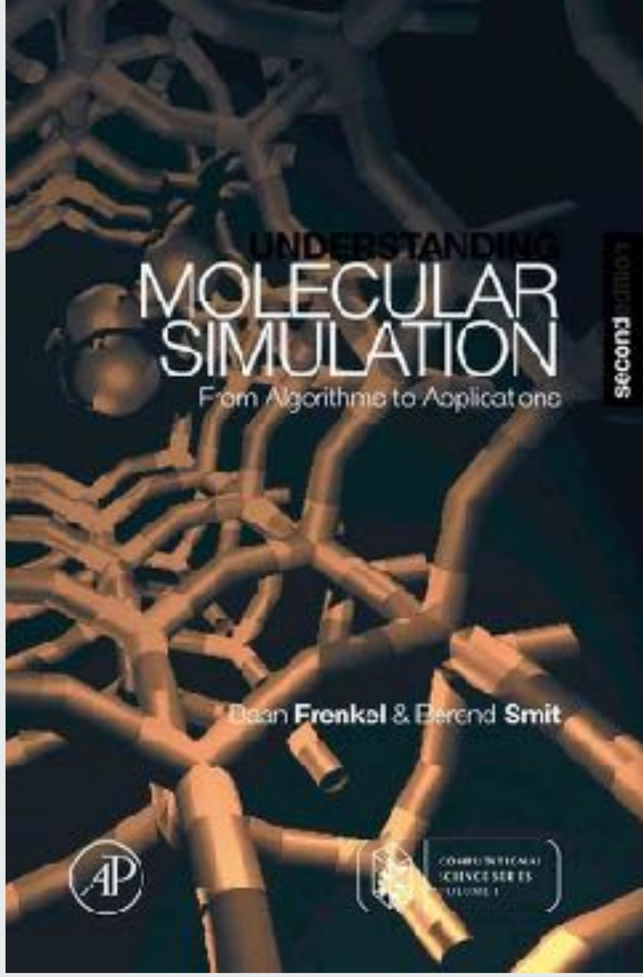
$$U^{LJ}_{TR-SH}(r) = \begin{cases} U^{LJ}(r) - U^{LJ}(r_c) & r \leq r_c \\ 0 & r > r_c \end{cases}$$



The Lennard-Jones potentials



Understanding Molecular Simulation

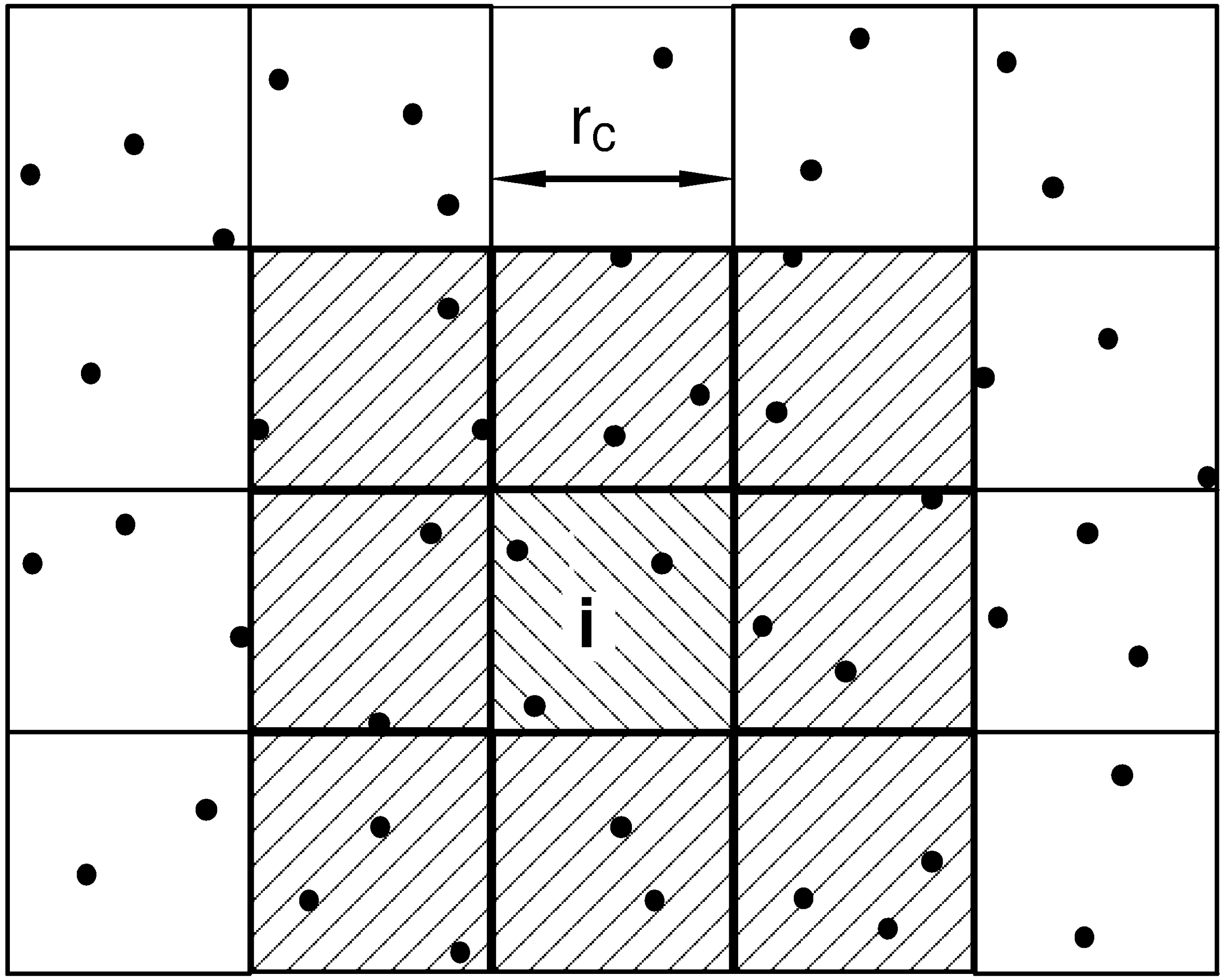


4. Molecular Dynamics

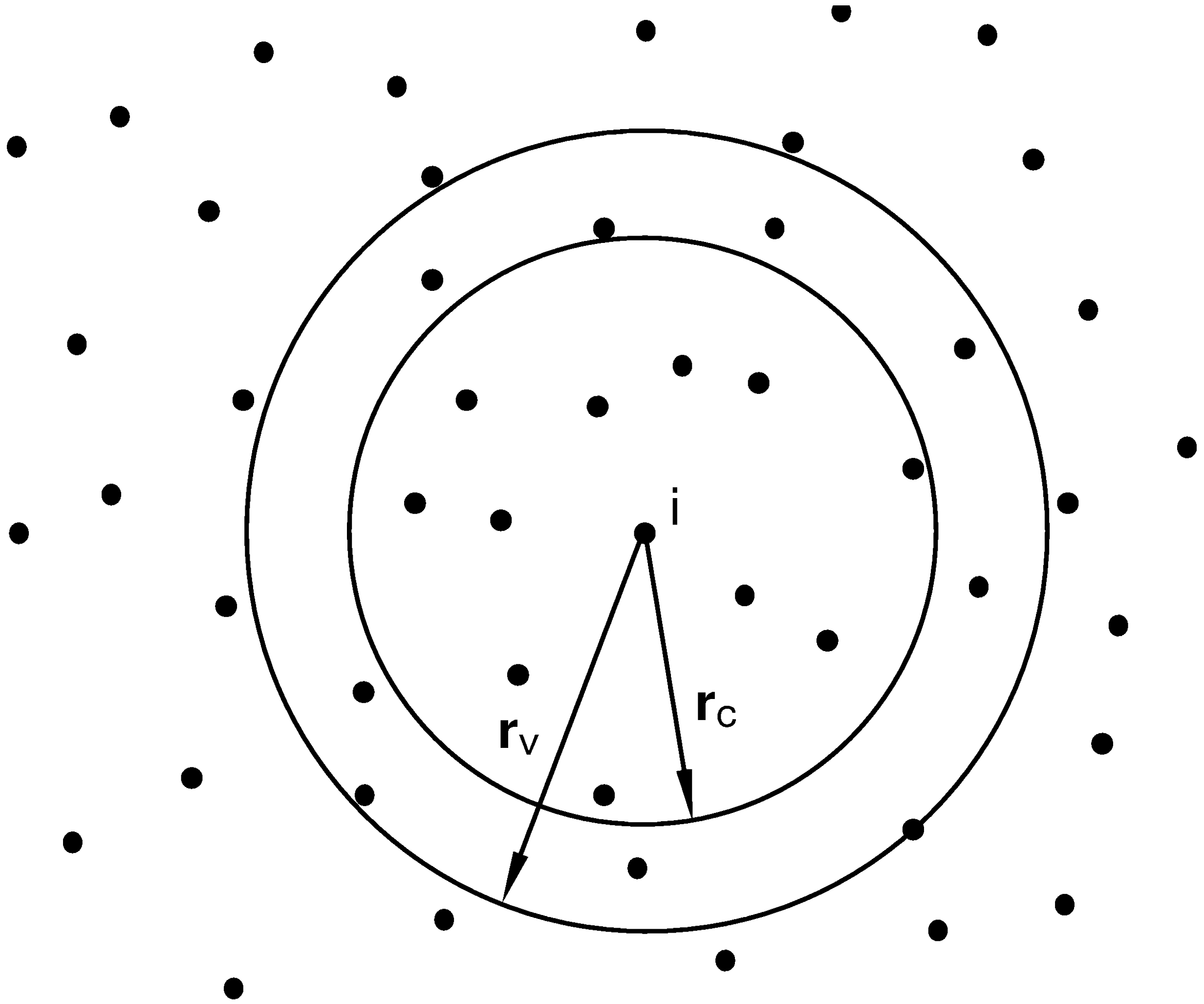
4.1.2 Basics: Force Calculation - saving CPU time

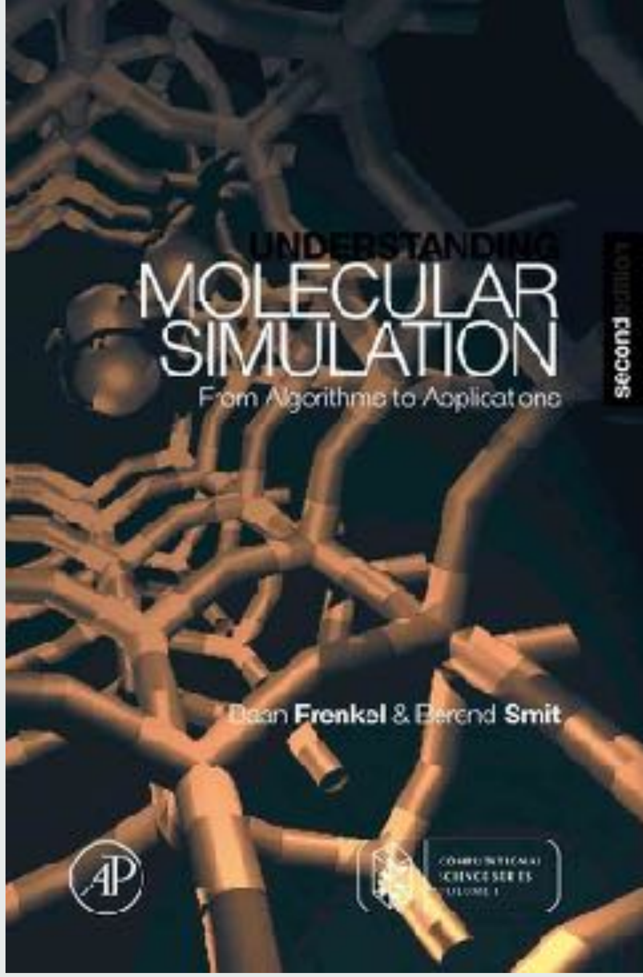
Saving CPU-time

Cell list



Verlet-list





4. Molecular Dynamics

4.1.3 Basics: Equations of Motion

Algorithm 6 (Integrating the Equations of Motion)

```
subroutine integrate(f,en)          integrate equations of motion
sumv=0
sumv2=0
do i=1,npart                        MD loop
   xx=2*x(i)-xm(i)+delt**2*f(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 positions previous time
   x(i)=xx                          update positions current time
enddo
temp=sumv2/(3*npart)                instantaneous temperature
etot=(en+0.5*sumv2)/npart           total energy per particle
return
end
```

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:

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

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

When we add the two:

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

Verlet algorithm

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

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

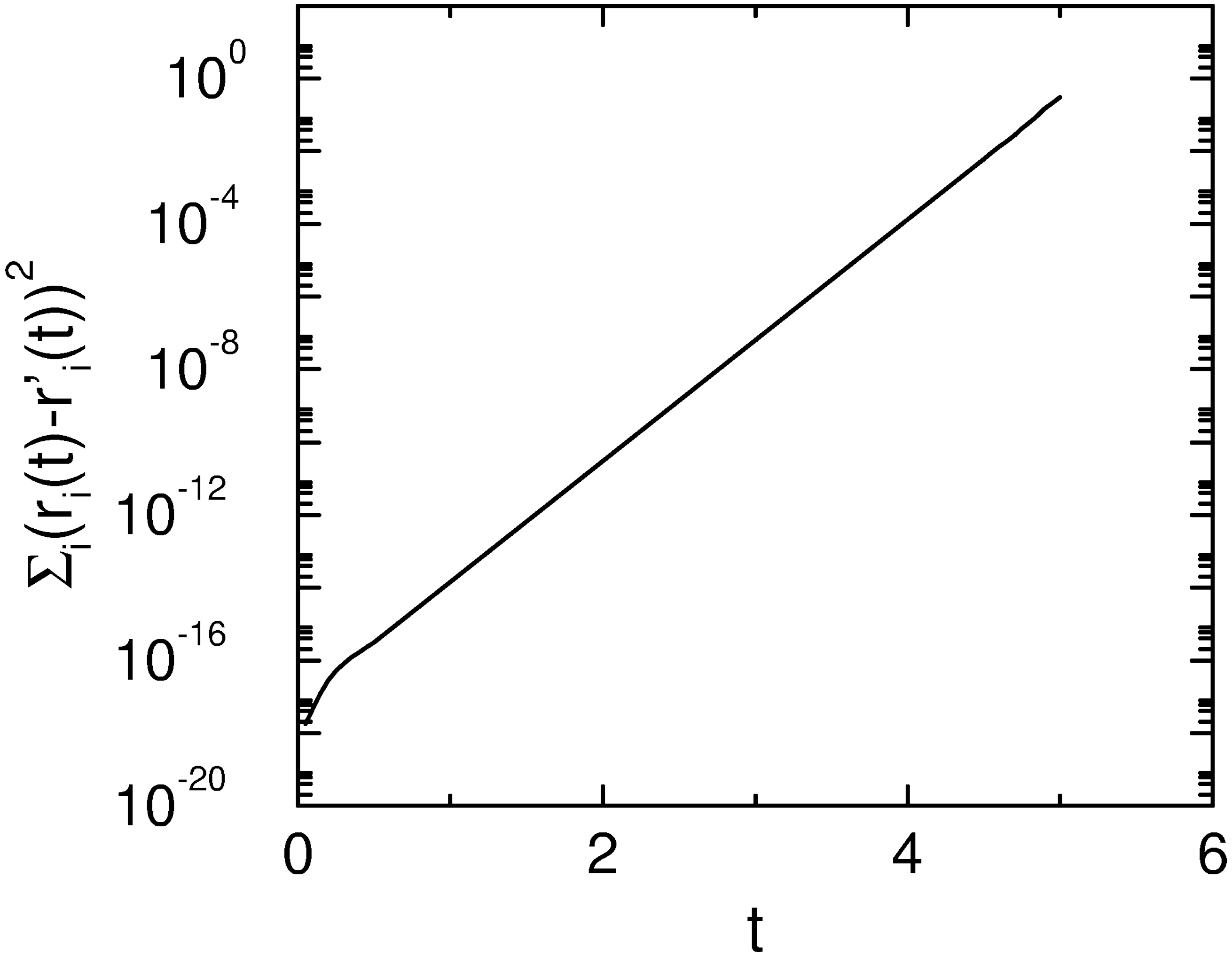
MD: reference trajectory with initial condition:

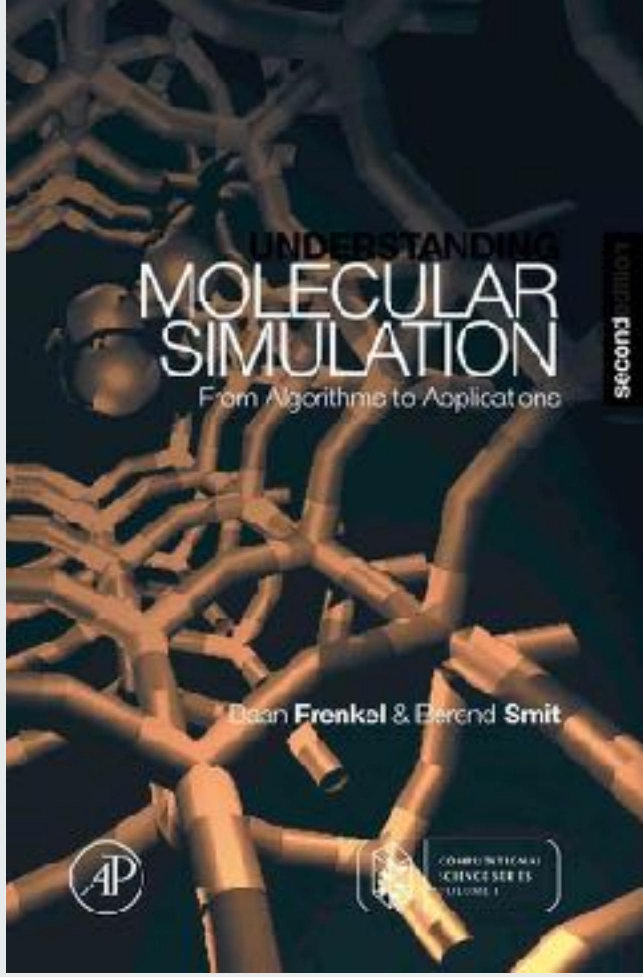
$$\left(r_1\left(0\right),\cdots,r_N\left(0\right),p_1\left(0\right),\cdots,p_N\left(0\right)\right)$$

MD: compare:

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

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





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:

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

beware: the solution is
equally useless as the
differential equation

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:

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

Taylor expansion:

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

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

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

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

the operator iLr gives a shift of the positions

Hence:

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

The operation *iL* 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:

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

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

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

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:

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

… and the operator *iL_p* a shift of the momenta:

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

This would have been useful if the operators would commute

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

Trotter expansion:

we have the non-commuting operators A and B:

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

then the following expansion holds:

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

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

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

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:

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

gives us a shift of the position:

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

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

gives us a shift of the momenta:

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

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

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

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:

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

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

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

which gives after one step

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

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

which gives after one step

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

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

Velocity Verlet algorithm

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

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

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

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

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

Transformations:

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

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

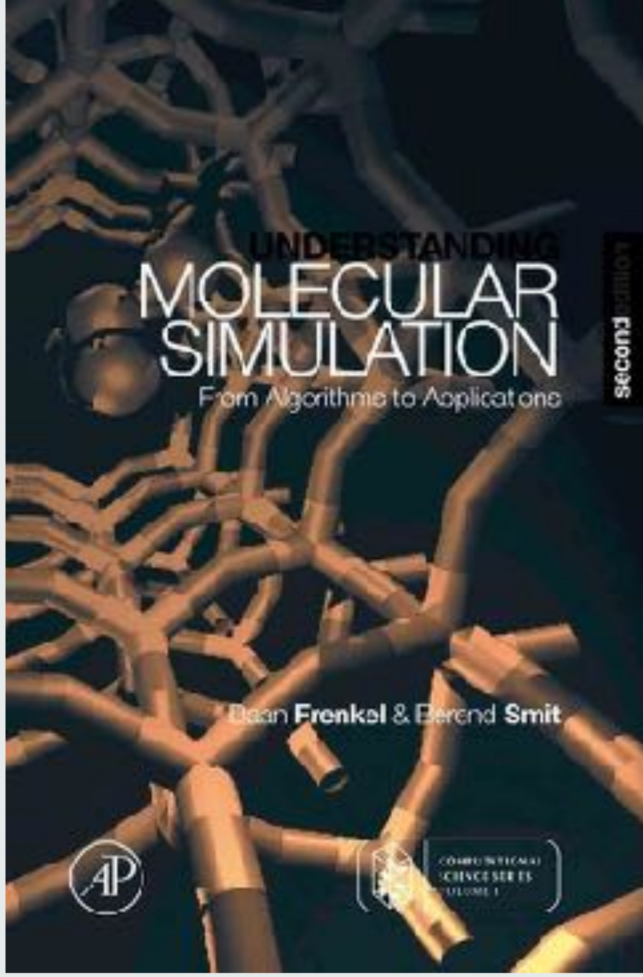
$$J_p=Det\begin{vmatrix} 1 & 0 \\ \left(\frac{\partial f}{\partial r}\right)\frac{\Delta t}{2m} & 1 \end{vmatrix}=1$$

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

$$v\left(t\right)\to v\left(t\right)$$

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

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

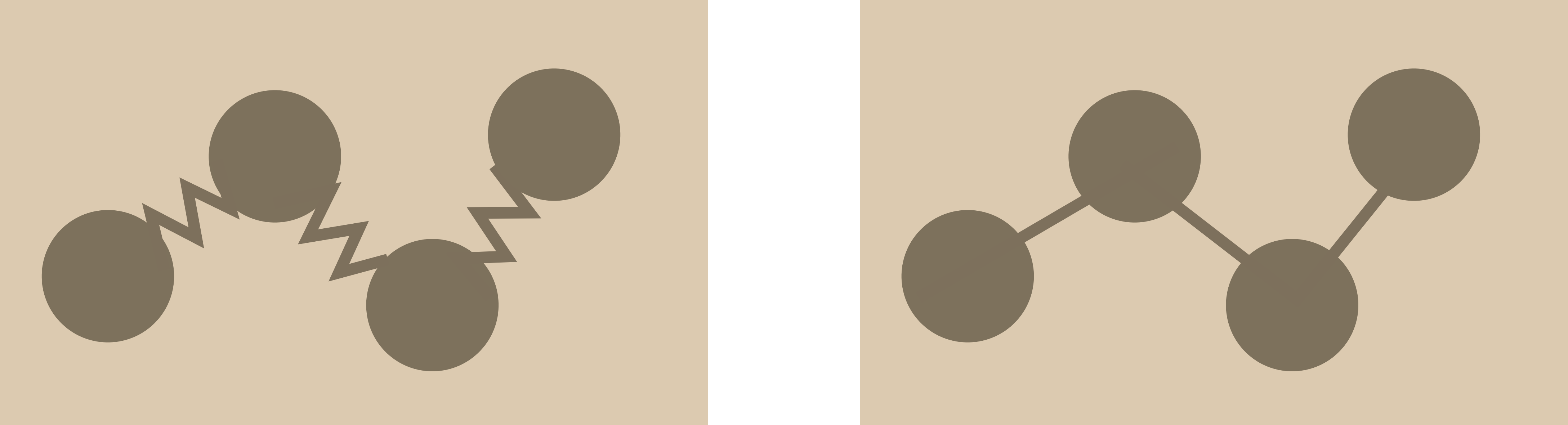


4. Molecular Dynamics

4.3 Multiple Time Steps

Multiple time steps

What to do with “stiff” potentials?



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

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

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

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

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

```
Call force(fx_long,f_short)

vx=vx+delt*fx_long/2

Do ddt=1,n
```

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

```
    vx=vx+ddelt*fx_short/2
```

$$iL_{r}\delta t: \quad r\left(t+\delta t\right) \rightarrow r\left(t\right)+v\left(t\right)\delta t$$

```
    x=x+ddelt*vx
    Call force_short(fx_short)
```

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

```
    vx=vx+ddelt*fx_short/2
enddo
```

Algorithm 29 (Multiple Time Step)

```
 subroutine                                Multiple time step, f_long is
+    multi(f_long,f_short)                 the long-range part and f_short
                                           the short-range part of the force
 vx=vx+0.5*delt*f_long                     velocity Verlet with time step Δt
 do it=1,n                                 loop for the small time step
    vx=vx+0.5*(delt/n)*f_short             velocity Verlet with timestep Δt/n
    x=x+(delt/n)2*vx
    call force_short(f_short)              short-range forces
    vx=vx+0.5*(delt/n)*f_short
 enddo
 call force_all(f_long,f_short)            all forces
 vx=vx+0.5*delt*f_long
 return
 end
```