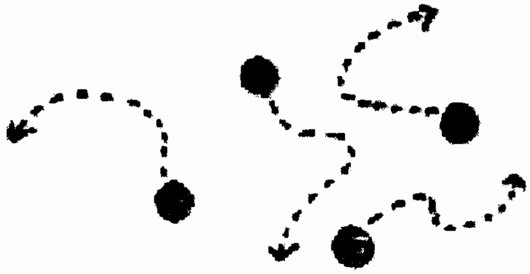
THEORY: MOLECULAR DYNAMICS



- All atoms are moving
- Forces between atoms are complicated functions of time

- Analytical solution of equations of motion to give $x(t)$ and $V(t)$ is impossible.

- Numerical solution is trivially easy.

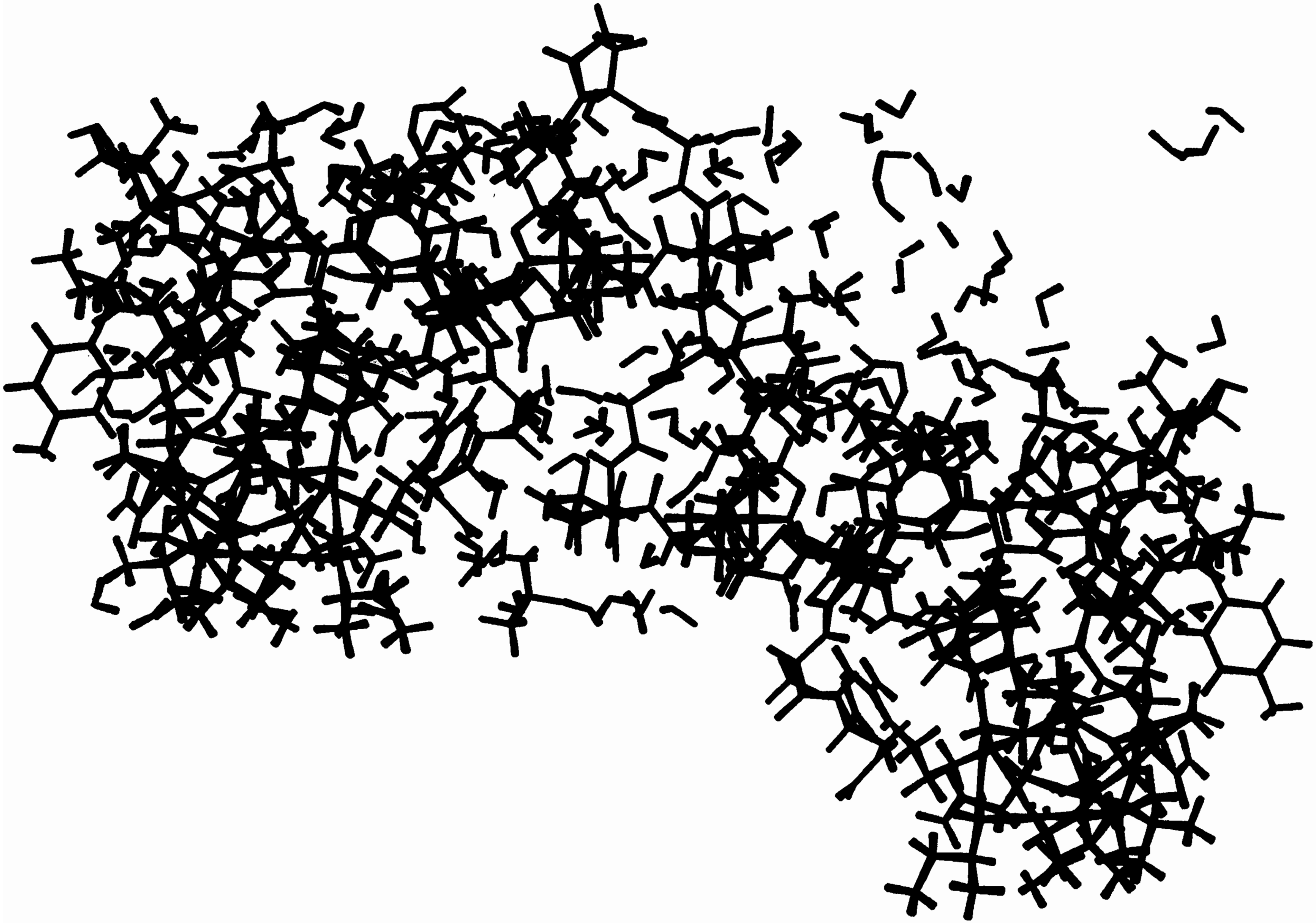
- Atoms move for very short times of 10^{-15} seconds or 0.001 picoseconds (ps)

$$\underset{\uparrow \text{ new position}}{x}(t+\Delta t) = \underset{\uparrow \text{ old position}}{x}(t) + \overset{\text{old velocity } \downarrow}{V}(t)\Delta t + [4a(t) \underset{\uparrow \text{ accelle rations}}{-} a(t-\Delta t)]\,\Delta t^2/6$$

$$v\underset{\uparrow \text{ new velocity}}{(}t+\Delta t) = V\underset{\uparrow \text{ old velocity}}{(}t) + [2a(t+\Delta t) + \underset{\uparrow \text{ accellerations}}{5}a(t) - a(t-\Delta t)]\,\Delta t/6$$

$$\underset{\text{kinetic energy}}{U_{\text{kinetic}}} = \frac{1}{2}\sum \underset{\substack{\uparrow \\ \text{atomic} \\ \text{masses}}}{m_i}\, \underset{\substack{\uparrow \\ \text{velocities}}}{V_i(t)}^2 = \frac{1}{2}\underset{\substack{\uparrow \\ \text{number of} \\ \text{coordinates}}}{n}\, \overset{\substack{\text{Boltzmann constant} \\ \downarrow}}{k_B}\, \underset{\substack{\uparrow \\ \text{Temperature}}}{T}$$

- Total Energy ($U_{\text{potential}} + U_{\text{kinetic}}$) must not change during simulation.



CRAMBIN (UNIT CELL; 2 CRAMBIN, 4 ETOH, 170 WATER)

APPLICATIONS OF MD

1. OBSERVE SHORT TIME SCALE THERMAL MOTIONS

2. TIME CORRELATION FUNCTIONS

3. ACTIVATED PROCESSES

4. ACCURACY OF POTENTIAL FUNCTIONS

5. CONFORMATIONAL SEARCHES

6. LIGAND BINDING, etc. VIA THERMODYNAMIC PERTURBATION METHOD

MOLECULAR DYNAMICS

$$d^2x/dt^2 = a_0$$

$$dx/dt = V_0 + a_0 t$$

$$x = x_0 + V_0 t + \frac{1}{2} a_0 t^2$$

SO, FROM $t=0$ TO $t=\Delta t$

$$V_1 = V_0 + a_0 \Delta t$$

$$x_1 = x_0 + V_0 \Delta t + \frac{1}{2} a_0 \Delta t^2$$

OR, IN GENERAL

$$V_{N+1} = V_N + a_N \Delta t$$

$$x_{N+1} = x_N + V_N \Delta t + \frac{1}{2} a_N \Delta t^2$$

ALSO $V_{N-1/2}$ IS AVERAGE VELOCITY FROM x_{N-1} TO x_N,

$$V_{N-1/2} = (x_N - x_{N-1})/\Delta t$$

THEN ASSUMING $a_{N-1/2} = a_N$,

$$V_N = V_{N-1/2} + a_N(\Delta t/2)$$

$$= (x_N - x_{N-1})/\Delta t + a_N(\Delta t/2)$$

SUBSTITUTION GIVES,

$$x_{N+1} = x_N + V_N \Delta t + \frac{1}{2} a_N \Delta t^2$$

$$= \ldots.$$

$$= 2x_N - x_{N-1} + a_N \Delta t^2$$

AND BY AVERAGE AS ABOVE,

$$V_N = (x_{N+1} - x_{N-1})/2\Delta t$$

LOUP VERLET (1967)

$$x_{N+1} = 2x_N - x_{N-1} + a_N \Delta t^2$$

$$v_N = (x_{N+1} - x_{N-1})/2\Delta t$$

D. BEEMAN (1976)

$$x_{N+1} = x_N + v_N \Delta t + \frac{1}{6}(4a_N - a_{N-1})\Delta t^2$$

$$v_{N+1} = v_N + \frac{1}{6}(2a_{N+1} + 5a_N - a_{N-1})\Delta t$$

ALSO, LEAPFROG METHOD

& THE "GEAR" METHODS

Appeal to Sir Issac,

$$-dE/dx = m \cdot d^2x/dt^2$$

Then just start at x_0, V_0, a_0 and solve Verlet or Beeman recursions....

– But: Where do we start?

How do we control the temperature?

How do we choose Δt?

Start "near" native structure...

FOR N ATOMS,

$$\frac{3}{2} N K_B T = \frac{1}{2} \sum_1^N m_i v_i^2$$

$$= \text{KINETIC ENERGY}$$

SO WE INCREASE TEMP BY ADDING KINETIC ENERGY via VELOCITY INCREASES

BUT HOW DO WE INCREASE VELOCITIES?

→ RANDOM EQUILIBRATION!!

- ASSIGN FROM GAUSSIAN
- RANDOM p-NEUTRAL INCREASE

HOW FAST IS FAST?

ANSWER: 100 m/sec

SO HOW BIG IS Δt?

ANSWER: $\sim 10^{-15}$ sec

10^{-15} = FEMTO

10^{-12} = PICO

- -

WE HAVE OUTLINED "CLASSICAL" DYNAMICS; CONSTANT TEMP, CONSTANT PRESSURE VERSIONS ALSO EXIST....

4.4.1 Verlet's Algorithm

The simplest finite-difference method that has been widely used in molecular dynamics is a third-order Störmer algorithm, first used by Verlet [4] and known to simulators as Verlet's method. The algorithm is a combination of two Taylor expansions, combined as follows. First write the Taylor series for position from time t forward to $t + \Delta t$:

$$x(t+\Delta t) = x(t) + \frac{dx(t)}{dt}\Delta t + \frac{1}{2}\frac{d^2x(t)}{dt^2}\Delta t^2 + \frac{1}{3!}\frac{d^3x(t)}{dt^3}\Delta t^3 + O(\Delta t^4) \tag{4.34}$$

Then write the Taylor series from t backward to $t - \Delta t$:

$$x(t-\Delta t) = x(t) - \frac{dx(t)}{dt}\Delta t + \frac{1}{2}\frac{d^2x(t)}{dt^2}\Delta t^2 - \frac{1}{3!}\frac{d^3x(t)}{dt^3}\Delta t^3 + O(\Delta t^4) \tag{4.35}$$

Adding these two expansions eliminates all odd-order terms, leaving

$$x(t+\Delta t) = 2x(t) - x(t-\Delta t) + \frac{d^2x(t)}{dt^2}\Delta t^2 + O(\Delta t^4) \tag{4.36}$$

This is Verlet's algorithm for positions. It has a local truncation error that varies as $(\Delta t)^4$ and hence is third order, even though it contains no third-order derivatives. Nor does (4.36) for positions involve any function of the velocities; the acceleration in (4.36) is, of course, obtained from the intermolecular forces and Newton's second law. To estimate velocities, practitioners have contrived various schemes, one being an estimate for the velocity at the half-step:

$$v\left(t + \tfrac{1}{2}\Delta t\right) \approx \frac{x(t+\Delta t) - x(t)}{\Delta t} \tag{4.37}$$

Verlet himself used the first-order central difference estimator

$$v(t) \approx \frac{x(t+\Delta t) - x(t-\Delta t)}{2\Delta t} \tag{4.38}$$

Verlet's algorithm is a two-step method because it estimates $x(t+\Delta t)$ from the current position $x(t)$ and the previous position $x(t-\Delta t)$. Therefore it is not self-starting: initial positions $x(0)$ and velocities $v(0)$ are not sufficient to begin a calculation, and something special must be done at $t = 0$ (say, a backward Euler method) to get $x(-\Delta t)$.

Modifications to the basic Verlet scheme have been proposed to tackle these deficiencies. One of these is a so-called half-step 'leap-frog' scheme [Hockney 1970; Potter 1972, Chapter 5]. The origin of the name becomes apparent when we write the algorithm down:

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \delta t\, \mathbf{v}(t+\tfrac{1}{2}\delta t) \tag{3.17a}$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t) = \mathbf{v}(t-\tfrac{1}{2}\delta t) + \delta t\, \mathbf{a}(t). \tag{3.17b}$$

The stored quantities are the current positions $\mathbf{r}(t)$ and accelerations $\mathbf{a}(t)$ together with the mid-step velocities $\mathbf{v}(t - 1/2\delta t)$. The velocity equation (3.17b) is implemented first, and the velocities leap over the coordinates to give the next mid-step values $\mathbf{v}(t + 1/2\delta t)$. During this step, the current velocities may be calculated

$$\mathbf{v}(t) = \tfrac{1}{2}(\mathbf{v}(t+\tfrac{1}{2}\delta t) + \mathbf{v}(t-\tfrac{1}{2}\delta t)). \tag{3.18}$$

This is necessary so that the energy ($\mathscr{H} = \mathscr{K} + \mathscr{V}$) at time t can be calculated, as well as any other quantities that require positions and velocities at the same instant. Following this, eqn (3.17a) is used to propel the positions once more ahead of the velocities. After this, the new accelerations may be evaluated ready for the next step. This is illustrated in Fig. 3.2. Elimination of the velocities from these equations shows that the method is algebraically equivalent to Verlet's algorithm. There are some advantages in programming eqns (3.17)–(3.18), however, since the velocities (admittedly not at time t) appear explicitly [Fincham and Heyes 1982]; for example, adjusting the simulation energy is usually achieved by appropriately scaling the velocities. Numerical benefits derive from the fact that at no stage do we take the difference of two large quantities to obtain a small one; this minimizes loss of precision on a computer. If there is a desperate need to conserve storage space, the accelerations may be directly accumulated onto the velocities, thus making the overall requirements of order $6N$ words [Fincham and Heyes 1982]. The cost is that eqn (3.18) may no longer be used, and it becomes necessary to estimate the kinetic energy at time t from the known mid-step values. An example of the leap-frog technique in use in a low-storage program coded in FORTRAN and in BASIC (for a microcomputer) is given in F.3. Finally, we note that the leap-frog approach may be applied to other algorithms as well as Verlet's [Fincham and Heyes 1982].

As eqn (3.18) shows, leap-frog methods still do not handle the velocities in a completely satisfactory manner. A Verlet-equivalent algorithm which does store positions, velocities, and accelerations all at the same time t, and which minimizes round-off error, has recently been proposed [Swope, Andersen, Berens, and Wilson 1982]. This 'velocity Verlet' algorithm takes the form

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t)+\delta t\,\mathbf{v}(t)+\tfrac{1}{2}\delta t^2\mathbf{a}(t) \tag{3.19a}$$

$$\mathbf{v}(t+\delta t) = \mathbf{v}(t)+\tfrac{1}{2}\delta t\,[\mathbf{a}(t)+\mathbf{a}(t+\delta t)]\,. \tag{3.19b}$$

Again, the Verlet algorithm may be recovered by eliminating the velocities. In this form, the method resembles a three-value predictor–corrector algorithm (see Appendix E), where the position corrector coefficient is zero [van Gunsteren and Berendsen 1977]. The algorithm only requires storage of **r**, **v**, and **a**. Although it is not implemented in exactly the form of a Gear predictor–corrector, it does involve two stages, with a force evaluation in between. Firstly, the new positions at time $t+\delta t$ are calculated using eqn (3.19a), and the velocities at mid-step are computed using

$$\mathbf{v}(t+\tfrac{1}{2}\delta t) = \mathbf{v}(t)+\tfrac{1}{2}\delta t\,\mathbf{a}(t)\,. \tag{3.20}$$

The forces and accelerations at time $t+\delta t$ are then computed, and the velocity move completed

$$\mathbf{v}(t+\delta t) = \mathbf{v}(t+\tfrac{1}{2}\delta t)+\tfrac{1}{2}\,\delta t\,\mathbf{a}(t+\delta t). \tag{3.21}$$

At this point, the kinetic energy at time $t+\delta t$ is available. The potential energy at this time will have been evaluated in the force loop. The whole process is shown in Fig. 3.2. The method once more uses $9N$ words of storage, and its numerical stability, convenience, and simplicity make it perhaps the most attractive proposed to date. The code for the velocity version of Verlet's method is a straightforward transcription of eqns (3.19)–(3.21) (see program F.4).

Before we leave Verlet, we should mention the investigation by Beeman [1976] of several algorithms, one of which reduces to eqn (3.14) when the velocities are eliminated [Sangster and Dixon 1976; Hockney and Eastwood 1981]. The algorithm is

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t)+\delta t\,\mathbf{v}(t)+\tfrac{2}{3}\delta t^2\mathbf{a}(t)-\tfrac{1}{6}\delta t^2\mathbf{a}(t-\delta t) \tag{3.22a}$$

$$\mathbf{v}(t+\delta t) = \mathbf{v}(t)+\tfrac{1}{3}\delta t\,\mathbf{a}(t+\delta t)+\tfrac{5}{6}\delta t\,\mathbf{a}(t)-\tfrac{1}{6}\delta t\,\mathbf{a}(t-\delta t)\,. \tag{3.22b}$$

The method stores $\mathbf{r}(t)$, $\mathbf{v}(t)$, $\mathbf{a}(t)$, and $\mathbf{a}(t-\delta t)$. Offsetting the complexity of these formulae, and the need to store the 'old' accelerations, is a more accurate equation for the velocities than eqn (3.16), and consequently an apparent improvement in energy conservation. However, once again, all the methods described in this section are essentially equivalent in that they have identical global errors and in fact generate identical position trajectories.

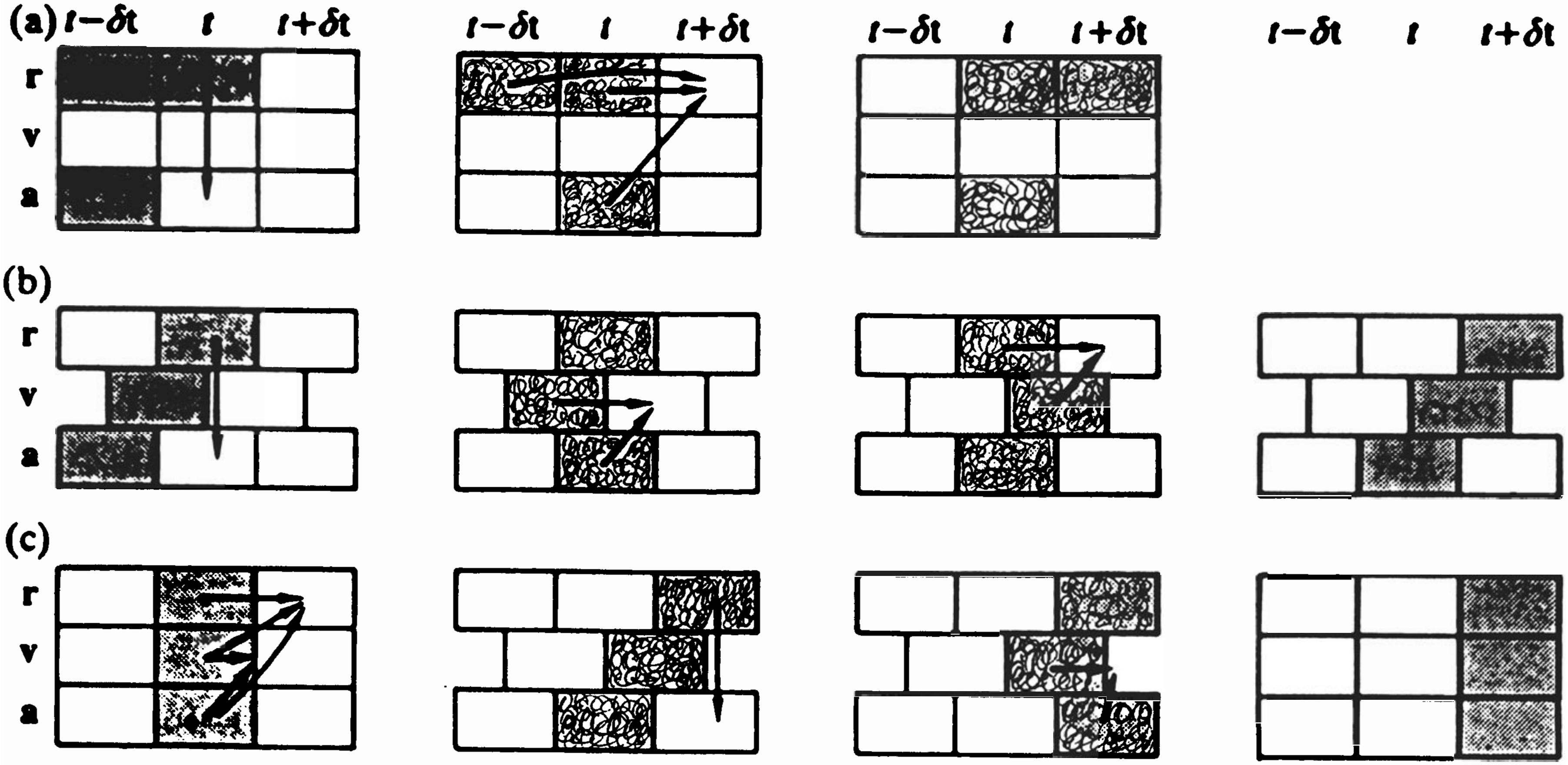


Fig. 3.2 Various forms of the Verlet algorithm. (a) Verlet's original method. (b) The leap-frog form. (c) The velocity form. We show successive steps in the implementation of each algorithm. In each case, the stored variables are in grey boxes.

STATISTICAL MECHANICS :

MICROCANONICAL ENSEMBLE

$$\Omega\,(N, V, E)$$

$$S = k \ln \Omega$$

CANONICAL ENSEMBLE

$$Q\,(N, V, T)$$

$$A = -kT \ln Q$$

ISOTHERMAL-ISOBARIC ENSEMBLE

$$\Delta\,(N, T, P)$$

$$G = -kT \ln \Delta$$

$$\mu = \left\{1 + \frac{\Delta t}{\tau}\,\beta\left[P(t) - P_0\right]\right\}^{1/3}$$

$$\lambda = \left\{1 + \frac{\Delta t}{\tau}\left[T_0/T(t) - 1\right]\right\}^{1/2}$$

Table 3.1. *Typical features of some internal motions of proteins and nucleic acids*

Motion	Spatial extent (nm)	Amplitude (nm)	Log_{10} of characteristic time (s)
Relative vibration of bonded atoms	0.2 to 0.5	0.001 to 0.01	−14 to −13
Longitudinal motions of bases in double helices (nucleic acids)	0.5	0.01	−14 to −13
Lateral motions of bases in double helices (nucleic acids)	0.5	0.1	−13 to −12
Global stretching (nucleic acids)	1 to 30	0.03 to 0.3	−13 to −11
Global twisting (nucleic acids)	1 to 30	0.1 to 1.0	−13 to −11
Elastic vibration of globular region	1 to 2	0.005 to 0.05	−12 to −11
Sugar repuckering (nucleic acids)	0.5	0.2	−12 to −9
Rotation of sidechains at surface (protein)	0.5 to 1	0.5 to 1	−11 to −10
Torsional libration of buried groups	0.5 to 1	0.05	−11 to −9
Relative motion of different globular regions (hinge bending)	1 to 2	0.1 to 0.5	−11 to −7
Global bending (nucleic acids)	10 to 100	5 to 20	−10 to −7
Rotation of medium-sized sidechains in interior (protein)	0.5	0.5	−4 to 0
Allosteric transitions	0.5 to 4	0.1 to 0.5	−5 to 0
Local denaturation	0.5 to 1	0.5 to 1	−5 to +1

Activated Dynamics:

$$k = (1/2)\, K \langle |\varepsilon'| \rangle \left[\rho(\varepsilon^{\ddagger}) / \int \rho(\varepsilon)\, d\varepsilon \right]$$

ADD UMBRELLA TERM,

$$U(\varepsilon) = (k/2)(\varepsilon - \varepsilon^{\ddagger})^2$$

Start Usual MD From Several Points Near Transition State

Brownian (Stochastic) Dynamics:

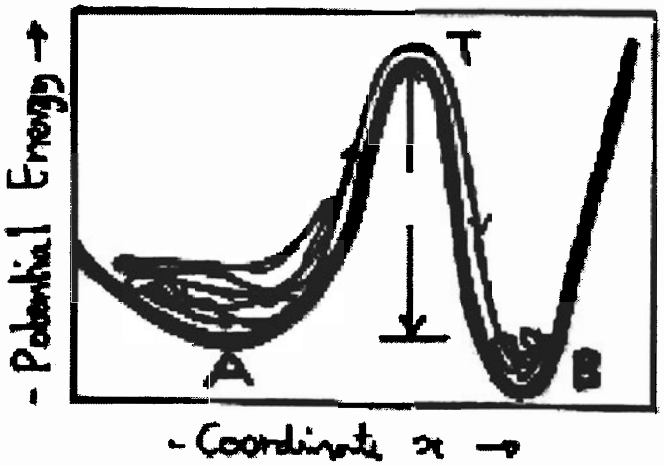
Langevin Equations of Motion,

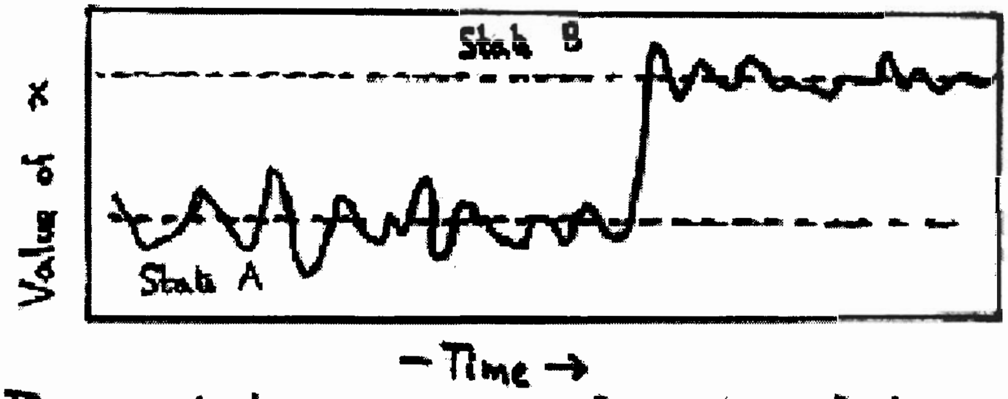
$$m \frac{d^2x}{dt^2} = F(x) - \underset{\substack{\uparrow \\ \text{FRICTION}}}{f} \frac{dx}{dt} + \underset{\substack{\uparrow \\ \text{RANDOM} \\ \text{FORCE}}}{R(t)}$$

Models Behavior of An Ensemble of Diffusing Molecules

CROSSING ENERGY BARRIERS

Basic theory of changes of state, chemical reaction. Look at a time-course. Wait. Rates. Jump.





The actual transition from A to B is very quick (few pico seconds). What takes time is the waiting. Theory gives the average wait time in state A as

Boltzman constant in kcal/mol

$$\tau_{A\to B} = \left(\frac{h}{kT}\right) \exp\left(+ (U_T - U_A)/RT \right)$$

Planck's constant

Boltzmann constant

$\left(\frac{h}{kT}\right) \approx 0.16$ picoseconds at $T = 300°K$ (27°C).

(remember as $h\nu_0 = kT$, $\nu_0 = 208$ cm^{-1}, $\tau_0 = 0.16$ ps)

Table 12.1. Selected biomolecular sampling methods. Continuum solvation includes empirical constructs, generalized Born models, stochastic dynamics, or Poisson Boltzmann solutions, as discussed in Chapter 9. See Figure 12.1 and Figure 13.17 for illustrations of these techniques.

Method	Pros	Cons	CPU
• *Molecular Dynamics* (MD) [37, 38, 834]	continuous motion, experimental bridge between structures and macroscopic kinetic data	expensive; short timespan	high
• *Targeted MD* (TMD) [835]	connection between two states; useful for ruling out steric clashes and suggesting high barriers	not necessarily physical	moderate
• *Stochastic Path Approach* [836]	high-frequency motion filtering; approximate long-time trajectories	expensive (global optimization of entire trajectory)	high
• *Continuum Solvation* [703, 712, 704, 711, 701, 702]	mean-force potential approximates environment and reduces model's cost; useful information on ionic atmosphere and intermolecular associations	approximate	high (if repeated in time)
• *Brownian Dynamics* (BD) [703, 269]	large-scale and long-time motion	approximate hydrodynamics; limited to systems with small inertia	moderate
• *Monte Carlo* (MC) [833]	large-scale sampling; useful statistics	move definitions are difficult; unphysical paths	low
• *Minimization* [758]	valuable equilibria information; experimental constraints can be incorporated	no dynamic information	low

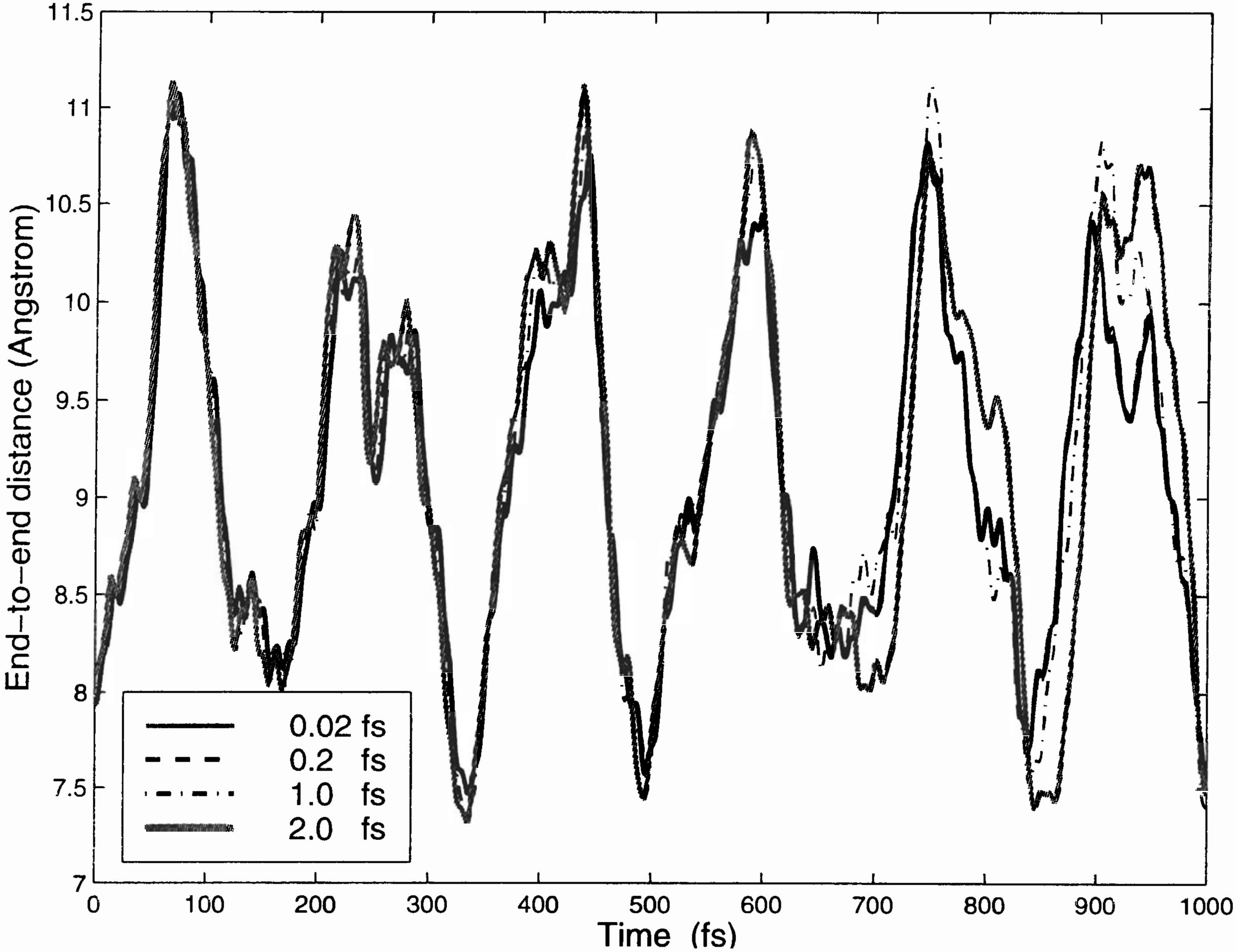


Figure 12.4. Time evolution of the end-to-end distance of butane for different timesteps.

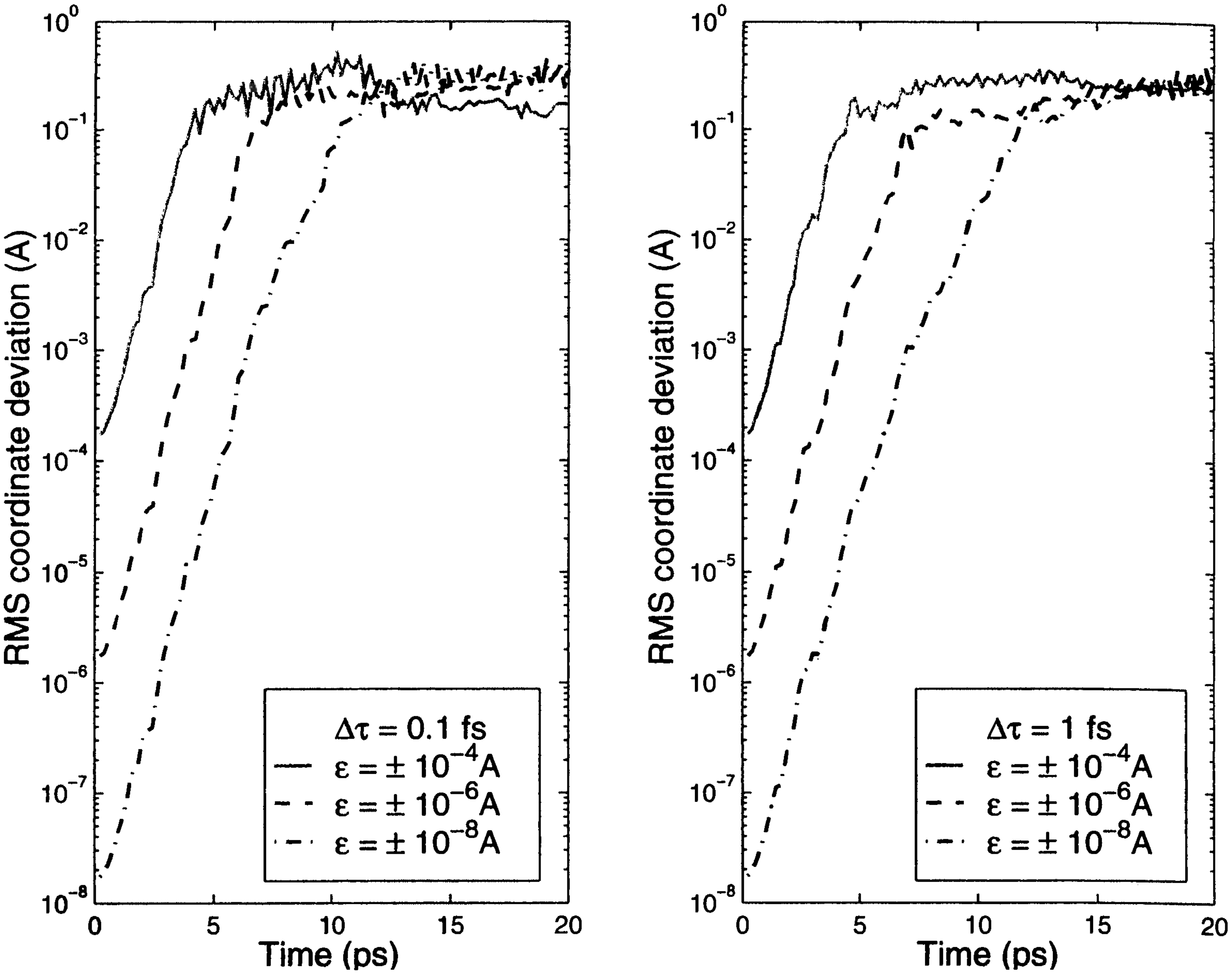


Figure 12.3. The rapid divergence of dynamic trajectories for four water molecules differing slightly in initial conditions.

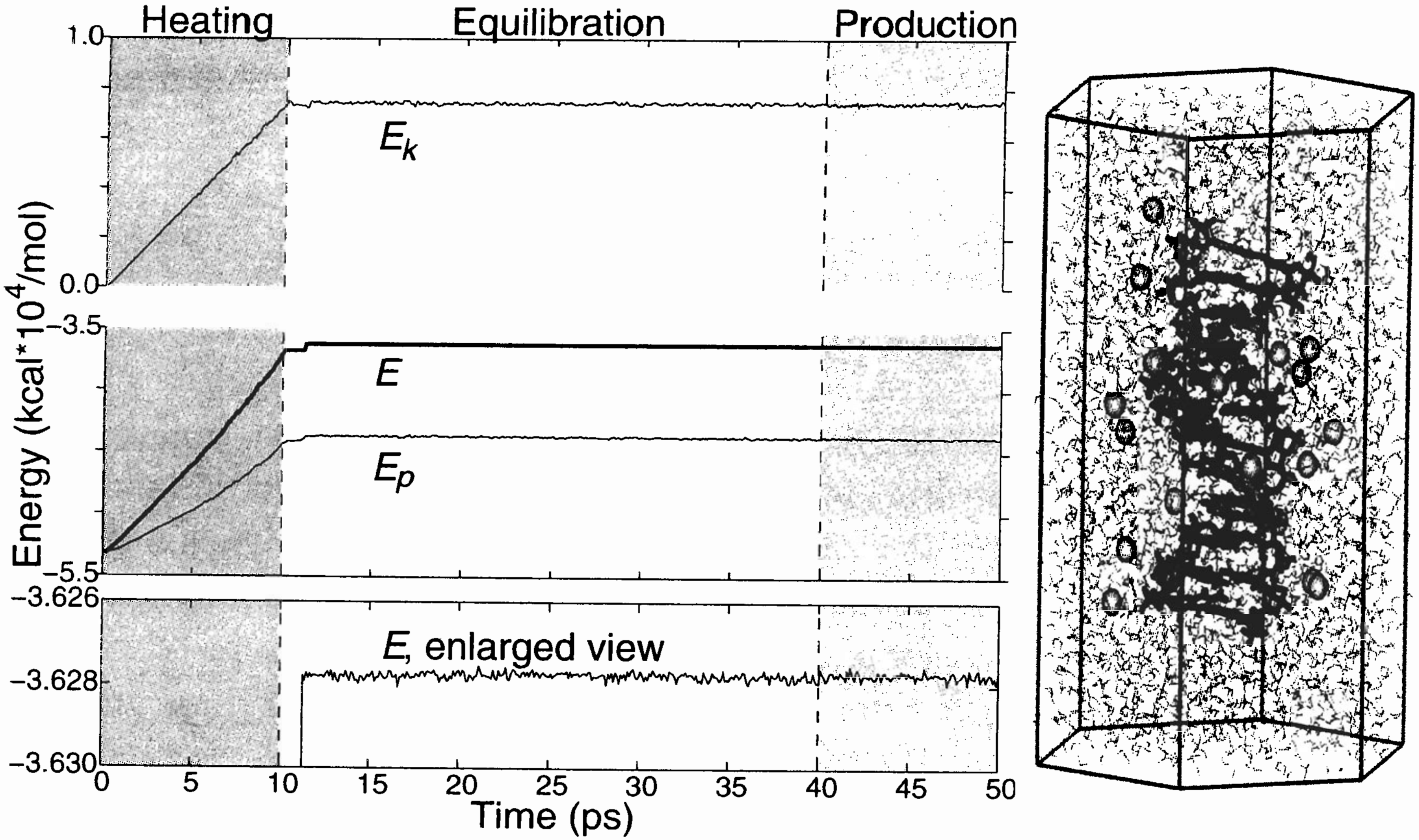


Figure 12.2. Heating and equilibration of a hydrated DNA dodecamer system of 12389 atoms in a hexagonal-prism periodic domain.

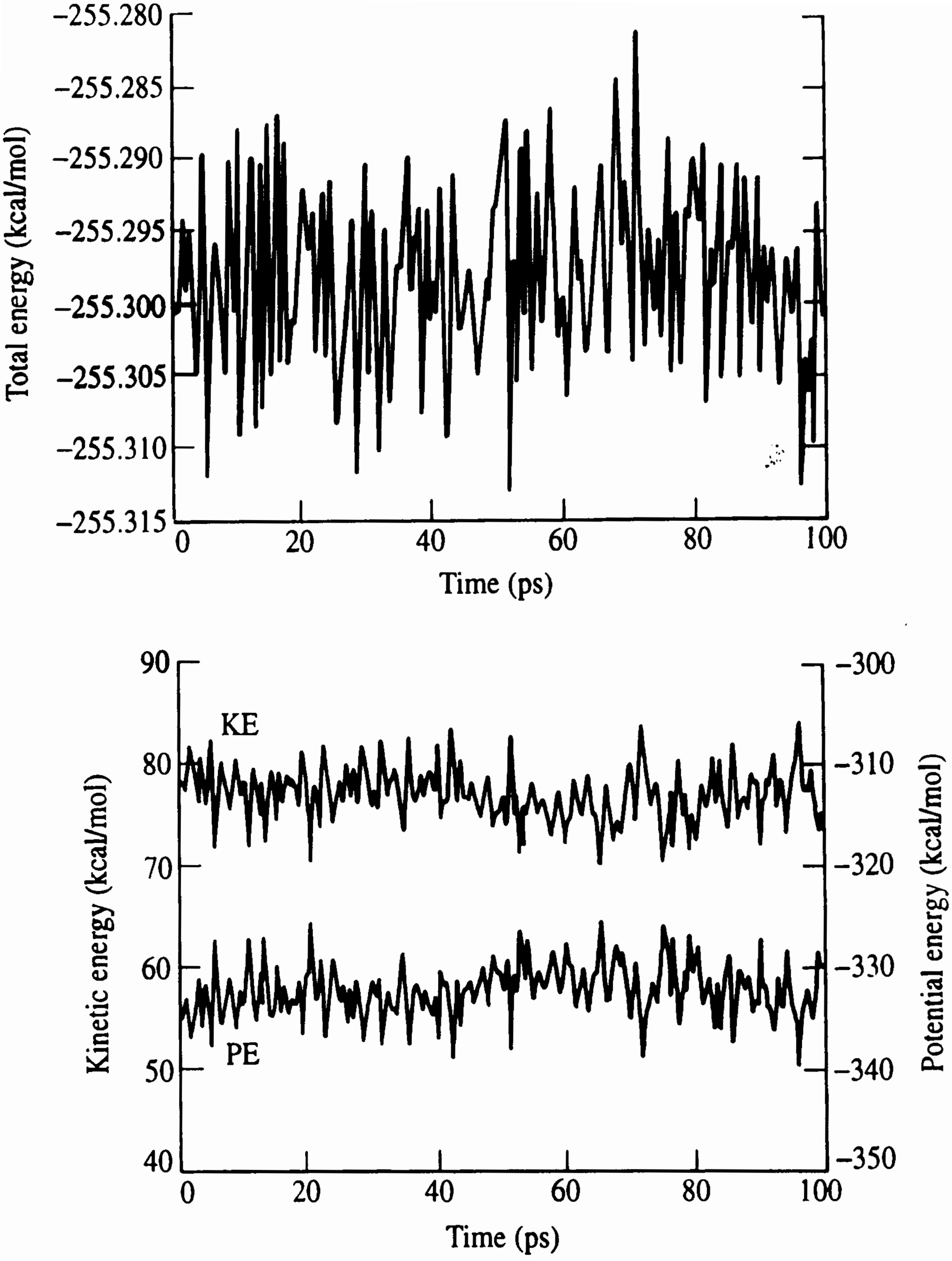


Fig. 6.3 Variation in total energy versus time for the production phase of a molecular dynamics simulation of 256 argon atoms at a temperature of 100K and a density of 1.396 g cm^{-3}(top). The time step was 10 fs and the equations of motion were integrated using the velocity Verlet algorithm. The variations in the kinetic and potential energies are also shown. The graphs have different scales.