

Molecular Dynamics

Aiichiro Nakano

Collaboratory for Advanced Computing & Simulations

Department of Computer Science

Department of Physics & Astronomy

Department of Chemical Engineering & Materials Science

Department of Quantitative & Computational Biology

University of Southern California

Email: anakano@usc.edu

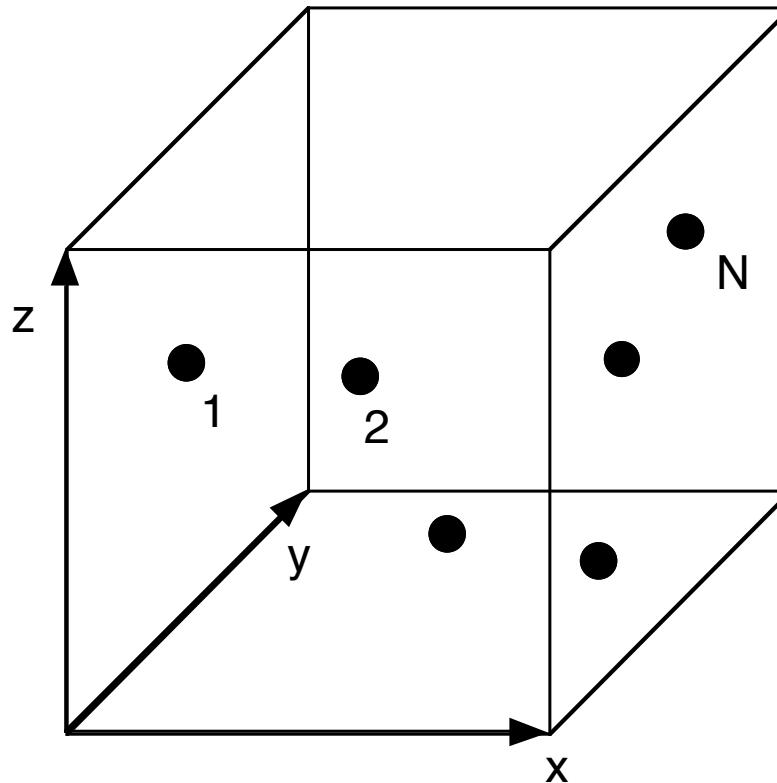
Goals:

1. Math: ordinary differential equations
2. Practical MD simulation



System: A Set of Point Atoms

$$\{\vec{r}_i = (x_i, y_i, z_i) \mid x_i, y_i, z_i \in \mathfrak{N}, i = 0, \dots, N - 1\}$$



In program md.c:

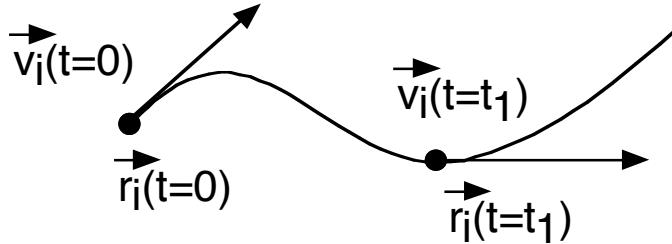
int nAtom: N , Number of atoms.

NMAX: Maximum number of atoms.

double r[NMAX][3]: $r[i][0|1|2] = x_i|y_i|z_i$.

Trajectory

- Trace of atom positions



$t \in \mathfrak{N} \mapsto \vec{r}_i(t) \in \mathfrak{N}^3$
Mapping

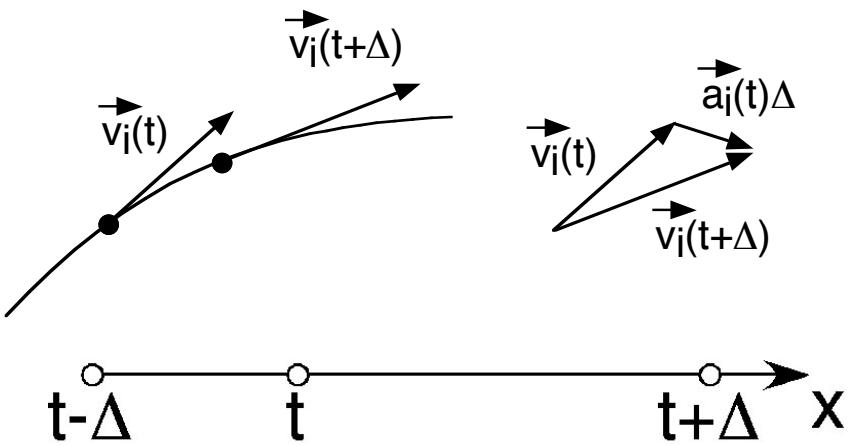
- Velocity $\vec{v}_i(t) = \dot{\vec{r}}_i(t) = \frac{d\vec{r}_i}{dt} \equiv \lim_{\Delta \rightarrow 0} \frac{\vec{r}_i(t + \Delta) - \vec{r}_i(t)}{\Delta}$ 2-point difference

double rv[NMAX][3]: $rv[i][0 | 1 | 2] = v_{ix} | v_{iy} | v_{iz}$

- Acceleration $\vec{a}_i(t) = \ddot{\vec{r}}_i(t) = \frac{d^2\vec{r}_i}{dt^2} = \frac{d\vec{v}_i}{dt} \equiv \lim_{\Delta \rightarrow 0} \frac{\vec{v}_i(t + \Delta) - \vec{v}_i(t)}{\Delta}$

double ra[NMAX][3]: $ra[i][0 | 1 | 2] = a_{ix} | a_{iy} | a_{iz}$

It's vector arithmetic



$$\begin{aligned}
 \vec{a}_i(t) &= \lim_{\Delta \rightarrow 0} \frac{\vec{v}_i(t + \Delta/2) - \vec{v}_i(t - \Delta/2)}{\Delta} \\
 &= \lim_{\Delta \rightarrow 0} \frac{\frac{\vec{r}_i(t + \Delta) - \vec{r}_i(t)}{\Delta} - \frac{\vec{r}_i(t) - \vec{r}_i(t - \Delta)}{\Delta}}{\Delta} \\
 &= \lim_{\Delta \rightarrow 0} \frac{\vec{r}_i(t + \Delta) - 2\vec{r}_i(t) + \vec{r}_i(t - \Delta)}{\Delta^2} \quad \text{3-point difference}
 \end{aligned}$$

Newton's Equation of Motion

Trajectory is determined by Newton's 2nd law:

$$\text{mass} \longrightarrow m\ddot{\vec{r}}_i(t) = \vec{F}_i(t) \longleftarrow \text{force}$$

- **Initial value problem:** Given initial particle positions & velocities $\{(\vec{r}_i(0), \vec{v}_i(0)) | i = 0, \dots, N-1\}$, obtain those at later times $\{(\vec{r}_i(t), \vec{v}_i(t)) | i = 0, \dots, N-1; t > 0\}$ by numerically integrating Newton's equation of motion
- **Force from potential energy:** $\vec{F}_k = -\frac{\partial}{\partial \vec{r}_k} V(\vec{r}^N) = -\left(\frac{\partial V}{\partial x_k}, \frac{\partial V}{\partial y_k}, \frac{\partial V}{\partial z_k}\right)$

where the partial derivative is

$$\frac{\partial V}{\partial x_k} = \lim_{h \rightarrow 0} \frac{V(x_0, y_0, z_0, \dots, \cancel{x_k + h}, y_k, z_k, \dots, x_{N-1}, y_{N-1}, z_{N-1}) - V(x_0, y_0, z_0, \dots, \cancel{x_k}, y_k, z_k, \dots, x_{N-1}, y_{N-1}, z_{N-1})}{h}$$

“for all pairs of atoms”

- **Pair potential:** $V(\vec{r}^N) = \sum_{i < j} u(r_{ij}) = \sum_{i=0}^{N-2} \sum_{j=i+1}^{N-1} u(|\vec{r}_{ij}|)$

$$\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$$

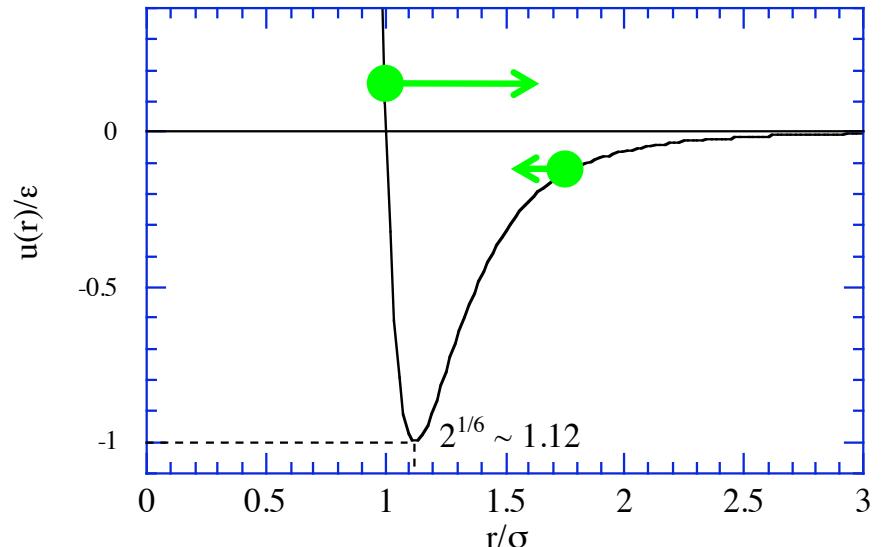
$i \setminus j$	0	1	2	3
0	X			
1		X		
2				X
3				

Lennard-Jones Potential

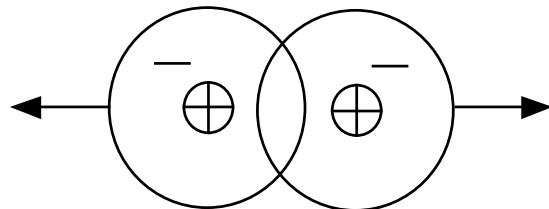
$$V(\vec{r}^N) = \sum_{i < j} u(r_{ij}) = \sum_{i=0}^{N-2} \sum_{j=i+1}^{N-1} u(|\vec{r}_{ij}|)$$

where $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ and

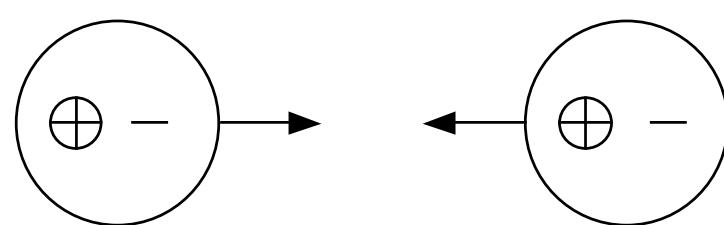
$$u(r) = 4\epsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right]$$



Short-range repulsion by Pauli exclusion between electrons



Long-range attraction by polarization



Johannes D. van der Waals
Nobel Physics Prize (1910)



Wolfgang Pauli
Nobel Physics
Prize (1945)



John E. Lennard-Jones
PPS 43, 461 (1931)

Normalization

For Argon atoms:

$$> m = 6.6 \times 10^{-23} \text{ gram}$$

$$> \varepsilon = 1.66 \times 10^{-14} \text{ erg} \quad (\text{erg} = \text{gram} \cdot \text{cm}^2/\text{second}^2)$$

$$> \sigma = 3.4 \times 10^{-8} \text{ cm}$$

Define length, energy & time units as

$$\begin{cases} \vec{r}_i = \vec{r}'_i \sigma = 3.4 \times 10^{-8} [\text{cm}] \times \vec{r}'_i \\ V = V' \varepsilon = 1.66 \times 10^{-14} [\text{erg}] \times V' \\ t = \sigma \sqrt{m / \varepsilon} t' = 2.2 \times 10^{-12} [\text{sec}] \times t' \end{cases}$$

$$\begin{aligned} 1.66 \times 10^{-14} \text{ erg} \\ = 3.97 \times 10^{-25} \text{ kcal} \\ = 7.35 \times 10^{-28} \text{ Big Mac} \end{aligned}$$

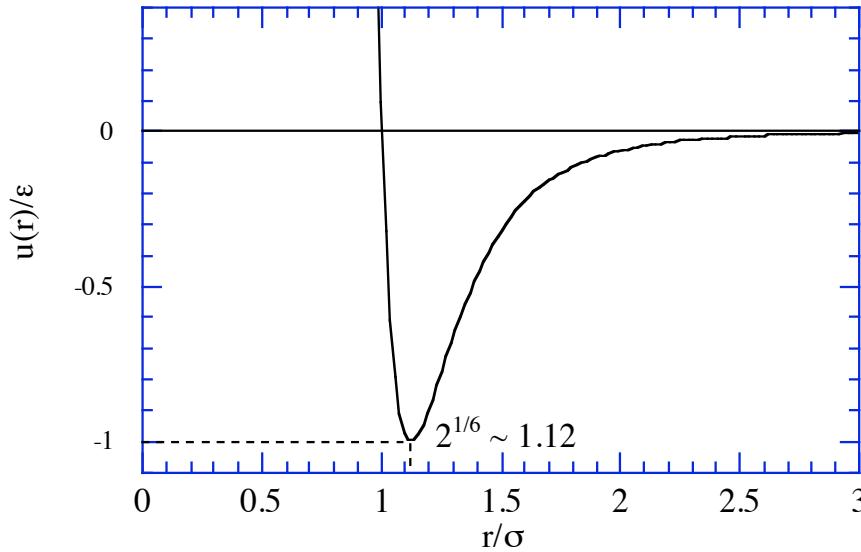


The equation of motion in these units:

$$\begin{aligned} \frac{d^2 \vec{r}_i}{dt^2} &= -\frac{\partial V}{\partial \vec{r}_i} = \vec{a}_i \\ V(\vec{r}^N) &= \sum_{i < j} u(r_{ij}) \\ u(r) &= 4 \left(\frac{1}{r^{12}} - \frac{1}{r^6} \right) \end{aligned}$$

540 kcal

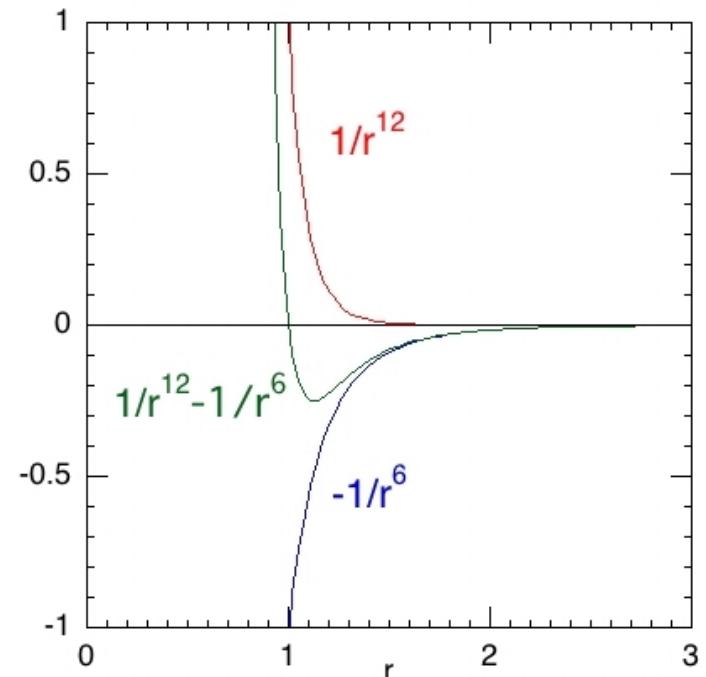
Repulsion vs. Attraction



$$u(r) = 4 \left(\frac{1}{r^{12}} - \frac{1}{r^6} \right)$$

$$\frac{du}{dr} = 4 \left(-\frac{12}{r^{13}} + \frac{6}{r^7} \right) = -\frac{48}{r^7} \left(\frac{1}{r^6} - \frac{1}{2} \right)$$

r	0	...	$2^{1/6}$...	∞
du/dr		-	0	+	
u	∞	\searrow	-1	\nearrow	0



Father of Molecular Dynamics

PHYSICAL REVIEW

VOLUME 136, NUMBER 2A

19 OCTOBER 1964

Correlations in the Motion of Atoms in Liquid Argon*

A. RAHMAN

Argonne National Laboratory, Argonne, Illinois

(Received 6 May 1964)

A system of 864 particles interacting with a Lennard-Jones potential and obeying classical equations of motion has been studied on a digital computer (CDC 3600) to simulate molecular dynamics in liquid argon at 94.4°K and a density of 1.374 g cm^{-3} .

Aneesur Rahman—Father of molecular dynamics

Argonne physicist Aneesur Rahman, known worldwide as the “father of molecular dynamics,” pioneered the application of computer science to physical systems.

In 1960, Rahman successfully modeled the behavior of a cluster of 864 argon atoms on a computer that could perform only 150,000 calculations per second.

While Argonne's new IBM Blue Gene ® /P supercomputer runs nearly 3 million times faster than Rahman's CDC 3600, today's scientists still base the code for their models on Rahman's algorithms.

Since 1993, the [American Physical Society](#) has annually awarded the Aneesur Rahman Prize for outstanding achievement in computational physics research.



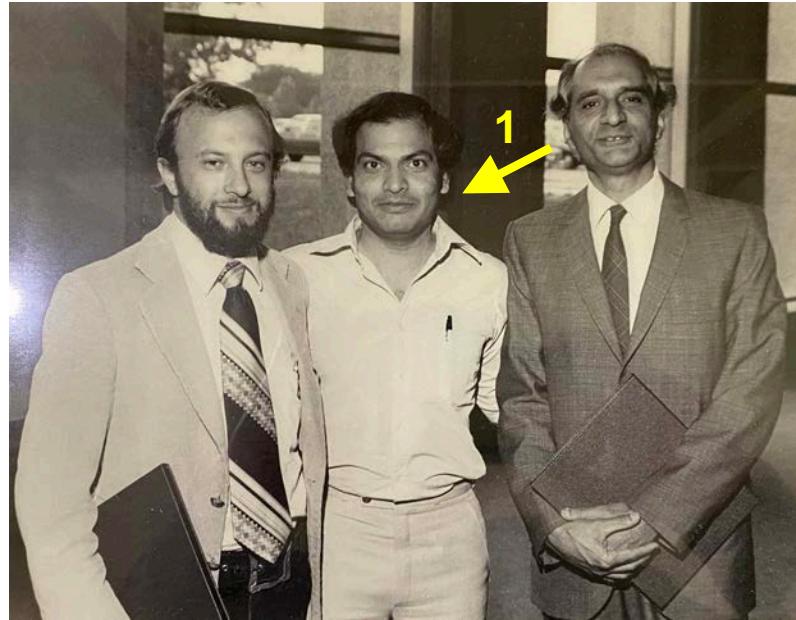
**Anees Rahman
(1927-1987)**

See the [Nobel lecture](#) by Michael Levitt

Rahman Number?



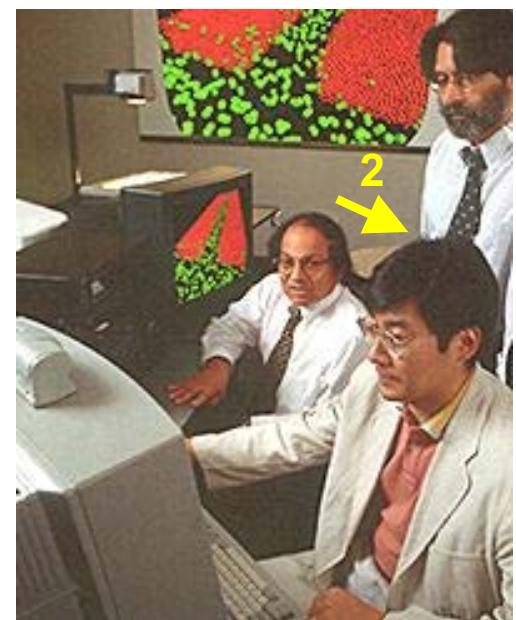
Anees Rahman ('67)



Anees Rahman & Priya Vashishta
at Argonne National Lab ('81)

<https://aiichironakano.github.io/phys516/Battimelli-ComputerMeetsPhysics-Springer20.pdf>, pp. 58 & 128

Priya Vashishta,
Rajiv Kalia
and AN ('02)



Your Rahman number is 3

Molecular Dynamics Equations

Given initial atomic positions & velocities $\{(\vec{r}_i(0), \vec{v}_i(0)) | i = 0, \dots, N - 1\}$,
 obtain those at later times $\{(\vec{r}_i(t), \vec{v}_i(t)) | i = 0, \dots, N - 1; t > 0\}$
 by integrating the ordinary differential equation,

$$\ddot{\vec{r}}_k(t) = \vec{a}_k(t) = -\frac{\partial}{\partial \vec{r}_k} \sum_{i < j} u(r_{ij}) = \sum_{i < j} \vec{r}_{ij}(t) \left(-\frac{1}{r} \frac{du}{dr} \right)_{r=r_{ij}(t)} (\delta_{ik} - \delta_{jk})$$

$$\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$$

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

where

$$-\frac{1}{r} \frac{du}{dr} = \frac{48}{r^2} \left(\frac{1}{r^{12}} - \frac{1}{2r^6} \right)$$

(for the Lennard-Jones potential
in a dimensionless unit)

Force calculation algorithm— $O(N^2)$:

for $i = 0$ to $N-1$, $\vec{a}_i = 0$

for $i = 0$ to $N-2$

 for $j = i+1$ to $N-1$

$$\text{compute } \vec{a} = \vec{r}_{ij} \left(-\frac{1}{r} \frac{du}{dr} \right)_{r=|\vec{r}_{ij}|}$$

$$\vec{a}_i += \vec{a}$$

$$\vec{a}_{j-} = \vec{a}$$

```

for (n=0; n<nAtom; n++)
    for (k=0; k<3; k++) ra[n][k] = 0.0;
for (j1=0; j1<nAtom-1; j1++) {
    for (j2=j1+1; j2<nAtom; j2++) {
        for (rr=0.0, k=0; k<3; k++) {
            dr[k] = r[j1][k] - r[j2][k];
            dr[k] = dr[k] - SignR(RegionH[k], dr[k]-RegionH[k])
                    - SignR(RegionH[k], dr[k]+RegionH[k]);
            rr = rr + dr[k]*dr[k];
        }
        if (rr < rrCut) {
            ri2 = 1.0/rr; ri6 = ri2*ri2*ri2; r1 = sqrt(rr);
            fcVal = 48.0*ri2*ri6*(ri6-0.5) + Duc/r1;
            for (k=0; k<3; k++) {
                f = fcVal*dr[k];
                ra[j1][k] = ra[j1][k] + f;
                ra[j2][k] = ra[j2][k] - f;
            }
        }
    }
}
    }
```

in computeAccel() in md.c

Analytic Formula for Forces

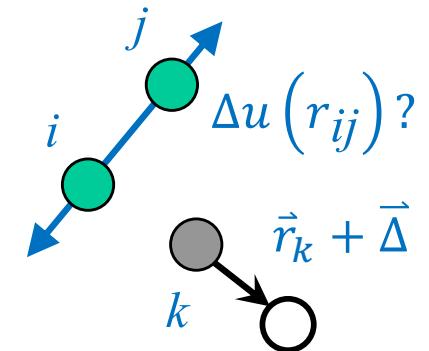
$$\vec{a}_k = -\frac{\partial}{\partial \vec{r}_k} \sum_{i < j} u(r_{ij}) = -\sum_{i < j} \frac{\partial r_{ij}}{\partial \vec{r}_k} \frac{du}{dr_{ij}} \quad \text{Chain rule}$$

$$\frac{\partial r_{ij}}{\partial \vec{r}_k} = \left(\frac{\partial}{\partial x_k}, \frac{\partial}{\partial y_k}, \frac{\partial}{\partial z_k} \right) \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$$

$$= \frac{(2(x_i - x_j), 2(y_i - y_j), 2(z_i - z_j))}{2\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} (\delta_{ik} - \delta_{jk})$$

$$= \frac{\vec{r}_{ij}}{r_{ij}} (\delta_{ik} - \delta_{jk})$$

$$\left(\because \frac{d}{dx} [f(x)]^{1/2} = \frac{1}{2} [f(x)]^{-1/2} \frac{df}{dx} \right)$$

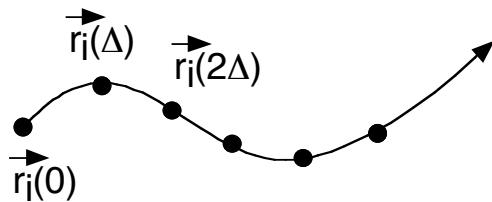


$$\begin{aligned} \frac{du}{dr} &= 4 \frac{d}{dr} \left(\frac{1}{r^{12}} + \frac{1}{r^6} \right) = 4 \left(-\frac{12}{r^{13}} + \frac{6}{r^7} \right) = -\frac{48}{r} \left(\frac{1}{r^{12}} - \frac{1}{2r^6} \right) \\ &\left(\because \frac{d}{dr} r^{-n} = -nr^{-n-1} \right) \end{aligned}$$

Time Discretization

- **Snapshots with time interval Δ :** double DeltaT

$$(\vec{r}_i(0), \vec{v}_i(0)) \mapsto (\vec{r}_i(\Delta), \vec{v}_i(\Delta)) \mapsto (\vec{r}_i(2\Delta), \vec{v}_i(2\Delta)) \mapsto \dots$$



- **Question:** How to predict the next state, $(\vec{r}_i(t + \Delta), \vec{v}_i(t + \Delta))$, from the current state, $(\vec{r}_i(t), \vec{v}_i(t))$?
- **Solution:** Taylor expansion

$$f(x_0 + h) = \sum_{n=0}^{\infty} \frac{h^n}{n!} \frac{d^n f}{dx^n} \Big|_{x=x_0} = f(x_0) + h \frac{df}{dx} \Big|_{x=x_0} + \frac{h^2}{2} \frac{d^2 f}{dx^2} \Big|_{x=x_0} + \frac{h^3}{3!} \frac{d^3 f}{dx^3} \Big|_{x=x_0} + \dots$$

See the note on Taylor's expansion

Verlet Discretization

Position:

$$\vec{r}_i(t + \Delta) = \vec{r}_i(t) + \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 + \frac{1}{6}\ddot{\vec{r}}_i(t)\Delta^3 + O(\Delta^4)$$
$$+ \quad \vec{r}_i(t - \Delta) = \vec{r}_i(t) - \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 - \frac{1}{6}\ddot{\vec{r}}_i(t)\Delta^3 + O(\Delta^4)$$

$$\vec{r}_i(t + \Delta) + \vec{r}_i(t - \Delta) = 2\vec{r}_i(t) + \vec{a}_i(t)\Delta^2 + O(\Delta^4)$$

$$\therefore \vec{r}_i(t + \Delta) = 2\vec{r}_i(t) - \vec{r}_i(t - \Delta) + \vec{a}_i(t)\Delta^2 + O(\Delta^4)$$

Velocity:

$$\vec{r}_i(t + \Delta) = \vec{r}_i(t) + \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 + \frac{1}{6}\ddot{\vec{r}}_i(t)\Delta^3 + O(\Delta^4)$$
$$- \quad \vec{r}_i(t - \Delta) = \vec{r}_i(t) - \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 - \frac{1}{6}\ddot{\vec{r}}_i(t)\Delta^3 + O(\Delta^4)$$

$$\vec{r}_i(t + \Delta) - \vec{r}_i(t - \Delta) = 2\vec{v}_i(t)\Delta + O(\Delta^3)$$

$$\therefore \vec{v}_i(t) = \frac{\vec{r}_i(t + \Delta) - \vec{r}_i(t - \Delta)}{2\Delta} + O(\Delta^2)$$

Verlet Algorithm

Verlet discretization:

$$\begin{cases} \vec{r}_i(t + \Delta) = 2\vec{r}_i(t) - \vec{r}_i(t - \Delta) + \vec{a}_i(t)\Delta^2 + O(\Delta^4) \\ \vec{v}_i(t) = \frac{\vec{r}_i(t + \Delta) - \vec{r}_i(t - \Delta)}{2\Delta} + O(\Delta^2) \end{cases}$$



Verlet algorithm:

Given $\vec{r}_i(t - \Delta)$ & $\vec{r}_i(t)$,

1. Compute $\vec{a}_i(t)$ as a function of $\{\vec{r}_i(t)\}$
2. $\vec{r}_i(t + \Delta) \leftarrow 2\vec{r}_i(t) - \vec{r}_i(t - \Delta) + \vec{a}_i(t)\Delta^2$
3. $\vec{v}_i(t) \leftarrow [\vec{r}_i(t + \Delta) - \vec{r}_i(t - \Delta)]/(2\Delta)$

Loup Verlet
(1931-2019)

```

Out           In
 $\{\vec{a}_i\} \leftarrow \text{computeAccel}(\{\vec{r}_i\})$ 

for (n=0; n<nAtom; n++) for (k=0; k<3; k++) ra[n][k] = 0.0;
for (j1=0; j1<nAtom-1; j1++) {
    for (j2=j1+1; j2<nAtom; j2++) {
        for (rr=0.0, k=0; k<3; k++) {
            dr[k] = r[j1][k] - r[j2][k] ← In
            dr[k] = dr[k] - SignR(RegionH[k], dr[k]-RegionH[k])
                - SignR(RegionH[k], dr[k]+RegionH[k]);
            rr = rr + dr[k]*dr[k];
        }
        if (rr < rrCut) {
            ri2 = 1.0/rr; ri6 = ri2*ri2*ri2; r1 = sqrt(rr);
            fcVal = 48.0*ri2*ri6*(ri6-0.5) + Duc/r1;
            for (k=0; k<3; k++) {
                f = fcVal*dr[k];
                ra[j1][k] = ra[j1][k] + f;
                ra[j2][k] = ra[j2][k] - f;
            }
        }
    }
}

```

Out ← **Out**

Drawback: Positions & velocities are not simultaneously updated for the same time step

Loup Verlet, Phys. Rev. 159, 98 ('67)

Velocity Verlet Algorithm

Theorem: The following algebraic equation gives the same sequence of states, $(\vec{r}_i(n\Delta), \vec{v}_i(n\Delta))$, as that obtained by the Verlet discretization (see [lecture note p. 7](#) for a proof and the velocity error).

$$\begin{cases} \vec{r}_i(t + \Delta) = \vec{r}_i(t) + \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 + O(\Delta^4) \\ \vec{v}_i(t + \Delta) = \vec{v}_i(t) + \frac{\vec{a}_i(t) + \vec{a}_i(t + \Delta)}{2}\Delta + O(\Delta^3) \end{cases}$$

Velocity Verlet algorithm:

Given $(\vec{r}_i(t), \vec{v}_i(t))$,

$$\vec{v}_i(t) + \frac{\vec{a}_i(t) + \vec{a}_i(t + \Delta)}{2}\Delta \stackrel{\equiv \vec{v}_i\left(t + \frac{\Delta}{2}\right)}{=} \vec{v}_i(t) + \frac{\vec{a}_i(t)}{2}\Delta + \frac{\vec{a}_i(t + \Delta)}{2}\Delta$$

1. Compute $\vec{a}_i(t)$ as a function of $\{\vec{r}_i(t)\}$ $\rightarrow \{\vec{a}_i(t)\} \leftarrow \text{computeAccel}(\{\vec{r}_i(t)\})$
2. $\vec{v}_i(t + \frac{\Delta}{2}) \leftarrow \vec{v}_i(t) + \frac{\Delta}{2}\vec{a}_i(t)$
3. $\vec{r}_i(t + \Delta) \leftarrow \vec{r}_i(t) + \vec{v}_i(t + \frac{\Delta}{2})\Delta$
4. Compute $\vec{a}_i(t + \Delta)$ as a function of $\{\vec{r}_i(t + \Delta)\}$ $\rightarrow \{\vec{a}_i(t + \Delta)\} \leftarrow \text{computeAccel}(\{\vec{r}_i(t + \Delta)\})$
5. $\vec{v}_i(t + \Delta) \leftarrow \vec{v}_i(t + \frac{\Delta}{2}) + \frac{\Delta}{2}\vec{a}_i(t + \Delta)$

$$r_i(t + \Delta) = r_i(t) + \left(\vec{v}_i(t) + \frac{\vec{a}_i(t)}{2}\Delta \right) \Delta$$

Velocity Verlet Algorithm for StepLimit Steps

Initialize (\vec{r}_i, \vec{v}_i) for all i

Compute \vec{a}_i for all i as a function of $\{\vec{r}_i\}$ **function computeAccel()**

for stepCount = 1 to StepLimit

do the following **function singleStep()**

$\vec{v}_i \leftarrow \vec{v}_i + \vec{a}_i \Delta / 2$ for all i

$\vec{r}_i \leftarrow \vec{r}_i + \vec{v}_i \Delta$ for all i

Compute \vec{a}_i for all i as a function of $\{\vec{r}_i\}$ **function computeAccel()**

$\vec{v}_i \leftarrow \vec{v}_i + \vec{a}_i \Delta / 2$ for all i

endfor

**stepLimit+1 calls to
function computeAccel()**

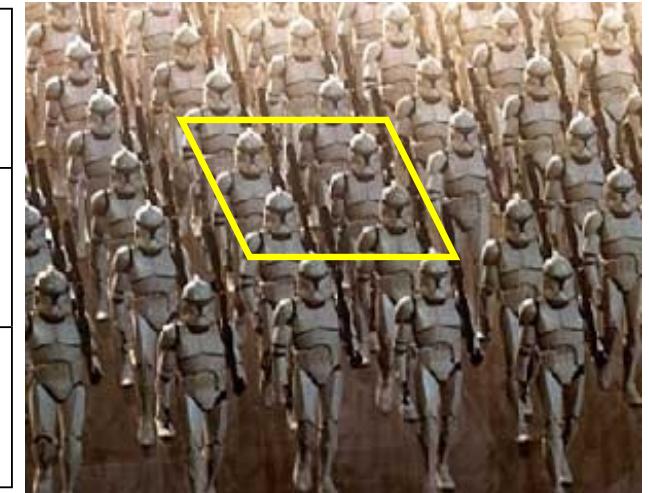
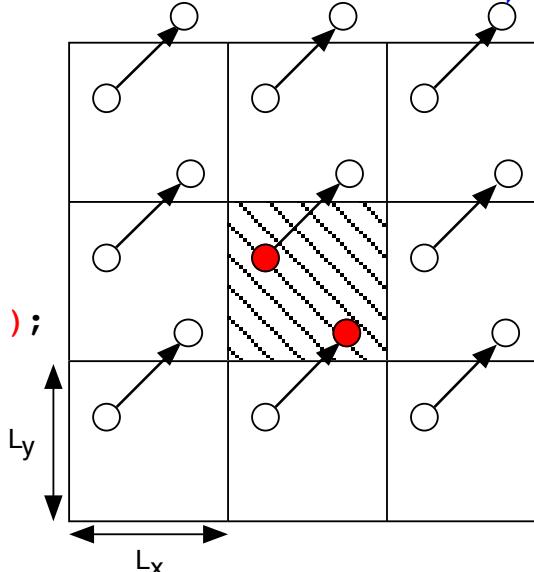
```
computeAccel();
for (stepCount=1; stepCount<=StepLimit; stepCount++) {
    singleStep(); // computeAccel() called once here
    if (stepCount%StepAvg == 0) evalProps();
}
```

in main() in md.c

Periodic Boundary Conditions

- Replicate a simulation box of size $L_x \times L_y \times L_z$ to form an infinite lattice to simulate bulk materials
- Keep only the central image: $0 \leq x_i < L_x, 0 \leq y_i < L_y, 0 \leq z_i < L_z$

```
void applyBoundaryCond() {
    int n,k;
    for (n=0; n<nAtom; n++)
        for (k=0; k<3; k++)
            r[n][k] = r[n][k]
            -SignR(RegionH[k],r[n][k])
            -SignR(RegionH[k],r[n][k]-Region[k]);
}
```



- Folding back atoms

$$\begin{cases} x_i \leftarrow x_i - \text{SignR}\left(\frac{L_x}{2}, x_i\right) - \text{SignR}\left(\frac{L_x}{2}, x_i - L_x\right) \\ y_i \leftarrow y_i - \text{SignR}\left(\frac{L_y}{2}, y_i\right) - \text{SignR}\left(\frac{L_y}{2}, y_i - L_y\right) \\ z_i \leftarrow z_i - \text{SignR}\left(\frac{L_z}{2}, z_i\right) - \text{SignR}\left(\frac{L_z}{2}, z_i - L_z\right) \end{cases}$$

$$\text{SignR}\left(\frac{L_x}{2}, x_i\right) = \begin{cases} \frac{L_x}{2} & x_i > 0 \\ -\frac{L_x}{2} & x_i \leq 0 \end{cases}$$

double Region[3]: (L_x, L_y, L_z) **double RegionH[3]:** $(L_x/2, L_y/2, L_z/2)$

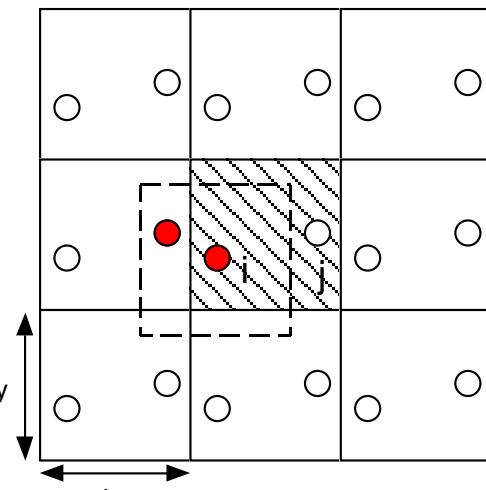
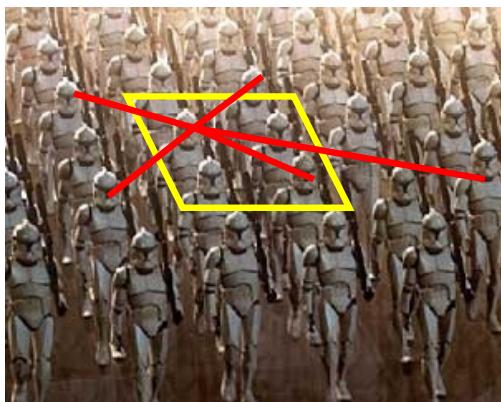
Minimum Image Convention

- Compute the interaction of an atom with only the nearest image
- Exact if the range of interaction, $r_c < \min(L_x, L_y, L_z)/2$
- Pick the other atoms only in the box centered around each atom

$$\vec{r}_{ij} \equiv \vec{r}_i - \vec{r}_j = (x_{ij}, y_{ij}, z_{ij})$$

$$\begin{cases} -\frac{L_x}{2} \leq x_{ij} < \frac{L_x}{2} \\ -\frac{L_y}{2} \leq y_{ij} < \frac{L_y}{2} \\ -\frac{L_z}{2} \leq z_{ij} < \frac{L_z}{2} \end{cases}$$

$$\begin{cases} x_{ij} \leftarrow x_{ij} - \text{SignR}\left(\frac{L_x}{2}, x_{ij} + \frac{L_x}{2}\right) - \text{SignR}\left(\frac{L_x}{2}, x_{ij} - \frac{L_x}{2}\right) \\ y_{ij} \leftarrow y_{ij} - \text{SignR}\left(\frac{L_y}{2}, y_{ij} + \frac{L_y}{2}\right) - \text{SignR}\left(\frac{L_y}{2}, y_{ij} - \frac{L_y}{2}\right) \\ z_{ij} \leftarrow z_{ij} - \text{SignR}\left(\frac{L_z}{2}, z_{ij} + \frac{L_z}{2}\right) - \text{SignR}\left(\frac{L_z}{2}, z_{ij} - \frac{L_z}{2}\right) \end{cases}$$



```

for (j1=0; j1<nAtom-1; j1++) {
    for (j2=j1+1; j2<nAtom; j2++) {
        for (rr=0.0, k=0; k<3; k++) {
            dr[k] = r[j1][k] - r[j2][k];
            dr[k] = dr[k] - SignR(RegionH[k], dr[k]-RegionH[k])
                    - SignR(RegionH[k], dr[k]+RegionH[k]);
            rr = rr + dr[k]*dr[k];
        }
        if (rr < rrCut) {
            ri2 = 1.0/rr; ri6 = ri2*ri2*ri2; r1 = sqrt(rr);
            fcVal = 48.0*ri2*ri6*(ri6-0.5) + Duc/r1;
            for (k=0; k<3; k++) {
                f = fcVal*dr[k];
                ra[j1][k] = ra[j1][k] + f;
                ra[j2][k] = ra[j2][k] - f;
            }
        }
    }
}

```

in computeAccel() in md.c

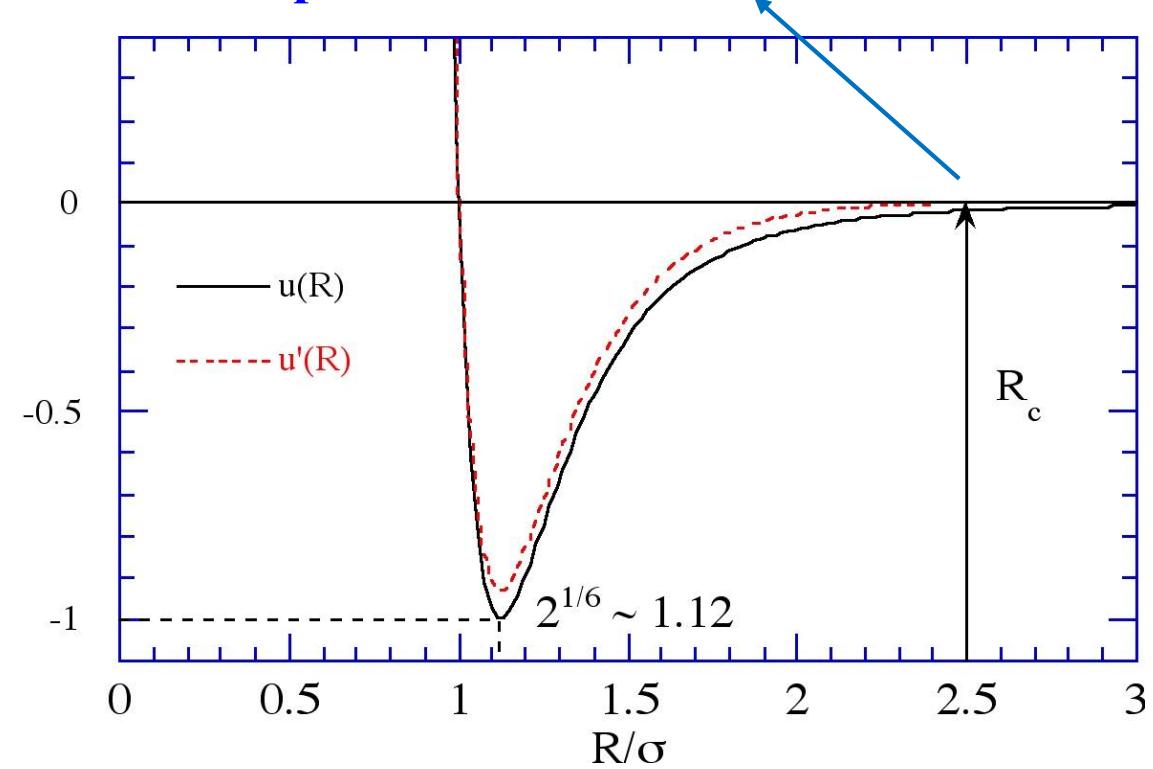
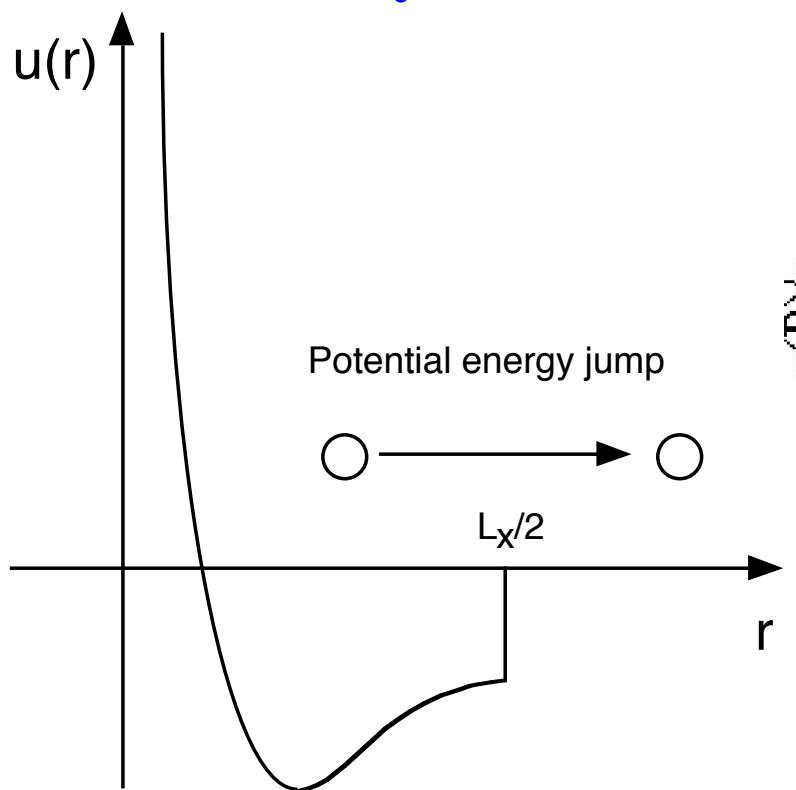
Shifted Potential

- Truncate the interaction $u(r)$ at a cut-off radius $r_c < \min(L_x, L_y, L_z)/2$ to make the minimal image convention exact
- Shift the potential to make $u(r)$ & du/dr continuous at $r = r_c$

$$u'(r) = \begin{cases} u(r) - u(r_c) - (r - r_c) \frac{du}{dr} \Big|_{r=r_c} & r < r_c \\ 0 & r \geq r_c \end{cases}$$

$$\begin{aligned} u'(r) &= u(r) - u(r_c) \xrightarrow[r \rightarrow r_c]{} 0 \\ \frac{du'}{dr} &= \frac{du}{dr} - \frac{du}{dr} \Big|_{r=r_c} \xrightarrow[r \rightarrow r_c]{} 0 \end{aligned}$$

- We take $r_c = 2.5\sigma$ for the Lennard-Jones potential



Energy & Temperature

- Energy

$$E = K + V = \sum_{i=0}^{N-1} \frac{m}{2} |\dot{\vec{r}}_i|^2 + \sum_{i < j} u(r_{ij})$$

↑ ↑ ↑
total, kinetic & potential energies

- Temperature, T

$$\frac{3Nk_B T}{2} = \sum_{i=0}^{N-1} \frac{m}{2} |\dot{\vec{r}}_i|^2$$

where $k_B = 1.38062 \times 10^{-16}$ erg/K—Boltzmann constant

- In the Lennard-Jones unit,

$$\frac{T}{\varepsilon/k_B} = \frac{1}{3N} \sum_{i=0}^{N-1} |\dot{\vec{r}}_i'|^2$$

where $\varepsilon/k_B \sim 120$ K

```
double kinEnergy;      /* Kinetic energy */  
double potEnergy;      /* Potential energy */  
double totEnergy;      /* Total energy */  
double temperature;
```

Energy Conservation

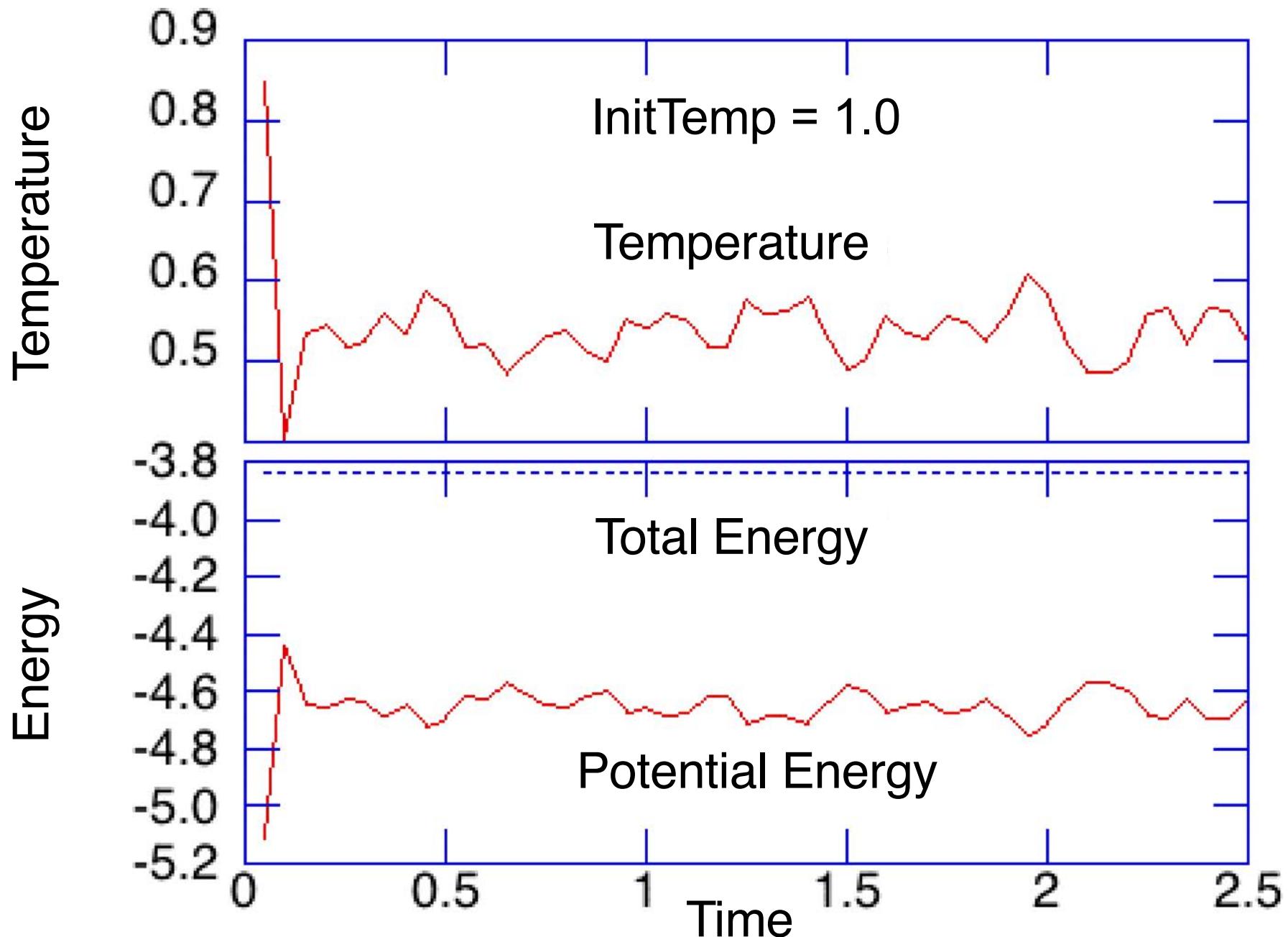
- The total energy, E , does not change in time, if Newton's equation is exactly integrated

$$\dot{E} = \dot{K} + \dot{V}$$

Chain rule!

$$\begin{aligned} &= \sum_{i=0}^{N-1} \frac{m}{2} \frac{d}{dt} \left(\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2 \right) + \sum_{i=0}^{N-1} \left(\frac{dx_i}{dt} \frac{\partial V}{\partial x_i} + \frac{dy_i}{dt} \frac{\partial V}{\partial y_i} + \frac{dz_i}{dt} \frac{\partial V}{\partial z_i} \right) \\ &= \sum_{i=0}^{N-1} \frac{m}{2} 2(\dot{x}_i \ddot{x}_i + \dot{y}_i \ddot{y}_i + \dot{z}_i \ddot{z}_i) + \sum_{i=0}^{N-1} \left(\dot{x}_i \frac{\partial V}{\partial x_i} + \dot{y}_i \frac{\partial V}{\partial y_i} + \dot{z}_i \frac{\partial V}{\partial z_i} \right) \\ &= \sum_{i=0}^{N-1} m \dot{\vec{r}}_i \bullet \ddot{\vec{r}}_i + \sum_{i=0}^{N-1} \dot{\vec{r}}_i \bullet \frac{\partial V}{\partial \vec{r}_i} \\ &= \sum_{i=0}^{N-1} \dot{\vec{r}}_i \bullet (m \ddot{\vec{r}}_i - \vec{F}_i) \\ &= 0 \end{aligned}$$

Standard Output from md.c



Initial Condition: Atomic Positions

- Face centered cubic (FCC) lattice
- 4 atoms per unit cell of size $c_x = c_y = c_z = c$, at positions (in unit of c)

$(0,0,0), (0,1/2,1/2), (1/2,0,1/2), (1/2,1/2,0)$

double origAtom[4][3]

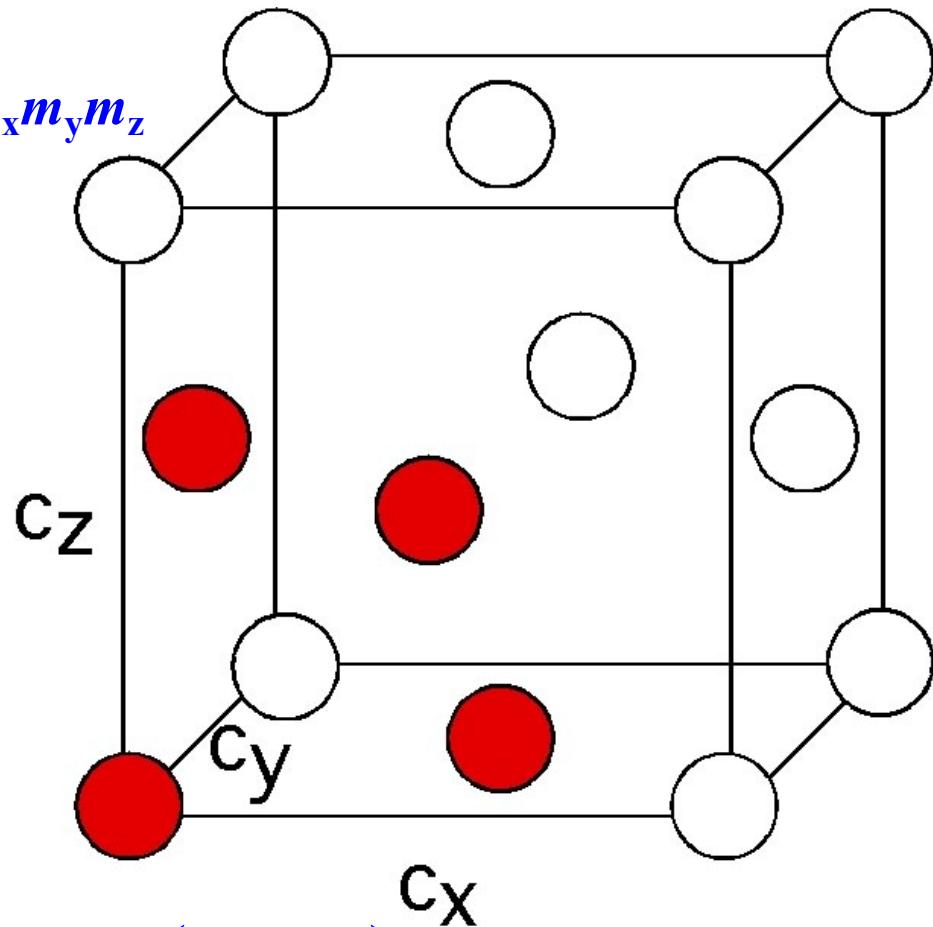
repeated $m_x \times m_y \times m_z$ times

- Total number of atoms $N = 4m_x m_y m_z$

- Density $\rho = 4/c^3$ or $c = (4/\rho)^{1/3}$

double Density

See [USC X-crystallography database](#)



double gap[3]: (c_x, c_y, c_z)

int InitUcell[3]: (m_x, m_y, m_z)

Initial Condition: Atomic Velocities

- Random velocities of the magnitude, v_0 , where

- Random velocities of the magnitude, v_0 , where

- For each atom, generate random velocity,

- where $\vec{\xi}$ is a randomly oriented vector of unit length

double InitTemp: Initial temperature

- For each atom, generate random velocity,

$$\vec{v}_i = v_0(\xi_0, \xi_1, \xi_2) = v_0 \vec{\xi}$$

where $\vec{\xi}$ is a randomly oriented vector of unit length

(Algorithm)

1. $\zeta_i = 2 * \text{rand}() / \text{RAND_MAX} - 1$ ($i = 0, 1$) so that $-1 \leq \zeta_i < 1$
2. $s^2 = \zeta_0^2 + \zeta_1^2$
3. if $s^2 < 1$, accept $(\xi_0, \xi_1, \xi_2) = (2\sqrt{1-s^2}\zeta_0, 2\sqrt{1-s^2}\zeta_1, 1-2s^2)$
4. else reject & go to 1

Random Point in a Unit Circle

(Algorithm)

1. $\zeta_i = 2 * \text{rand}() / \text{RAND_MAX} - 1$ ($i = 0, 1$) so that $-1 \leq \zeta_i < 1$
2. $s^2 = \zeta_0^2 + \zeta_1^2$
3. if $s^2 < 1$, accept $\vec{\zeta} = (\zeta_0, \zeta_1)$
4. else reject & go to 1

• Probability density

Probability to find a point in the shaded area

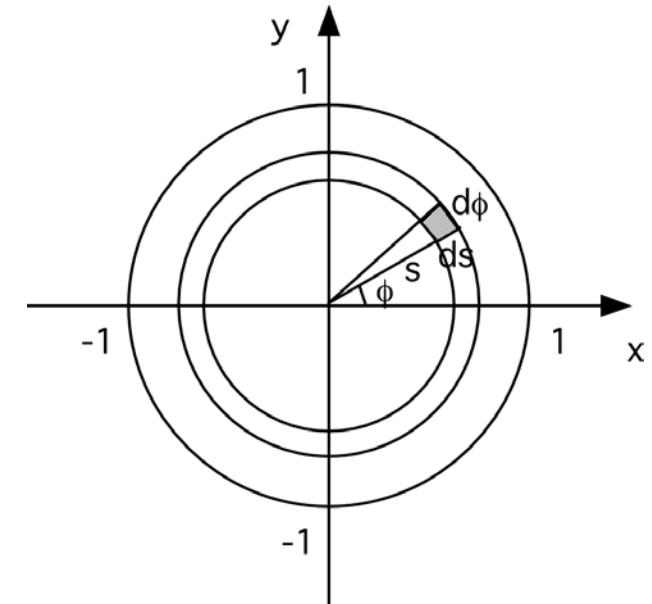
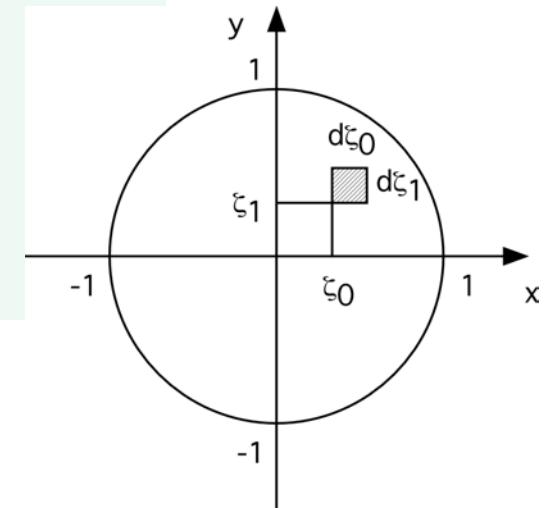
$$\begin{aligned}&= (\# \text{ of points in the shaded area}) / (\text{total } \# \text{ of generated points}) \\&= P(\zeta_0, \zeta_1) d\zeta_0 d\zeta_1 \\&= (d\zeta_0 d\zeta_1 \sim \text{shaded area}) / (\pi \cdot 1^2 \sim \text{area of unit circle}) \\&\therefore P(\zeta_0, \zeta_1) = 1/\pi\end{aligned}$$

• Probability density in polar coordinate

$$(\zeta_0, \zeta_1) = (s \cos \phi, s \sin \phi) \quad 0 \leq s < 1; 0 \leq \phi < 2\pi$$

$$P(s, \phi) ds d\phi = \frac{ds \cdot s d\phi}{\pi \cdot 1^2} \quad \text{if uniform}$$

$$\therefore P(s, \phi) = \frac{s}{\pi}$$



Random Point on a Unit Sphere

- Spherical coordinate

$$(\xi_0, \xi_1, \xi_2) = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta) \quad 0 \leq \theta < \pi, 0 \leq \varphi < 2\pi$$

- Probability distribution (if uniform)

$$P(\theta, \varphi)d\theta d\varphi = \frac{d\theta \cdot \sin\theta d\varphi}{4\pi \cdot 1^2}$$

$$\therefore P(\theta, \varphi) = \frac{\sin\theta}{4\pi}$$

- Equivalence to uniform distribution in a circle by coordinate transformation:

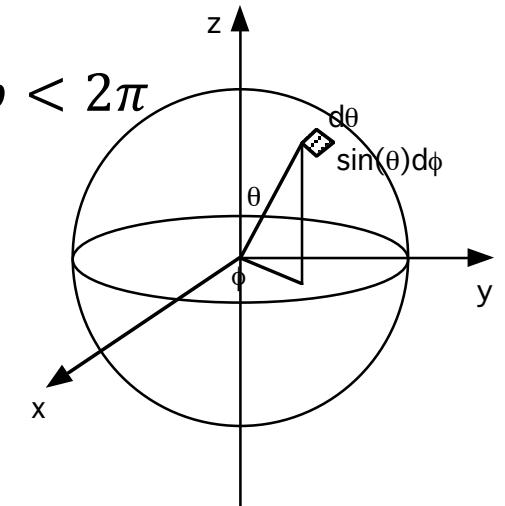
$$s \equiv \sin \frac{\theta}{2} \quad \because 0 \leq \theta < \pi \Leftrightarrow 0 \leq \frac{\theta}{2} < \frac{\pi}{2} \Leftrightarrow 0 \leq s \equiv \sin \frac{\theta}{2} < 1$$

$$\frac{sdsd\varphi}{\pi} = \frac{\sin \frac{\theta}{2} \frac{ds}{d\theta} d\theta d\varphi}{\pi} = \frac{\sin \frac{\theta}{2} \left(\frac{1}{2} \cos \frac{\theta}{2} \right) d\theta d\varphi}{\pi} = \frac{2 \sin \frac{\theta}{2} \cos \frac{\theta}{2} d\theta d\varphi}{4\pi} = \frac{\sin\theta d\theta d\varphi}{4\pi}$$

P(s,φ)

$$\begin{cases} \xi_0 = \sin\theta\cos\varphi = 2\sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\varphi = 2s\sqrt{1-s^2}\frac{\zeta_0}{s} = 2\sqrt{1-s^2}\zeta_0 \\ \xi_1 = \sin\theta\sin\varphi = 2\sin\frac{\theta}{2}\cos\frac{\theta}{2}\sin\varphi = 2s\sqrt{1-s^2}\frac{\zeta_1}{s} = 2\sqrt{1-s^2}\zeta_1 \\ \xi_2 = \cos\theta = 1 - 2\sin^2\frac{\theta}{2} = 1 - 2s^2 \end{cases}$$

P(θ,φ)



Liouville's Theorem

- Phase-space trajectory as a mapping

$$(x, p) \rightarrow (x', p')$$
$$t \qquad t + \Delta$$

- Phase-space volume conservation: Jacobian of the mapping (areal enlargement factor) = 1

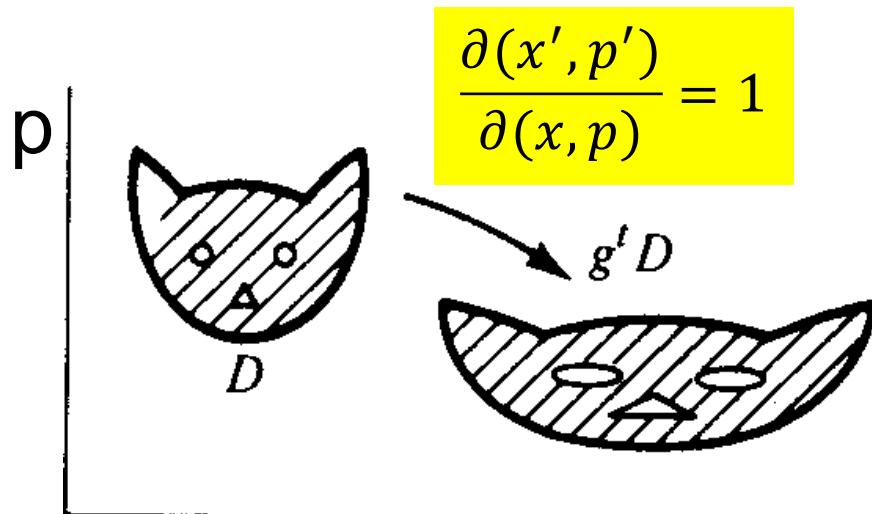
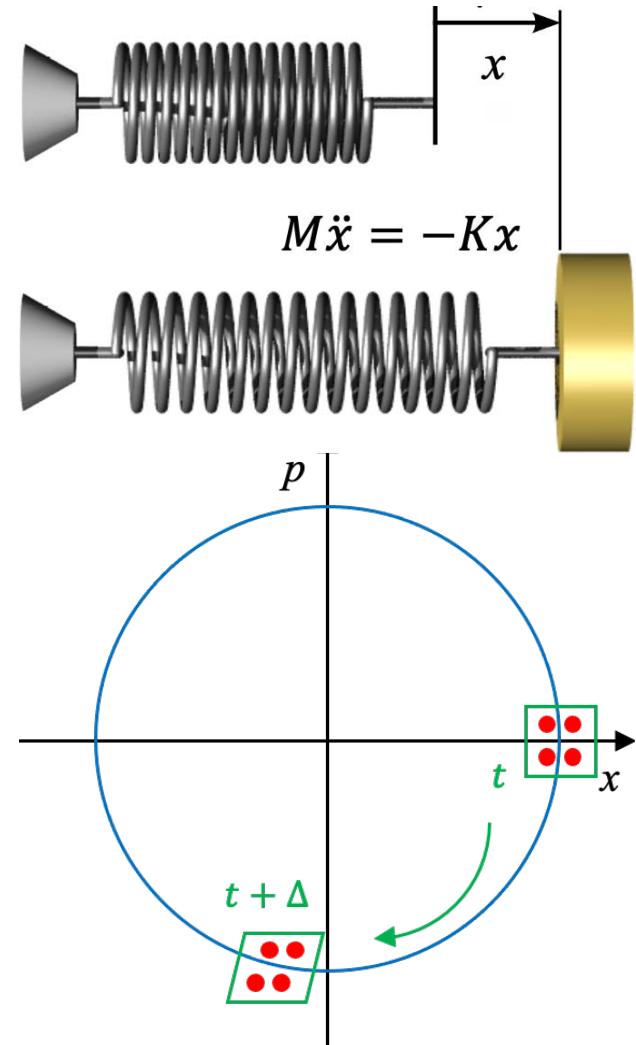


Figure 48 Conservation of volume \times

V. I. Arnold, *Mathematical Methods of Classical Mechanics*, 2nd Ed. (Springer, '89)

- The cat is a phase-space volume occupied by an ensemble of phase-space points, each representing a specific instance of the initial condition (see `anim_spring.c`)



Velocity Verlet Algorithm in 1D

$$\begin{cases} x' \leftarrow x + p\Delta + \frac{1}{2}a(x)\Delta^2 \equiv x'(x, p) \\ p' \leftarrow p + \frac{a(x) + a(x + p\Delta + \frac{1}{2}a(x)\Delta^2)}{2}\Delta \equiv p'(x, p) \end{cases}$$



Bottom-line: Velocity Verlet algorithm “exactly” satisfies Liouville’s theorem

Prove phase-space volume conservation: $\frac{\partial(x', p')}{\partial(x, p)} = 1$

What phase-space volume conservation means? It’s ensemble!

Compare an algorithm variant, Euler:

$$\begin{cases} x' \leftarrow x + p\Delta + \frac{1}{2}a(x)\Delta^2 \\ p' \leftarrow p + a(x)\Delta \end{cases}$$

$$\rightarrow \frac{\partial(x', p')}{\partial(x, p)} \neq 1$$

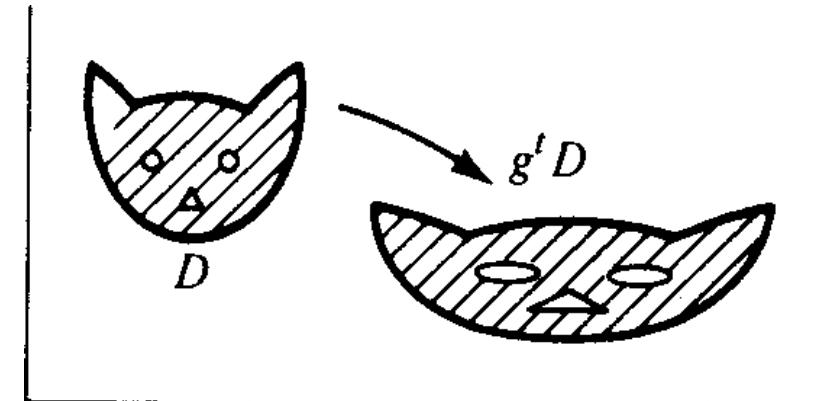


Figure 48 Conservation of volume

Algorithm Variants

(Velocity Verlet Algorithm)

Given a configuration, $\{\vec{r}_i(t), \vec{v}_i(t) \mid i = 1, \dots, N_{\text{atom}}\}$

1. (Compute the acceleration, $\vec{a}_i(t)$)
2. $\vec{v}_i(t + \Delta/2) \leftarrow \vec{v}_i(t) + \vec{a}_i(t)\Delta/2$
3. $\vec{r}_i(t + \Delta) \leftarrow \vec{r}_i(t) + \vec{v}_i(t + \Delta/2)\Delta$
4. Compute the updated acceleration, $\vec{a}_i(t + \Delta)$
5. $\vec{v}_i(t + \Delta) \leftarrow \vec{v}_i(t + \Delta/2) + \vec{a}_i(t + \Delta)\Delta/2$

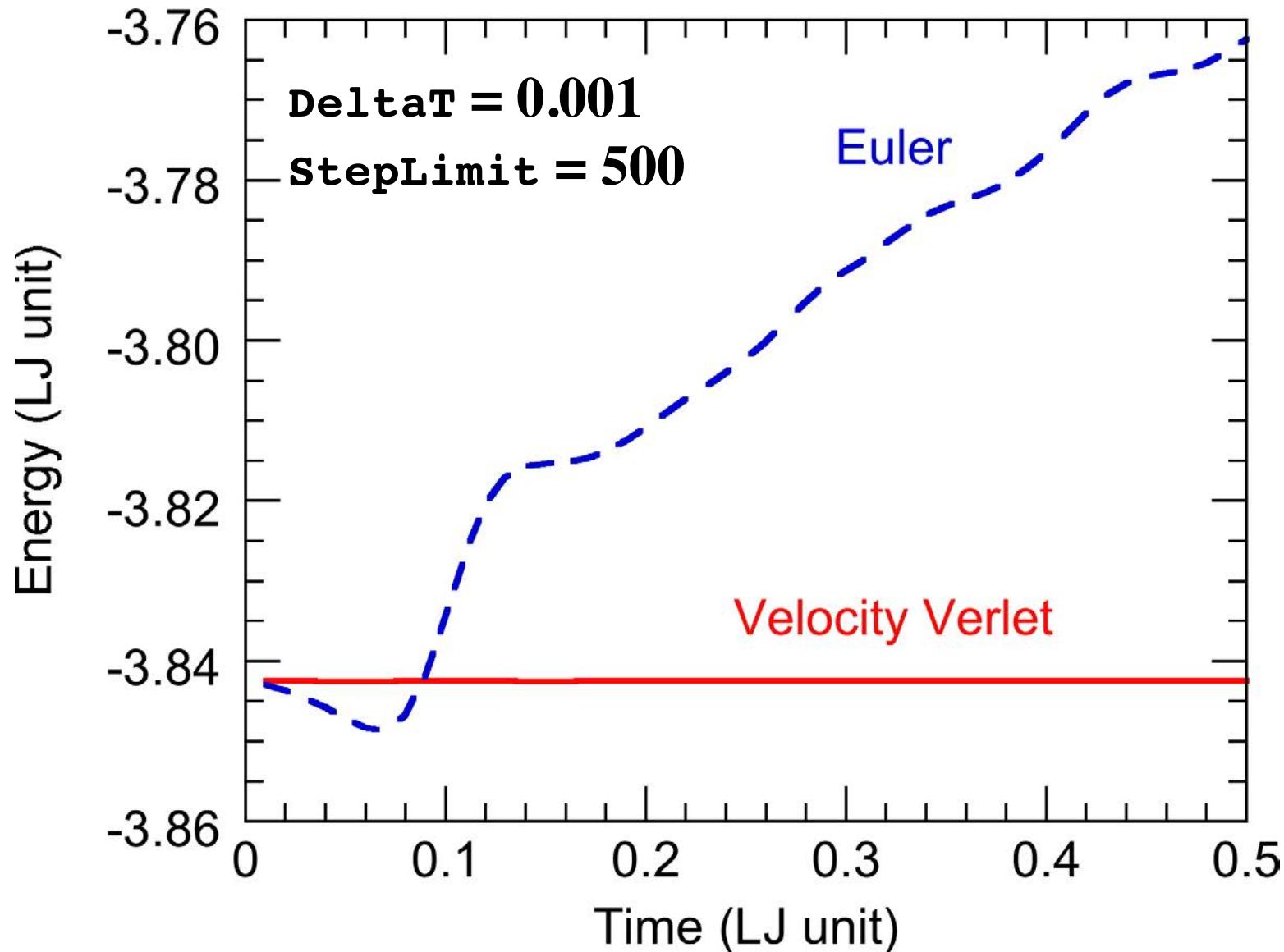
(Explicit Euler Algorithm) Only modify SingleStep()

Given a configuration, $\{\vec{r}_i(t), \vec{v}_i(t) \mid i = 1, \dots, N_{\text{atom}}\}$

1. Compute the acceleration, $\vec{a}_i(t)$
2. Update the positions, $\vec{r}_i(t + \Delta) \leftarrow \vec{r}_i(t) + \vec{v}_i(t)\Delta + \vec{a}_i(t)\Delta^2/2$
3. Update the velocities, $\vec{v}_i(t + \Delta) \leftarrow \vec{v}_i(t) + \vec{a}_i(t)\Delta$

Algorithm Variants

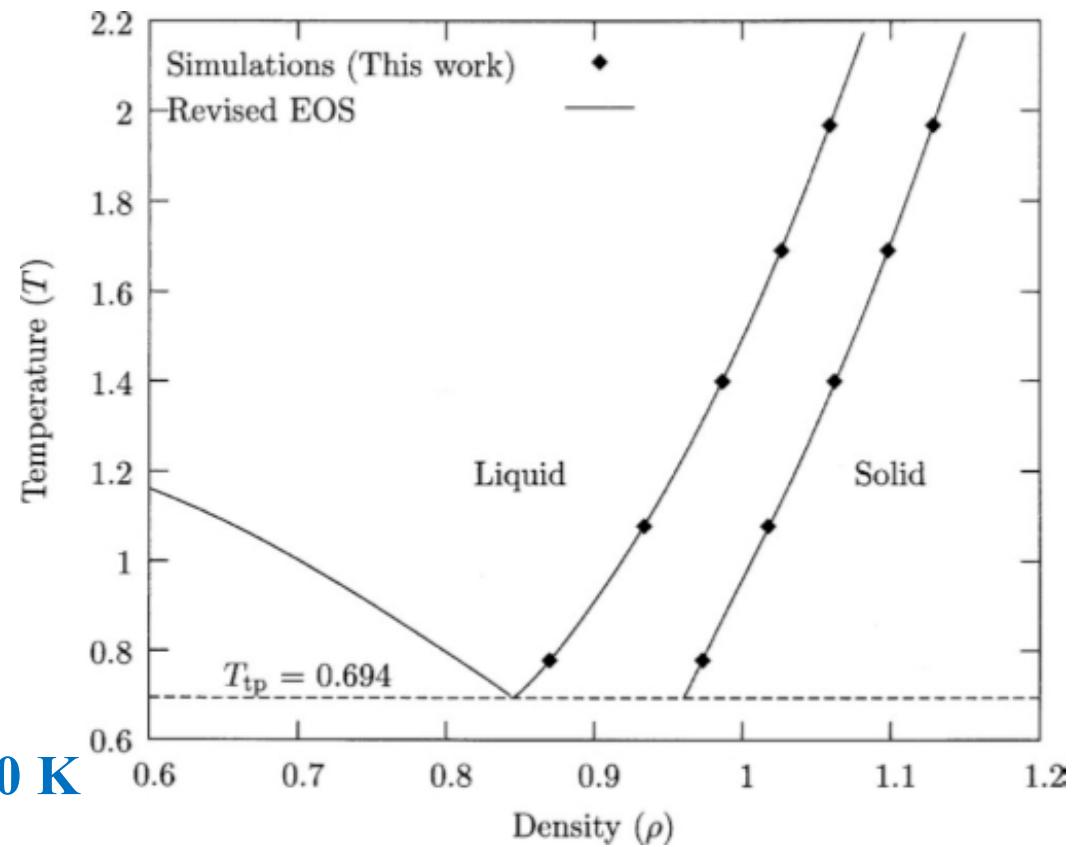
Energy conservation: Velocity-Verlet vs. explicit-Euler algorithms



MD Input Parameters

	in 0.0254 \AA^{-3} for Ar
3 3 3	InitUcell[3]
0.8	Density
1.0	InitTemp
0.001	DeltaT
500	StepLimit
10	StepAvg

in 120 K
in 2.2 ps



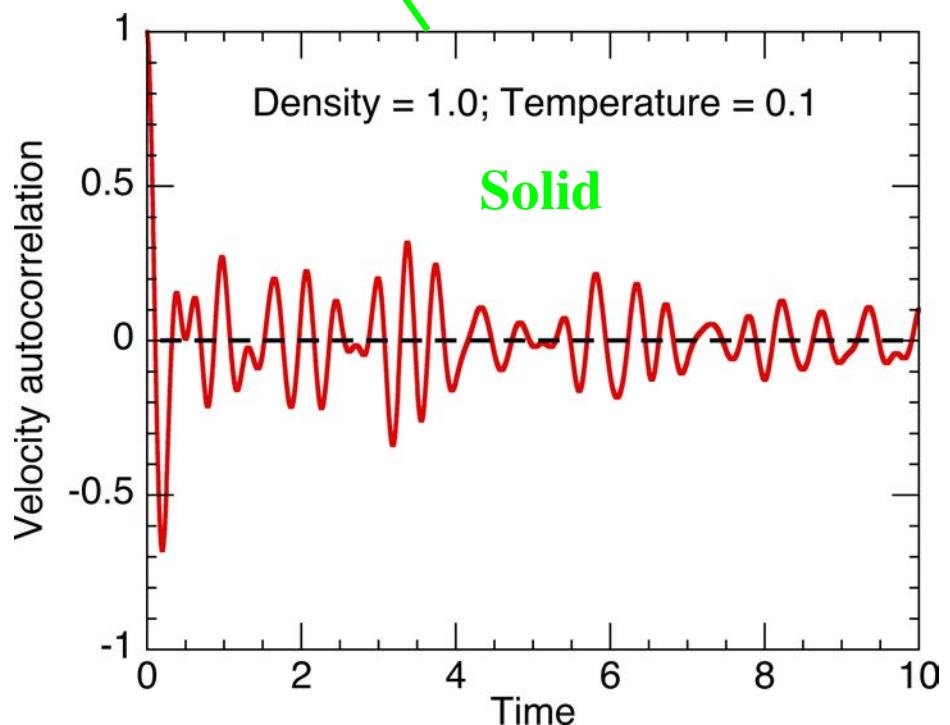
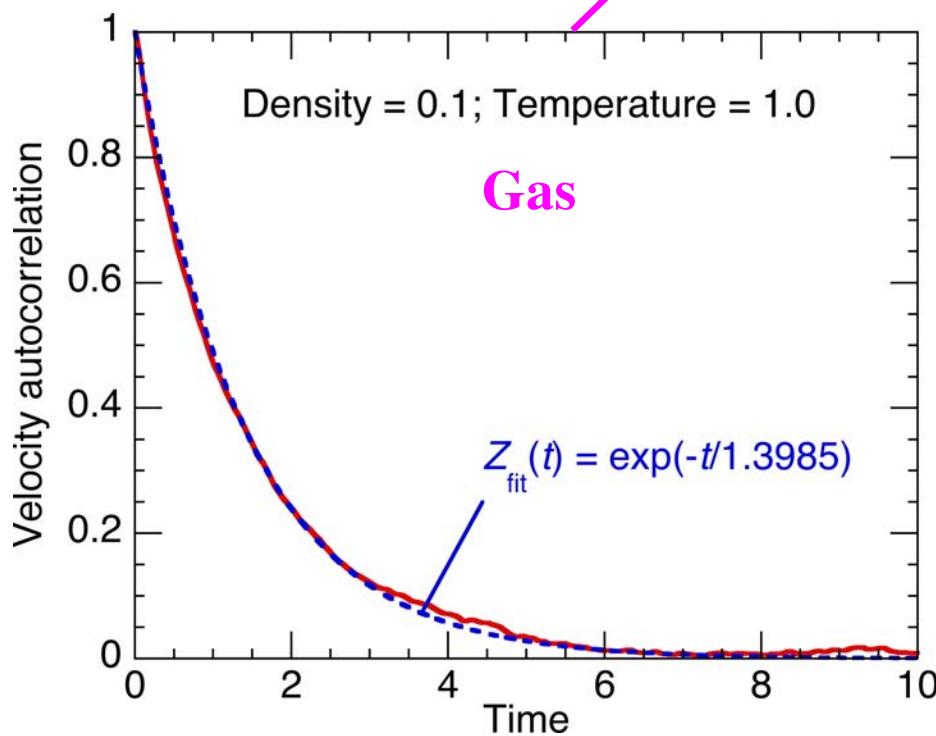
Velocity Autocorrelation

$$Z(t) = \frac{\langle \vec{v}_i(t + t_0) \cdot \vec{v}_i(t_0) \rangle}{\langle \vec{v}_i(t_0) \cdot \vec{v}_i(t_0) \rangle}$$

$$= \frac{\sum_{t_0} \sum_{i=0}^{N-1} \vec{v}_i(t + t_0) \cdot \vec{v}_i(t_0)}{\sum_{t_0} \sum_{i=0}^{N-1} \vec{v}_i(t_0) \cdot \vec{v}_i(t_0)}$$

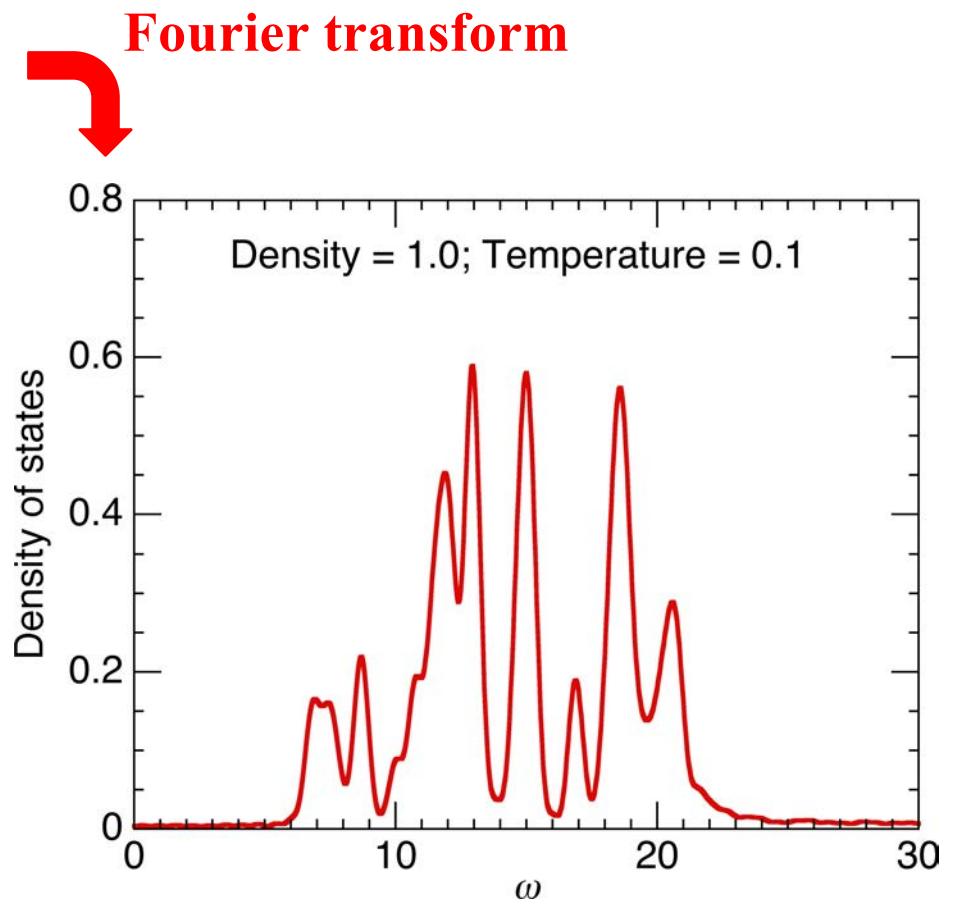
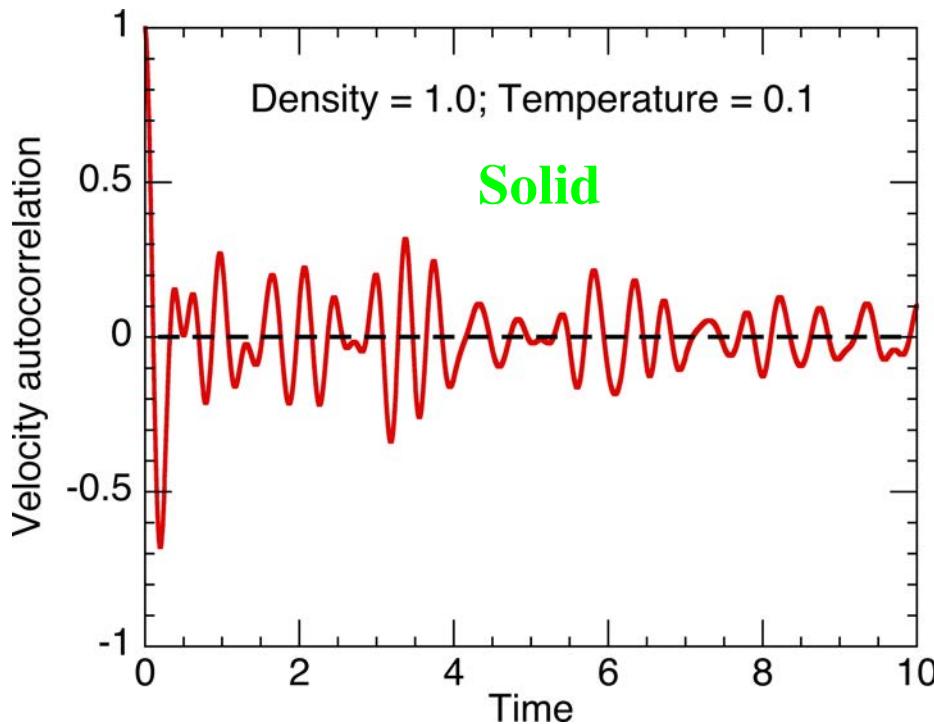
3	3	3
0.1	1.0	
1.0	0.1	
0.005		
2000		
1000		

InitUcell[3]
Density
InitTemp
DeltaT
StepLimit
StepAvg



Phonon Density of States

$$\tilde{Z}(\omega) \equiv \int_{-\infty}^{\infty} dt Z(t) e^{i\omega t} = 2 \int_0^{\infty} dt Z(t) \cos(\omega t)$$

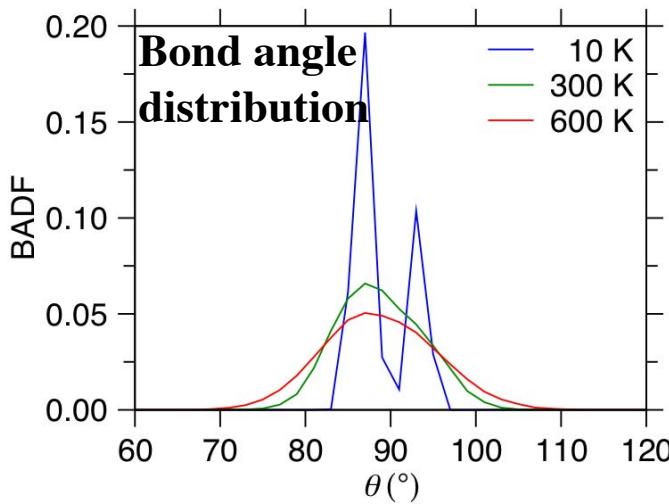
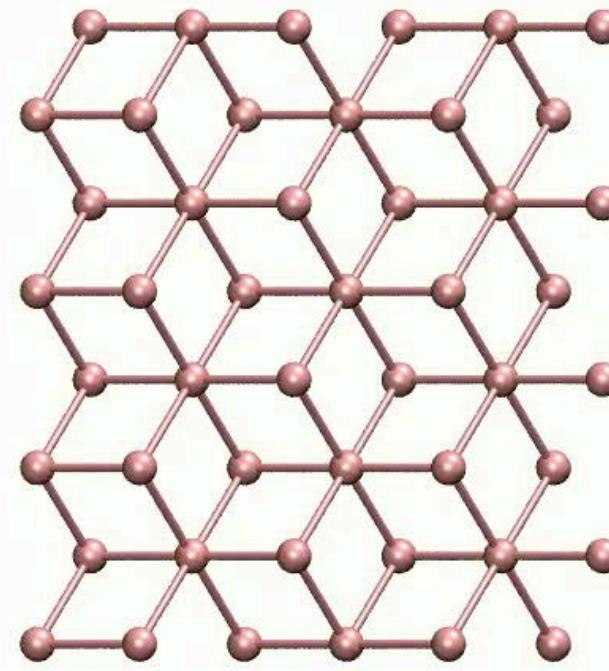
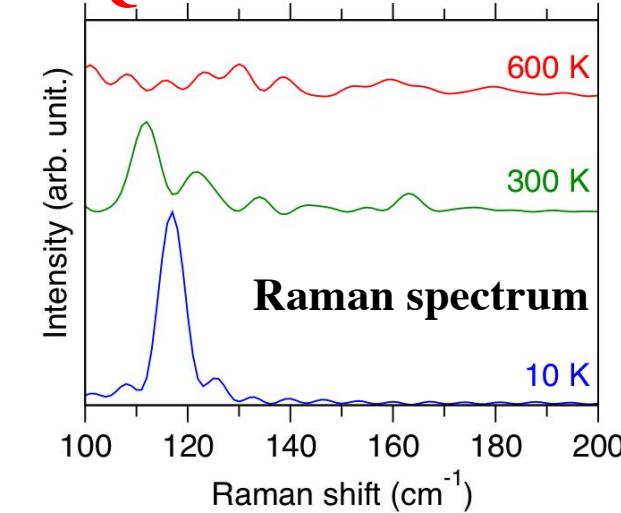


See lecture note on velocity autocorrelation function

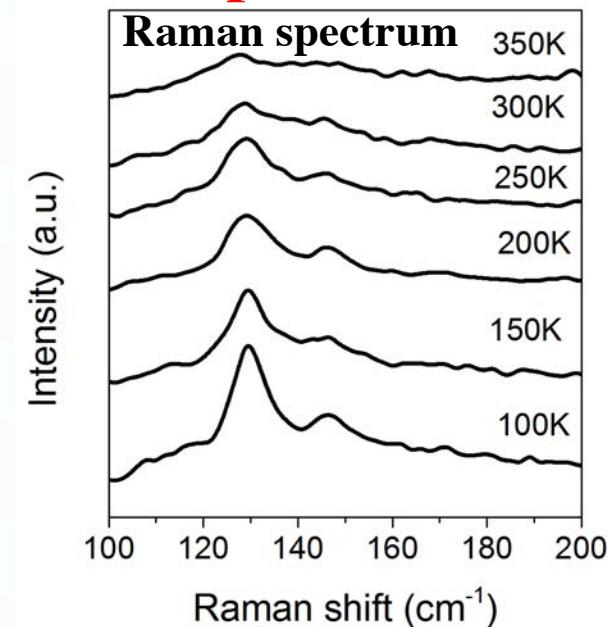
Example: Tellurene (Monolayer Tellurium)

- Quantum MD simulations explain the observed temperature dependence of Raman spectra as progressive angular disorder at elevated temperatures

QMD simulation



Experiment

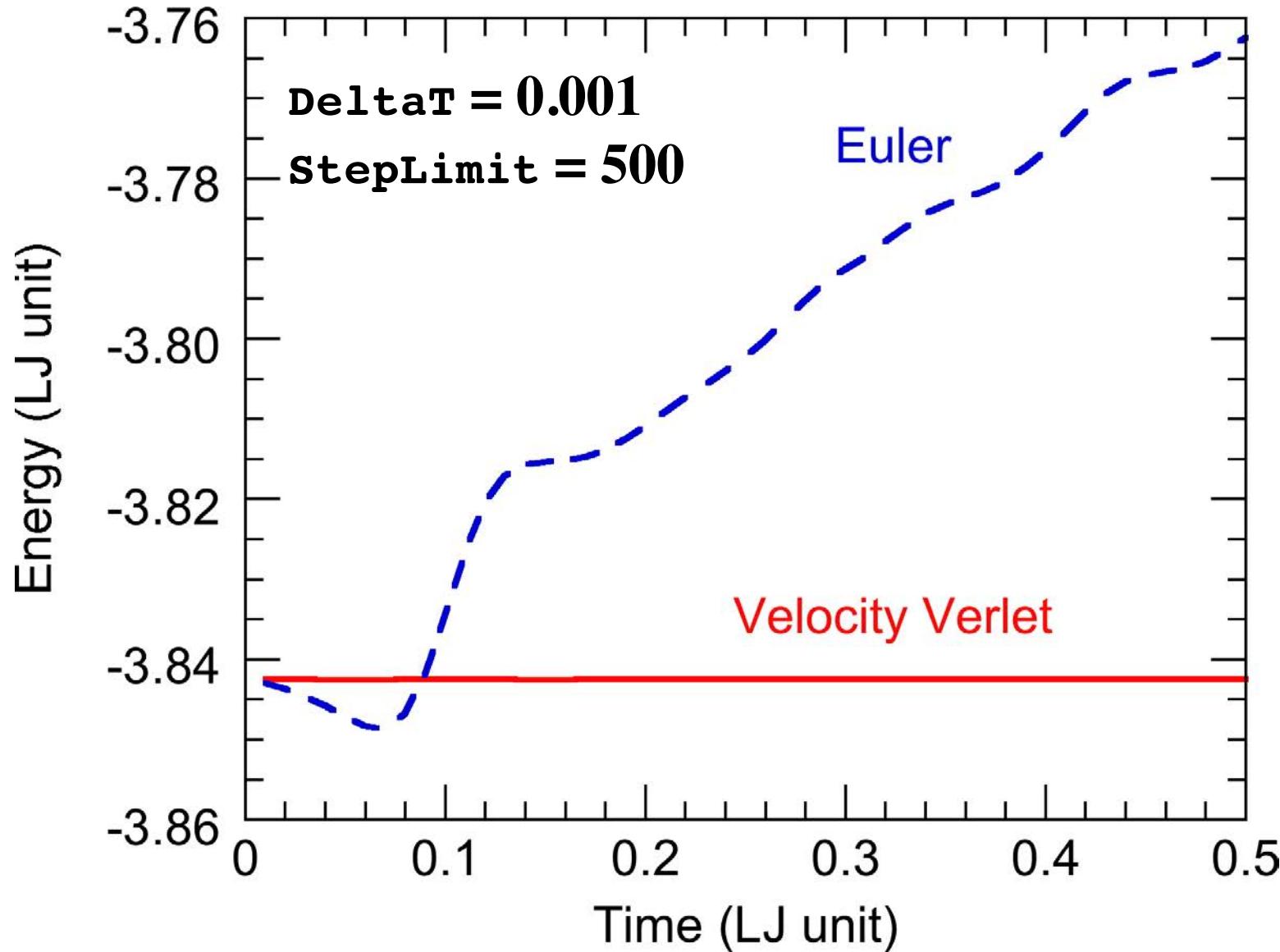


0.00 fs
600 K



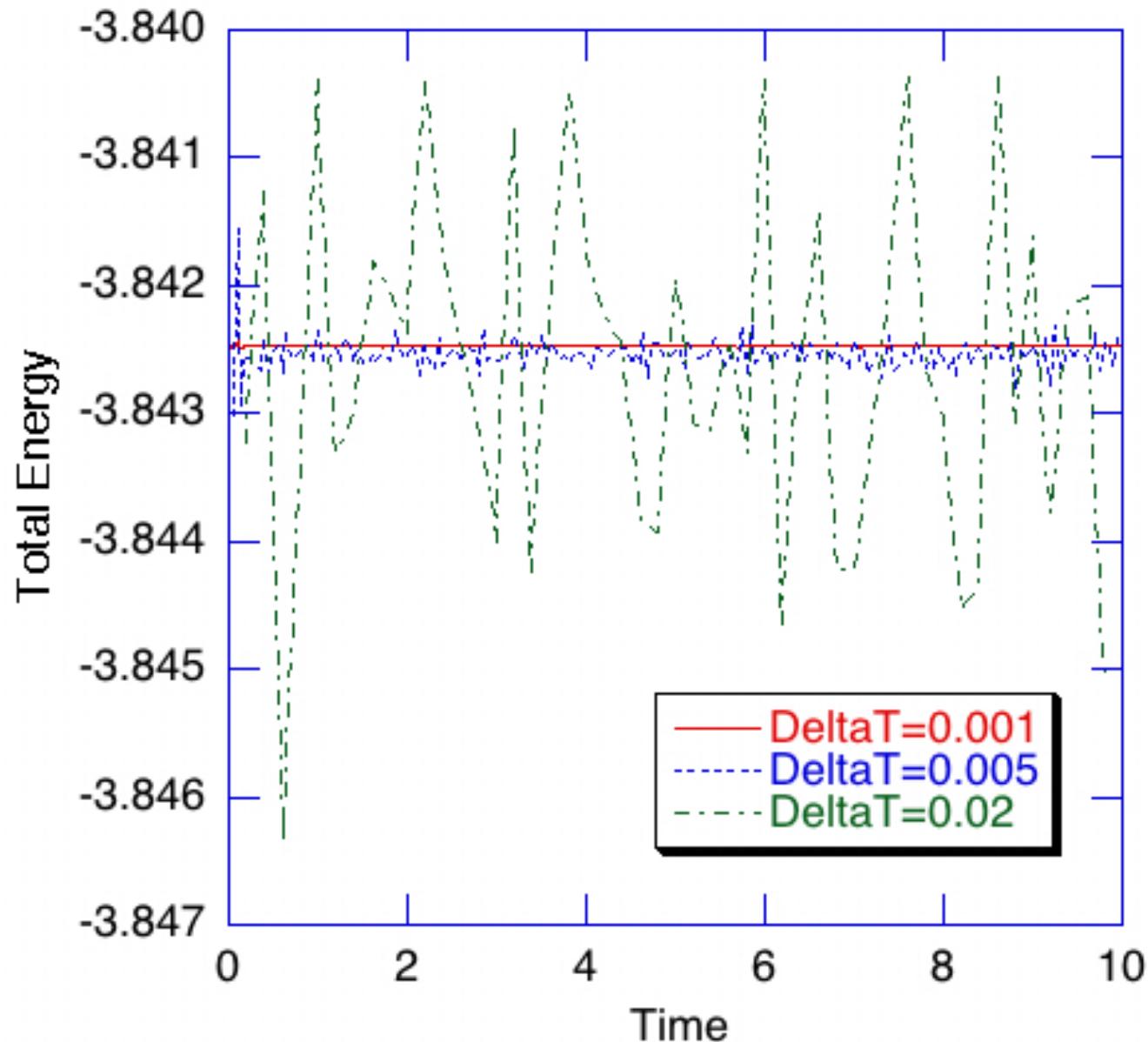
Energy Conservation Revisited

Energy conservation: Velocity-Verlet vs. explicit-Euler algorithms

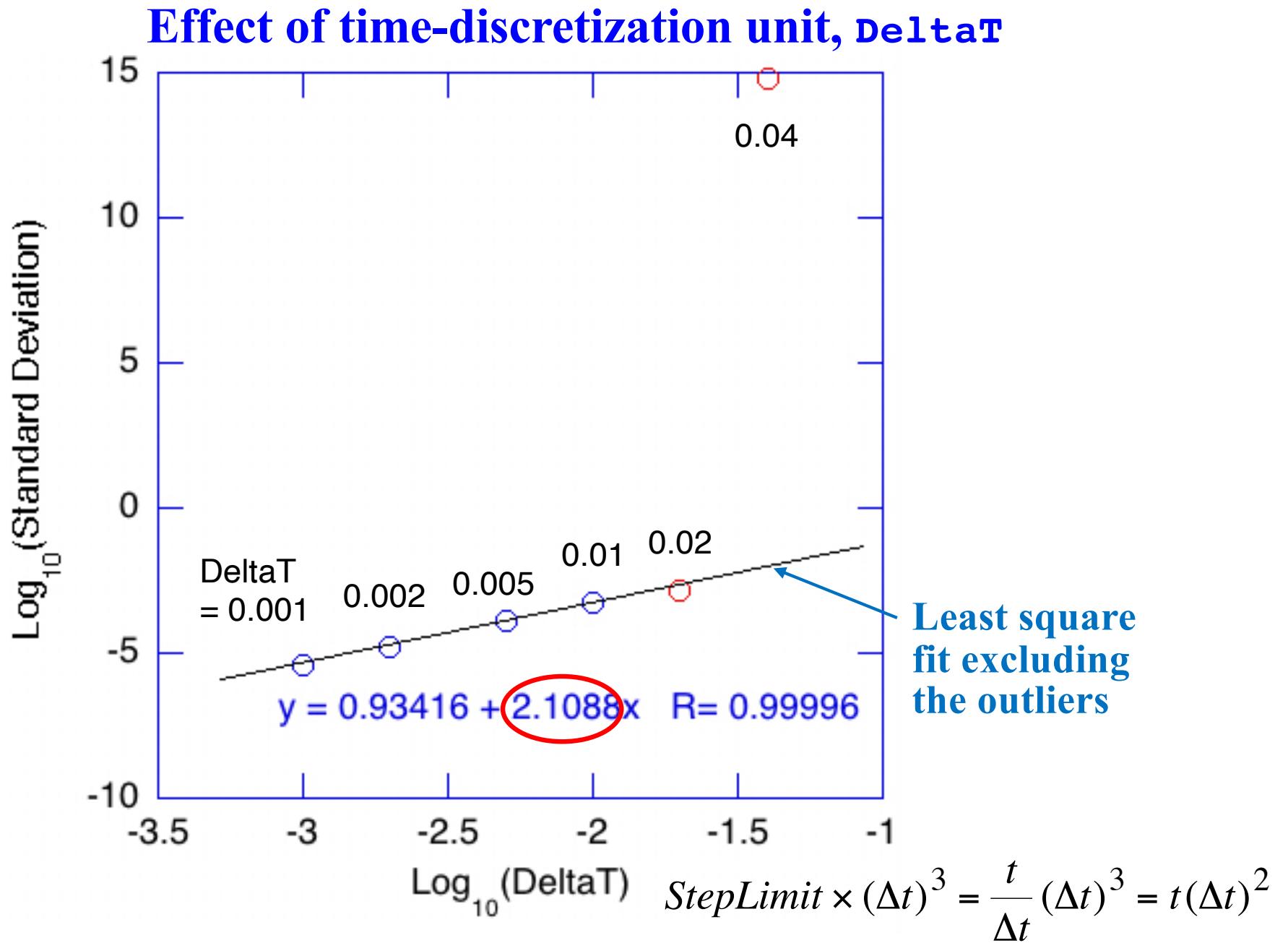


Energy Conservation

Effect of time-discretization unit, Δt



Error in Energy Conservation



Numerical Error in Energy Conservation

- Trajectory error due to time discretization (p. 7 in the [lecture note](#))

$$\begin{cases} \vec{r}_i(t + \Delta) = \vec{r}_i(t) + \vec{v}_i(t)\Delta + \frac{1}{2}\vec{a}_i(t)\Delta^2 + O(\Delta^4) \\ \vec{v}_i(t + \Delta) = \vec{v}_i(t) + \frac{\vec{a}_i(t) + \vec{a}_i(t + \Delta)}{2}\Delta + O(\Delta^3) \end{cases}$$

*cf. Trotter expansion
Tuckerman et al.,
[J. Chem. Phys. 97, 1990 \('92\)](#)*

- For $t = n\Delta$

$$\begin{cases} \Delta\vec{r}_i = O\left(n\Delta^4 = \frac{t}{\Delta}\Delta^4 = t\Delta^3\right) = O(\Delta^3) \\ \Delta\vec{v}_i = O\left(n\Delta^3 = \frac{t}{\Delta}\Delta^3 = t\Delta^2\right) = O(\Delta^2) \end{cases}$$

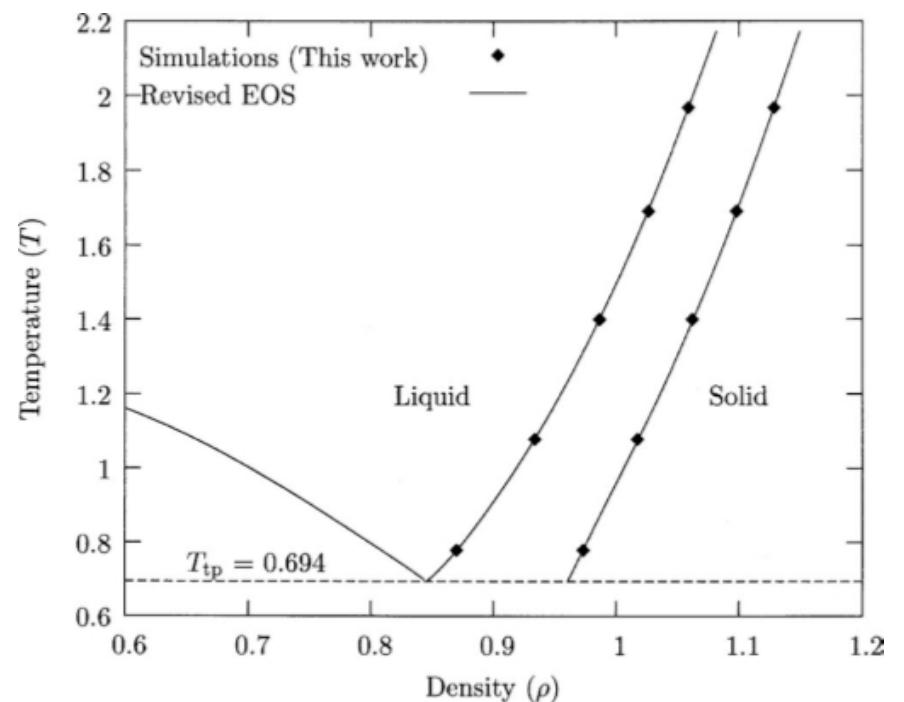
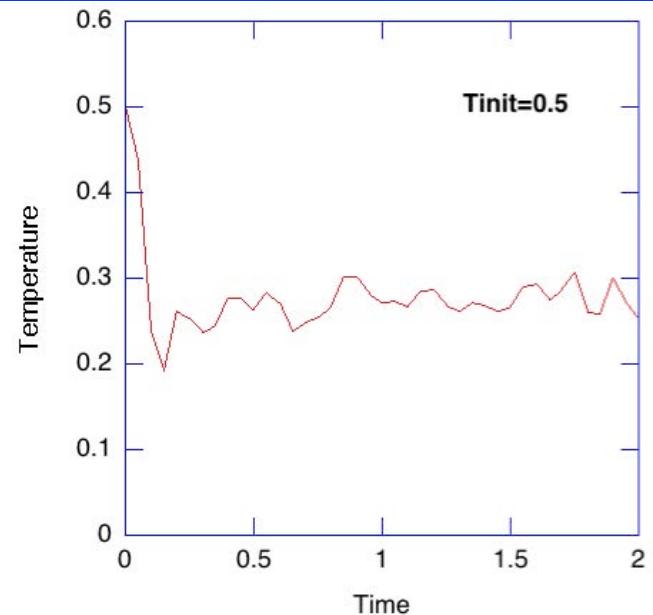
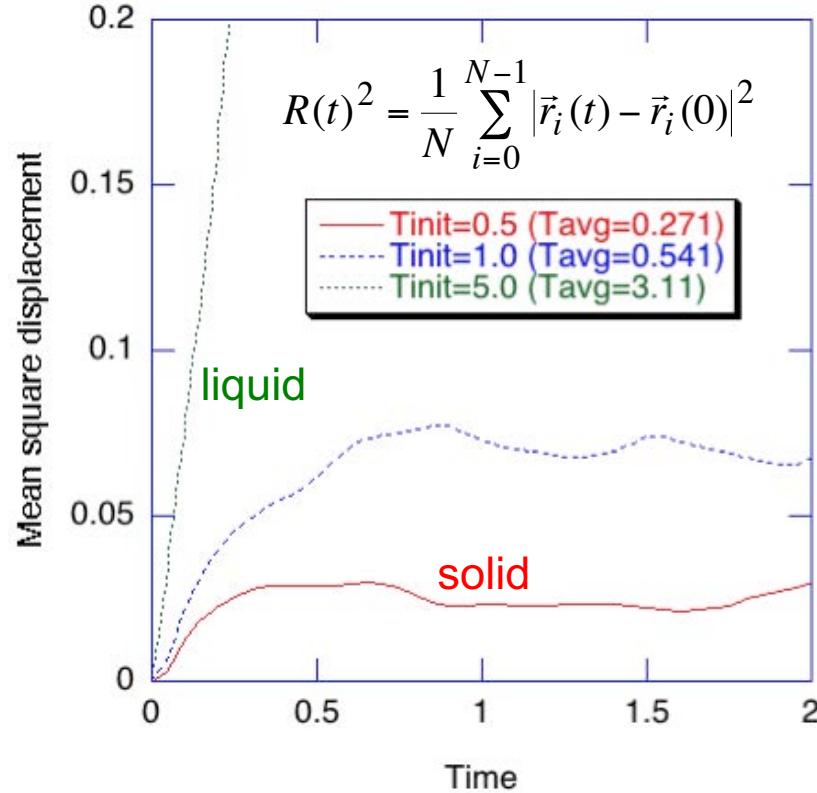
- Energy error (see the note on [error propagation](#))

$$\langle (\Delta E)^2 \rangle = \left| \frac{\partial E}{\partial p} \right|^2 \langle (\Delta p)^2 \rangle + \left| \frac{\partial E}{\partial x} \right|^2 \langle (\Delta x)^2 \rangle$$

$(O(\Delta^2))^2$ $(O(\Delta^3))^2$

- $O(\Delta^2)$ energy error due to time discretization!

Mean Square Displacement



3 3 3
0.8
0.5 | 1.0 | 5.0
0.005
400
10

InitUcell[3]
Density
InitTemp
DeltaT
StepLimit
StepAvg

Where to Go from Here

- MD simulation of materials

MASC 575: *Basics of atomic simulation of materials*

MASC 576: *Molecular dynamics simulations of materials & processes*

- MD simulation on massively parallel computers & visualization

CSCI 596: *Scientific computing and visualization*

<https://aiichironakano.github.io/cs596.html>

CSCI 653: *High performance computing and simulations*

<https://aiichironakano.github.io/cs653.html>

- Textbooks on advanced MD techniques

Understanding Molecular Simulation, 2nd Ed., D. Frenkel & B. Smit
(Academic Press, '01)

Computer Simulation of Liquids, 2nd Ed., M. P. Allen & D. J.
Tildesley (Oxford University Press, '17)