A Hands-On Introduction to Molecular Dynamics

Vincent E. Lamberti,*† Lloyd D. Fosdick, and Elizabeth R. Jessup
Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309; **lambertive@y12.doe.gov*

Carolyn J. C. Schauble
Department of Computer Science, Colorado State University, Fort Collins, CO 80523

Background

Over the past few decades, the computer experiment has become an invaluable tool for both chemistry and chemical engineering. The power of this new technique derives largely from the fact that it occupies a unique position between the traditional regimes of theory and laboratory experiment (*1*). Computer experiments can test new theoretical ideas, such as improved models of intermolecular forces, on systems far too complex for manual calculation. At the same time, computer simulations can approximate laboratory experiments to the extent that computer results can sometimes be compared with, and lend microscopic insight into, real experimental results. Furthermore, computer experiments can provide (virtual) access to extreme conditions that are not easily reproduced in the real laboratory, such as the pressures and temperatures at the earth's core.

Two different kinds of computer experiments based upon statistical mechanics are especially useful for modeling the dynamics of atoms and molecules. Both approaches have been applied successfully to systems as simple as an atomic fluid like liquid argon and as complicated as a protein composed of hundreds of amino acids. The first method, invented by B. J. Alder and T. W. Wainwright (*2, 3*) at Lawrence Livermore National Laboratory, is known as *molecular dynamics* (MD) *simulation* (*1, 4, 5*). It follows the classical time evolution—that is, the positions and velocities after each of a series of time steps—of a collection of virtual interacting particles by means of their integrated equations of motion. The dynamics are derived from Newton's second law,

$$\boldsymbol{a}_i = \frac{\mathrm{d}^2\boldsymbol{r}_i}{\mathrm{d}t^2} = \frac{1}{m_i}\boldsymbol{F}_i \qquad (1)$$

where the acceleration $\boldsymbol{a}_i$ is produced by an instantaneous force $\boldsymbol{F}_i$ acting on particle i with mass m_i and instantaneous position $\boldsymbol{r}_i$. With appropriate forces, MD experiments can simulate particles traveling through space, colliding with one another, oscillating in concert, and even evaporating from a free surface. During an MD experiment, the microscopic trajectories of a great many particles are averaged using the rules of statistical physics to give macroscopic quantities such as diffusion coefficients, temperatures of phase transitions, and the instantaneous potential and kinetic energies of the system.

The second method relies upon an element of chance rather than the classical equations of motion and is known as the *Monte Carlo* (MC) technique (*5*). In each step of an MC simulation, a randomly chosen particle is moved to a new randomly chosen location. If the new configuration has a lower energy than the previous one, the move is immediately accepted; otherwise, the new configuration is subjected to further statistical tests. If the move is ultimately rejected, the system is returned to its previous state. This process is repeated until the changes in energy become vanishingly small, at which point the system is deemed to have reached thermodynamic equilibrium. As in the molecular dynamics technique, the microscopic states of the particles are averaged to generate macroscopic equilibrium properties such as the internal energy and the entropy.

Overview

In this article, we present an introduction to both the chemical and the computational aspects of the molecular dynamics technique. Using just a few elementary ideas from classical mechanics and numerical analysis, we go through the steps required for the design and analysis of a simple molecular dynamics simulation. We use linear chains of interacting particles as examples, since the mathematical models describing these systems are relatively uncomplicated but still produce interesting dynamics. Along the way, we emphasize the compromises that arise in any computer-based experiment because of floating-point arithmetic, finite memory, and limitations in processor performance. Saiz and Tarazona have described the principles of molecular dynamics in this *Journal* and applied the technique to an isolated particle, two interacting particles, and the water molecule (*6*).

This paper is part of a larger project concerned with the development of instructional materials and laboratory facilities for an undergraduate course in high-performance scientific computing and scientific visualization. The course materials are available through anonymous FTP from *http://www.cs.colorado.edu/95-96/courses/materials.hpsc.html* and include a number of tutorials and reference manuals, a laboratory manual, and software to accompany the laboratory manual.

We begin with a brief description of the meanings of force and potential in a molecular dynamics experiment, with emphasis on the Hooke's law approximation. We then illustrate the derivation of the classical equations of motion for a linear chain of interacting particles by considering the three-particle chain in detail. We introduce two simple methods for numerically integrating the equations of motion, one based on Euler's method for differential equations and the other a more accurate algorithm developed by Verlet. Next, we present a molecular dynamics simulation of the three-particle system, with an analysis of the particle trajectories in terms of the normal modes of vibration of the chain. We conclude with a comparison of the cumulative errors produced by the two methods of numerical integration for the three-particle simulation. As supplemental material, we provide a basic MD implementation using the Euler integration algorithm in both Fortran and C and a few suggested exercises for the code.[W]

Forces and Potentials in Molecular Dynamics

To develop a classical description of our system of interacting particles, we must know the net force acting on each particle during each time step. In general, we assume conservative, two-body forces between the particles. That is, we assume that the total energy of the system is always conserved and that the total force acting on a particle i is equal to the pairwise vector sum of forces between particle i and all other particles $j \neq i$ in the system. For example, the total force acting on particle 3 in a six-particle system is

$$\boldsymbol{F}_3 = \boldsymbol{F}_{31} + \boldsymbol{F}_{32} + \boldsymbol{F}_{34} + \boldsymbol{F}_{35} + \boldsymbol{F}_{36} \qquad (2)$$

Note that the order of subscripts is important here: the symbol $\boldsymbol{F}_{ij}$ refers exclusively to the force on particle i due to particle j.

In an n-particle molecular dynamics simulation, the forces are computed from a potential energy function $V(\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_n)$ that depends upon the coordinates of all n particles. (V is usually called simply the *potential*.) For any particle i,

$$\boldsymbol{F}_i = -\frac{\partial V}{\partial \boldsymbol{r}_i} \qquad (3)$$

Thus, we may equivalently discuss molecular dynamics in terms of either forces or potentials, and the problem of modeling a system is sometimes posed as a search for an appropriate potential for that system (*1*). Given one potential, our virtual particles might represent a dilute noble gas; given another, they might represent covalently bonded atoms in an organic compound. The choice of potential also reflects one of the most important trade-offs in a molecular dynamics experiment: the more realistic and complicated the potential, the greater the demands made on the processor(s) and memory of the computational system.

Rigorously, an interatomic potential should include terms for all of the electron–electron, nucleus–nucleus, and electron–nucleus interactions in the system. In practice, approximate potentials are employed, since formulating a complete potential is an impossible demand for any but the most rudimentary of systems. Fortunately, in some cases, surprisingly insightful results can be obtained with potentials as simple as the *hard-sphere* potential, in which the atoms are represented as perfect spheres with infinitely hard surfaces, and the *Hooke's law* (HL) potential, in which the system is modeled as a collection of masses and springs. The HL potential is especially popular for a first approximation because its visualization in terms of springs provides a natural model for chemical bonds; in addition, it is one of the very few potentials that lead to equations of motion that can be solved in closed form. If $\boldsymbol{r}_{eq}$ designates the equilibrium position of a particle, then the HL potential energy of that particle at any position $\boldsymbol{r}$ is

$$V(\boldsymbol{r}) = \frac{k}{2}\left(\boldsymbol{r} - \boldsymbol{r}_{eq}\right)^2 + V_{min} \qquad (4)$$

where k is known as the *force* or *spring constant* and V_{min} is a constant representing the minimum potential energy. (V_{min} is the constant of integration in eq 3.) We use the Hooke's law potential exclusively in the following discussion.

Figure 1. A linear chain of three interacting particles. In the text we take the line joining the particles as the x axis.

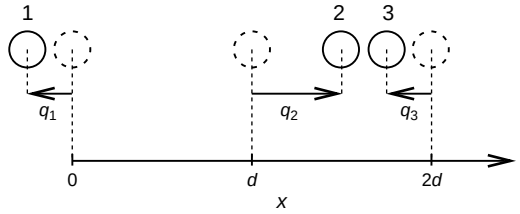

Figure 2. The linear three-particle chain of Figure 1 after the change of variables $q_i = x_i - (i-1)d$, where q_i denotes the position of the ith particle relative to its equilibrium position and d represents the equilibrium separation of any two neighbors. At equilibrium, the positions of particles 1, 2, and 3 are 0, d, and $2d$, respectively. The figure shows the equilibrium positions of the particles in dashed circles and a possible set of nonequilibrium positions in solid circles.

Equations of Motion for a Chain of Interacting Particles

Once the potential for the model system has been chosen, the equations of motion governing the particles of the system must be derived. To illustrate the procedure for a linear chain of particles, we consider in detail the three-particle system shown in Figure 1. We assume that there are no external forces acting upon the system and that each particle interacts with at most two other particles, namely, its nearest neighbors. In addition, we assume that all particles have the same mass m and that all interactions have a common force constant k.

Our use of the Hooke's law potential allows us to picture the interactions between the particles equivalently as a set of springs. The forces of the springs act along a line joining the particles that we take as the x axis; the position of particle i on this axis is denoted as x_i. We number the particles in increasing order from left to right and define a constant d that represents the equilibrium separation of the particles. We say that the system is in equilibrium if the distance between any two neighbors is equal to d. If the distance between two neighbors is less than d, the spring connecting the particles is compressed and a force acts to drive the particles apart; conversely, if the distance between two neighbors is greater than d, the connecting spring is stretched and a force acts to drive the particles together. The total Hooke's law potential energy of this system is

$$V(x_1, x_2, x_3) = \frac{k}{2}\left[(x_2 - x_1 - d)^2 + (x_3 - x_2 - d)^2\right] + V_{min} \qquad (5)$$

It can be easily verified that $V = V_{min}$ when $(x_2 - x_1)$ and $(x_3 - x_2)$ are equal to d.

Following eq 3, we obtain the force acting on each particle i by computing $-\partial V/\partial x_i$:

$$\begin{aligned} F_1(x_1, x_2) &= k(x_2 - x_1 - d) \\ F_2(x_1, x_2, x_3) &= k(x_1 - 2x_2 + x_3) \\ F_3(x_2, x_3) &= k(x_2 - x_3 + d) \end{aligned} \qquad (6)$$

The Newtonian equations of motion can be derived directly from these three relations, but it is usually advantageous to first perform a change of variables. We fix the *equilibrium* position of the first particle at the origin and define q_i to be the displacement of the ith particle from its equilibrium position:

$$q_i = x_i - (i-1)d, \quad i = 1, 2, 3 \tag{7}$$

The equilibrium points of particles 1, 2, and 3 are now 0, d, and $2d$, respectively, and any vanishing q_i means that particle i is at its equilibrium position. The new coordinate system is shown in Figure 2. The forces can now be written in a form that is independent of d

$$\begin{aligned} F_1(q_1, q_2) &= k(q_2 - q_1) \\ F_2(q_1, q_2, q_3) &= k(q_1 - 2q_2 + q_3) \\ F_3(q_2, q_3) &= k(q_2 - q_3) \end{aligned} \tag{8}$$

and substituted into eq 1 to give the final equations of motion:

$$\begin{aligned} m\ddot{q}_1 &= k(q_2 - q_1) \\ m\ddot{q}_2 &= k(q_1 - 2q_2 + q_3) \\ m\ddot{q}_3 &= k(q_2 - q_3) \end{aligned} \tag{9}$$

where we have used $\ddot{q}$ for the second time derivative of q_i. These equations describe oscillatory motions of the particles along the x axis. The closed-form solution for the three-particle system, to which we shall return later, reveals that each q_i is proportional to $\cos(\omega t)$, where the *angular frequency* ω is proportional to $(k/m)^{1/2}$.

We conclude this section by observing that even a model as elementary as ours, with a few minor extensions, can have real applications. By simply providing for unequal particle masses, for example, we can simulate a variegated linear molecule such as CO_2. Alternatively, by treating the first and last particles of the chain as nearest neighbors, we can model n-particle rings like those that occur in allotropes of the catenation-prone chalcogens S and Se (*7*).

Integration of the Equations of Motion

For the great majority of cases, the equations of motion cannot be solved in closed form and must be integrated numerically. The integration is usually performed using a *time-stepping* algorithm (*8*), which generates a solution incrementally in time starting from a set of initial conditions—that is, which uses the positions and velocities of the particles at some time t to compute the positions and velocities at some later time $t + \Delta t$. The time step Δt can be constant or variable, but in either case represents a trade-off between accuracy and efficiency: in order to obtain a faithful picture of the molecular choreography, Δt must be small compared to the characteristic periods of the system, but if it is too small, the computation becomes onerously slow. In general, the best compromise between accuracy and efficiency is found by using simple second-order schemes such as the Verlet algorithm (*8, 10*) (see below) and adjusting the time step appropriately. The initial conditions are usually chosen so that

$$\sum_{i=1}^{n} q_i(0) = 0, \qquad \sum_{i=1}^{n} v_i(0) = 0 \tag{10}$$

where $v_i(t)$ is the velocity of particle i at time t.

The simplest time integration algorithm is based on Euler's method for ordinary differential equations (*9*). In this procedure, the position and velocity of each particle in the chain is updated at the conclusion of each time step as follows, where we have followed custom and used h for Δt:

$$q_i(t+h) \approx q_i(t) + hv_i(t) \tag{11}$$

$$v_i(t+h) \approx v_i(t) + \frac{h}{m} F_i(q_i(t)) \tag{12}$$

Note that these relations tend toward the familiar differential equations

$$\frac{\mathrm{d}q_i}{\mathrm{d}t} = v_i, \qquad \frac{\mathrm{d}v_i}{\mathrm{d}t} = \frac{1}{m} F_i \tag{13}$$

as $h \to 0$. To carry out the calculation, a value for h is chosen, $q_i(h)$ and $v_i(h)$ are computed from the initial ($t = 0$) conditions, $q_i(2h)$ and $v_i(2h)$ are then computed from $q_i(h)$ and $v_i(h)$, and so on. A simple molecular dynamics implementation using the Hooke's law potential and the Euler integration algorithm is given online in both Fortran and C.[W]

Like all numerical integration algorithms, Euler's method suffers from *round-off* and *truncation* (or *discretization*) errors. Round-off errors result from the finite length of numbers within a computer, usually 32 bits for single precision and 64 bits for double precision, whereas truncation errors derive from the use of a finite or truncated expression to approximate the sum of an infinite series. A formal error analysis demonstrates that the overall, or *global*, numerical error associated with Euler's method is proportional to h, which means that the error decreases by a factor of 10 each time h is decreased by a factor of 10 (*4*). An error of this type is described as being *on the order of h*, or more simply as $O(h)$. The error is also *cumulative*, since each step uses values produced by the previous step. Because the error eventually becomes comparable in magnitude to the solution, Euler's method is rarely used in practice.

The most commonly employed numerical integration technique in molecular dynamics is probably the Verlet time integration algorithm (*1, 8, 10*), whose most basic form is

$$q_i(t+h) = 2q_i(t) - q_i(t-h) + \frac{h^2}{m} F_i(q_i(t)) \tag{14}$$

$$v_i(t) = \frac{1}{2h}\left(q_i(t+h) - q_i(t-h)\right) \tag{15}$$

Notice that two previous positions, rather than a single previous position and a single previous velocity, are required here for each calculation of $q_i(t+h)$ and that eq 15 provides the velocity at time t instead of $(t+h)$. Since the initial values for an MD problem usually consist of the positions and velocities of the particles at $t = 0$, an alternative formulation must be used for the first Verlet iteration. If we expand the displacement $q_i(t)$ in a Taylor series about h, then, after the first time step,

$$q_i(h) \approx q_i(0) + hv_i(0) + \frac{h^2}{2m} F_i(q_i(0)) \tag{16}$$

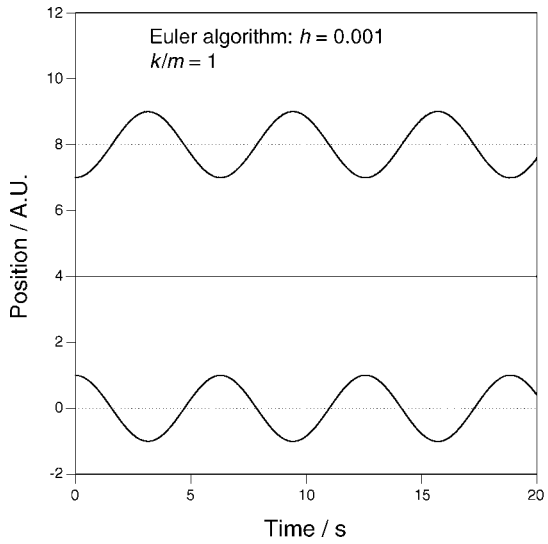

Figure 3. A molecular dynamics simulation of a linear chain of three particles interacting via the Hooke's law potential with k/m = 1. The equations of motion were integrated numerically by means of the Euler algorithm with h = 0.001. The starting conditions were $q_1(0) = -1$, $q_2(0) = 0$, $q_3(0) = 1$, $v_i(0) = 0$, $i = 1, 2, 3$.

Formal analysis reveals that the global error associated with the Verlet algorithm is $O(h^2)$ rather than $O(h)$, implying that the error decreases by a factor of 100, rather than 10, each time h is decreased by a factor of 10 (*4*). (In a later section, we provide a numerical illustration of the difference made by this single power of h.) In addition, h can be chosen so that the cumulative error remains small compared to the solution.

Molecular Dynamics of the Three-Particle Chain

We now wish to present an actual molecular dynamics simulation, so we return to our example of a linear chain of three particles interacting via the Hooke's law potential. Figure 3 shows the results of a 20-s simulation performed with k/m = 1 and h = 0.001 s. (Although this time step is convenient for our example, we emphasize that it is much larger than any real molecular period; the fundamental period of N_2, for instance, is about 14.3 fs (10^{-15} s) (*11*). Typically, we set h = 1 fs [*4*].) We employed the Euler time-integration algorithm and began the simulation with the conditions

$$q_1(0) = -1, \quad q_2(0) = 0, \quad q_3(0) = 1, \quad v_i(0) = 0, \quad i = 1, 2, 3$$

These conditions lead to dynamics known as the *symmetric stretch* of the chain (see below). The equilibrium positions of the particles are x = 0, 4, and 8; that is, d = 4 here.

To fully appreciate the trajectories described in Figure 3, we must understand something about the fundamental vibrational properties of our system (*4, 12, 13*). In general, a linear n-particle chain has n normal modes of vibration along its axis (one mode for each vibrational degree of freedom). A *normal mode of vibration* is a vibration for which (i) each participating particle executes simple harmonic motion about its equilibrium position, (ii) all participating particles oscillate with the same frequency, (iii) any two participating particles oscillate exactly in phase or exactly out of phase, and (iv) the center of gravity of the system remains unchanged (*13*). The normal modes of vibration for the three-particle chain are shown in Figure 4. The mode shown in Figure 4A has $\omega = 0$ and corresponds to simple translation of the system along its axis. (Because the equilibrium distance is maintained between each pair of neighboring particles, the potential V is always equal to V_{min} and there are no compressive or tensive forces.) The other two modes represent true oscillatory motions of the system and have nonzero frequencies that are proportional to $(k/m)^{1/2}$. In the mode pictured in Figure 4B, which is known as the *symmetric stretch*, the center atom remains at rest while the two outer atoms vibrate exactly out of phase with the same amplitude. Alternatively, in the *asymmetric stretch* of Figure 4C, the two outer atoms vibrate in phase with the same amplitude, and the center atom vibrates exactly out of phase with the outer atoms with an amplitude modulated by a factor of 2.

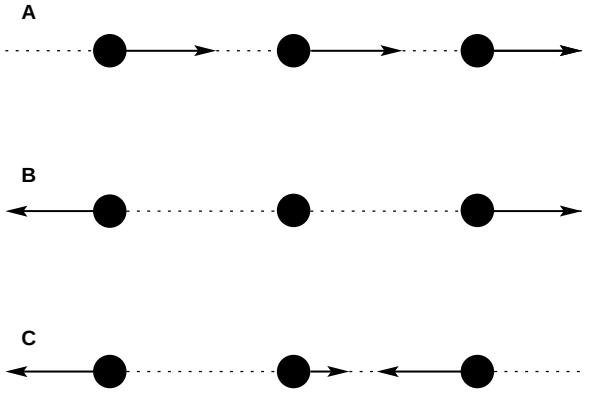

Figure 4. The longitudinal normal modes of vibration of a linear three-particle chain. A: Simple translation. B: Symmetric stretch. C: Asymmetric stretch.

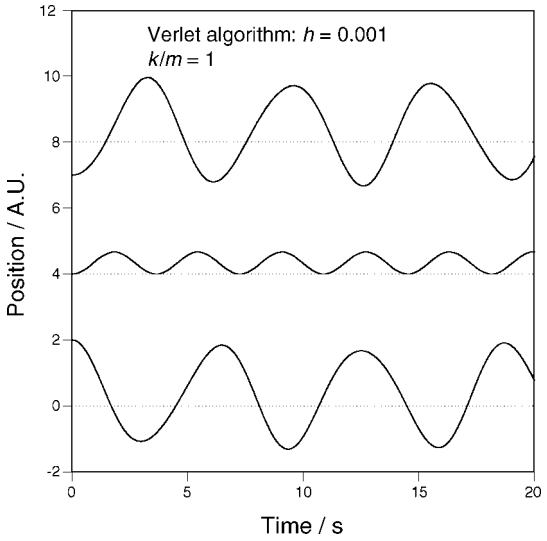

Figure 5. A molecular dynamics simulation of a linear chain of three particles interacting via the Hooke's law potential with k/m = 1. The equations of motion were integrated numerically by means of the Verlet algorithm with h = 0.001. The starting conditions were $q_1(0) = -1$, $q_2(0) = 0$, $q_3(0) = 2$, $v_i(0) = 0$, $i = 1, 2, 3$.

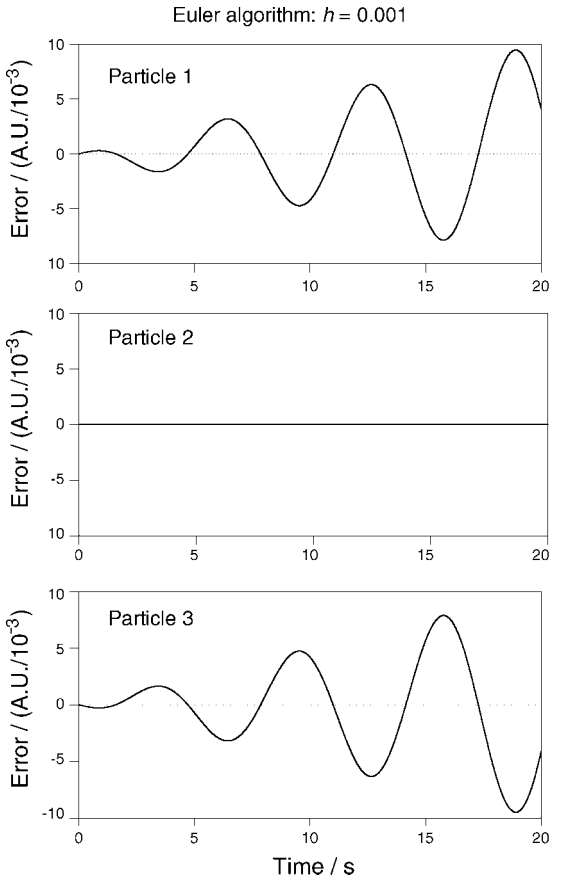

Figure 6. The errors accumulated by the Euler time integration algorithm for a 20-s simulation of the symmetric stretch mode of a three-particle linear chain with k/m = 1 and h = 0.001.

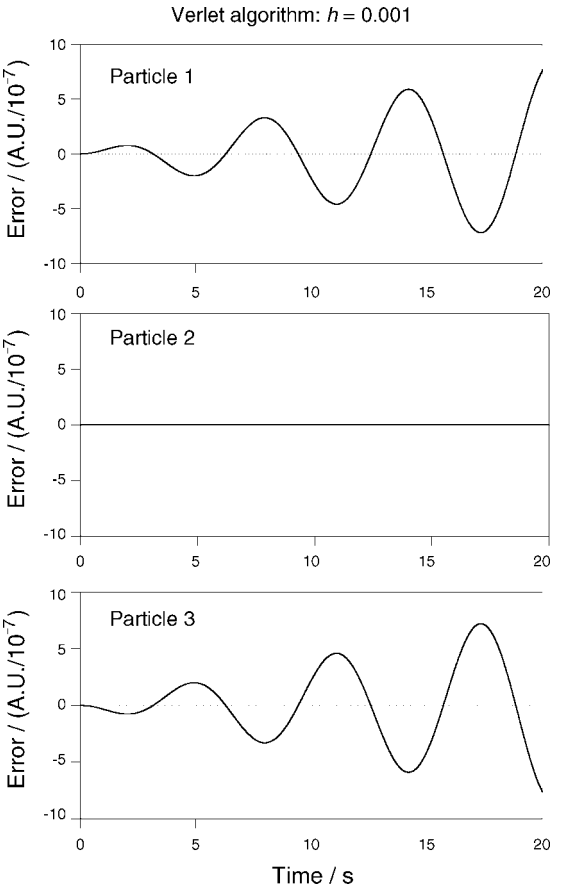

Figure 7. The errors accumulated by the Verlet time integration algorithm for a 20-s simulation of the symmetric stretch mode of a three-particle linear chain with k/m = 1 and h = 0.001. Note the difference in vertical scale compared to Figure 6.

Our description of the system in terms of its normal modes of vibration is advantageous in part because it can be shown that if the initial positions and velocities of the particles correspond to one of the normal modes, the system adopts that mode of vibration and remains in it forever (assuming no friction). Thus, the initial conditions chosen for our example simulation lead to the symmetric stretch mode of Figure 4B and we expect these dynamics to continue until the system is forcibly damped. However, the real importance of a *normal mode analysis* lies in the much more general property that any oscillatory motion of the system can be represented as a linear combination of the normal modes of vibration. This means that the dynamics of our linear chain of particles, no matter how complex or apparently random, can always be viewed as the sum of the individual modes shown in Figure 4, each weighted with appropriate amplitude and phase factors. Thus, we can consider the motion described in Figure 3 as being composed of a nonzero-valued contribution of the symmetric stretch mode and a zero-valued contribution of the asymmetric stretch mode of the three-particle chain. As an example of the dynamics resulting from initial conditions that do not correspond to a normal mode of vibration, Figure 5 depicts the motion of our test system with initial conditions

$$q_1(0) = -1, \quad q_2(0) = 0, \quad q_3(0) = 2, \quad v_i(0) = 0, \quad i = 1, 2, 3$$

In this case, the motion reflects nonzero-valued contributions from both the symmetric and asymmetric stretch modes of the chain. Notice that, in comparison with Figure 3, the middle particle now executes an oscillatory motion and the symmetry of the vibrations of each atom about its equilibrium position has been lost.

Cumulative Errors in the Three-Particle Simulation

One advantage of the Hooke's law potential is that it admits closed-form solutions to the classical equations of motion. For the n-particle chain, one solution has the form

$$q_i(t) = Q_{ij} \cos(\omega_j t + \delta), \quad i,j = 1, 2, 3, \ldots, n \qquad (17)$$

where Q_{ij} is an amplitude factor for particle i oscillating in mode j, ω_j is the angular frequency of mode j, and δ is a phase constant determined by the initial displacements and velocities. Q_j and ω_j are an eigenvector–eigenvalue pair and may be determined with some effort through matrix methods; the details are in Fosdick et al. (*4*) and Goldstein (*12*). If the amplitudes are normalized so that $\max|Q_{ij}| = 1$, the positions for the symmetric stretch (j = 2) of the three-particle system are given by

$$q_1(t) = \cos\left(\sqrt{\frac{k}{m}}\, t + \delta\right), \quad q_2(t) = 0,$$
$$q_3(t) = -\cos\left(\sqrt{\frac{k}{m}}\, t + \delta\right) \tag{18}$$

that is, $Q_{12} = -Q_{32} = 1$, $Q_{22} = 0$, and $\omega_2 = (k/m)^{1/2}$.

We can exploit this exact solution to compare the accuracies of our two numerical integration methods. Figures 6 and 7 present the errors accumulated by the Euler and Verlet time-integration algorithms, respectively, during a 20-s simulation of the symmetric stretch mode of our three-particle chain with $k/m = 1$, $h = 0.001$, and the initial conditions of Figure 3. (These initial conditions correspond to $\delta = 0$, as may be readily verified by solving for the phase constant when $t = 0$.) Recall that we characterized the error of the Euler algorithm as $O(h)$ and the error of the Verlet algorithm as $O(h^2)$. As expected, the error associated with the middle particle is always zero, because this particle is motionless during the symmetric stretch. In contrast, the error curves associated with the end particles are oscillatory and have maxima whose amplitudes clearly increase with time. The difference in accuracy between the two integration methods is quite striking: the maximum error associated with the Verlet algorithm (near $t = 20$) is about 10,000 times less than the maximum error associated with the Euler algorithm. (Note that the vertical scales of the two plots are different.) Repetitions of this simulation with different values of h confirm that for a given elapsed time, the maximum error of the Euler method increases linearly with increasing h and the maximum error of the Verlet method increases quadratically with increasing h.

Acknowledgments

This paper is based on work that was supported by National Science Foundation grants ACI-0072119 and CDA-9017953 (CISE Educational Infrastructure).

[W]Supplemental Material

A basic MD implementation using the Euler integration algorithm in both Fortran and C and a few suggested exercises for the code are available in this issue of *JCE Online*.

Literature Cited

1. Ercolessi, F. *A Molecular Dynamics Primer*; *http://www.fisica.uniud.it/vercolessi/md/md.html* (accessed Mar 2002).
2. Alder, B. J.; Wainwright, T. W. *J. Chem. Phys*. **1957**, *27*, 1208.
3. Alder, B. J.; Wainwright, T. W. *J. Chem. Phys*. **1959**, *31*, 459.
4. Fosdick, L. D.; Jessup, E. R.; Schauble, C. J. C.; Domik, G. *An Introduction to High-Performance Scientific Computing*; The MIT Press: Cambridge, MA, 1996; Chapter 15.
5. Grant, G. H.; Richards, W. G. *Computational Chemistry*; Oxford University Press: Oxford, 1995; pp 49–51.
6. Saiz, E.; Tarazona, M. P. *J. Chem. Educ.* **1997**, *74*, 1350.
7. Cotton, F. A.; Wilkinson, G.; Murillo, C. A.; Bochmann, M. *Advanced Inorganic Chemistry*, 6th ed.; Wiley: New York, 1999; pp 499–502.
8. Hockney, R. W.; Eastwood, J. W. *Computer Simulation Using Particles*; Adam Hilger: Philadelphia, 1988; Chapter 4.
9. See, for example, Cheney, E. W.; Kincaid, D. R. *Numerical Mathematics and Computing*; 4th ed.; Brooks/Cole: Pacific Grove, CA, 1999; pp 374–380.
10. Dove, M. T. *Introduction to Lattice Dynamics*; Cambridge University Press: Cambridge, 1993; pp 181–182.
11. Barrow, G. M. *Introduction to Molecular Spectroscopy*; McGraw-Hill: New York, 1962; p 42.
12. Goldstein, H. *Classical Mechanics*, 2nd ed.; Addison-Wesley: Reading, MA, 1980; Chapter 6.
13. Wilson, E. B. Jr.; Decius, J. C.; Cross, P. C. *Molecular Vibrations: The Theory of Infrared and Raman Vibrational Spectra*; Dover: New York, 1980; p 17.

†Present address: Chemistry and Chemical Engineering Department, BWXT Y-12, L.L.C. P.O. Box 2009, Oak Ridge, TN 37831.